

# **KOLABORATIVNÍ FILTROVÁNÍ**

## **DOPORUČOVÁNÍ PRODUKTŮ**

# Obsah

Popis projektu	3
Spôsob riešenia	3
Implementácia	4
Príklad výstupu	5
Experimentálna sekcia	7
Diskusia	8
Záver	8

## Popis projektu

Cieľom projektu bolo vytvoriť systém pre automatické odporúčanie produktov (v tomto prípade filmov) užívateľovi na základe jeho hodnotení. Odporúčané produkty sa vyhodnotia pomocou podobnosti medzi užívateľmi, ich hodnoteniami a ich zhodou v preferenciách.

## Spôsob riešenia

### Získanie a spracovanie dát

Na to, aby sa dali užívateľovi odporučiť filmy, ktoré by sa mu mohli páčiť, je potrebné tie filmy mať - rovnako ako aj informácie o jeho predošlých hodnoteniach a hodnoteniach ostatných užívateľov.

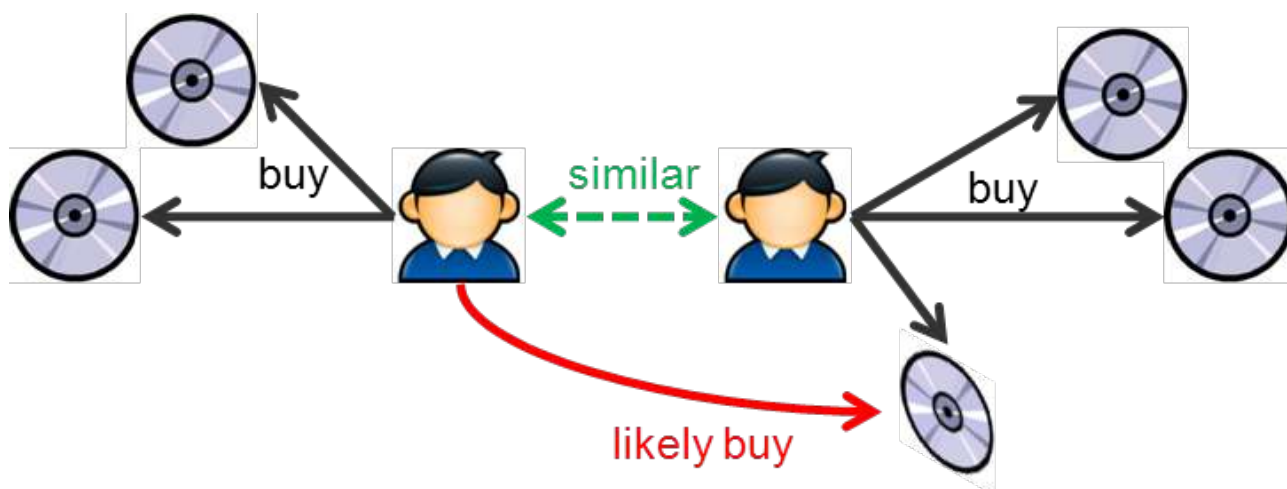
Ako dáta poslužil voľne stiahnuteľný dataset od [MovieLens.org](https://www.movielens.org/) obsahujúci informácie o viac než 9000 filmov a 100000 hodnotení od 600 užívateľov.

Informácie o filmoch však obsahovali iba názov filmu a jeho žánre, tak som sa rozhodol, pre lepší "užívateľský zážitok", stiahnuť pomocou [theMovieDatabase.org API](https://www.themoviedatabase.org/) dodatočné informácie - krátky popis filmu, fotku plagátu..

Získané dáta z datasetu a tMDB API som importoval do SQLAlchemy databáze, pre jednoduchšiu prácu vo webovej aplikácii Flask

### Odporúčanie filmov

Na odporúčanie filmov užívateľovi X som zvolil tzv. user-based model, ktorý hľadá K podobných užívateľov k užívateľovi X na základe ich hodnotení, vytriedi filmy, ktoré užívateľ X nevidel a podľa ohodnotenia týchto filmov mu ich následne odporúči.



Zdroj: <https://dzone.com/articles/recommendation-engine-models>

Na nájdenie KNN (K nearest neighbours) je treba vypočítať podobnosť medzi užívateľom X a ostatnými užívateľmi, vybrať K najbližších, a zobrať vážený priemer ich hodnotení s podobnosťou:

$$r_{ij} = \frac{\sum_k \text{Similarities}(u_i, u_k) r_{kj}}{\text{number of ratings}}$$

Zdroj: <https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26>

Na vypočítanie podobnosti medzi užívateľmi som sa rozhodol použiť kosínusovú podobnosť:

$$\text{Cosine Similarity : } \text{Sim}(u_i, u_k) = \frac{r_i \cdot r_k}{|r_i| |r_k|} = \frac{\sum_{j=1}^m r_{ij} r_{kj}}{\sqrt{\sum_{j=1}^m r_{ij}^2 \sum_{j=1}^m r_{kj}^2}}$$

Zdroj: <https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26>

## Implementácia

Po načítaní úvodnej stránky (počas čoho sa aj načítavajú dáta z flask-sqlalchemy databáze pre plynulejší chod) je užívateľovi sprístupnené prihlásenie sa pomocou ID (1-610, podľa datasetu).

Z úvodnej stránky sa taktiež dá dostať do administratívneho modulu (alebo prejdéním na adresu /admin), kde sa dá prezerať/meniť/mazať hodnotenia užívateľov a taktiež zmeniť parameter K - pre hľadanie najbližších susedov (default - 30)

## Webová aplikácia

Webová aplikácia je napísaná pomocou jednoduchého HTML + CSS + JavaScript template-u a frameworku Flask

## DataTables

jQuery plugin pre jednoduchšie zobrazovanie položiek, pohybovanie sa medzi nimi a prípadne vyhľadávanie v nich

## Dáta

### MovieLens Dataset

Bezplatný dataset reálnych dát s hodnoteniami užívateľov zo stránky MovieLens.org

### The Movie Database API

Bezplatná API na získavanie informácií o filmoch

## System odporúčania

Back-end aplikácie je napísaný v programovacom jazyku Python (3.6 a vyššie) s použitím knižníc na pomoc pre ukladanie a prácu s dátami

### Flask

WSGI framework pre webové aplikácie prepojené s Pythonom

### SQLAlchemy

Python SQL toolkit, využívaný s Flask, na ukladanie získaných dát

### Numpy

Python knižnica implementujúca mnoho-dimenzionálne polia a matice a výpočty s nimi

### Pandas

Python knižnica na prácu s uloženými dátami a na ich analýzu

## Príklad výstupu

Po prihlásení užívateľa sú mu zobrazené doposiaľ ohodnotené filmy, odporúčané filmy a filmy v databáze

Movie Recommender

Ratings of user: 2



Good Will Hunting  
Your rating:



LOVE & OTHER DRUGS  
Your rating:



JACK AND JILL  
Your rating:



GLOW: The Story of the Gorgeous Ladies of Wrestling  
Your rating:



A PIGEON SAT ON A BRANCH REFLECTING ON EXISTENCE  
Your rating:

First Previous **1** 2 3 4 5 6 7 Next Last

Recommendations for user: 2

Time with preprocessing the data: 5.914287090301514 s

Time after preprocessing the data: 2.570535182952881 s




The Return of Martin Guerre  
Score: 4.6707645146641585  
Rate the movie:



Strange Wilderness  
Score: 4.56178344110751  
Rate the movie:



THE LAST WINTER  
A FILM BY LARRY FESSENDEN  
THE SCARIEST MOVIE OF THE YEAR  
Score: 4.550893450143293  
Rate the movie:



GHOST IN THE SHELL  
Score: 4.53884832601861  
Rate the movie:



Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb  
Score: 4.530030934410581  
Rate the movie:

First Previous **1** 2 Next Last

All movies:

Search:



Toy Story  
Rate the movie:



Jumanji  
Rate the movie:



GRUMPIER OLD MEN  
Rate the movie:




Waiting to Exhale  
Rate the movie:



FATHER OF THE BRIDE PART II  
Rate the movie:



HEAT



Sabrina



JONATHAN TAYLOR THOMAS  
BRAD RENFRO  
A lot of kids get into trouble... These two invented it.  
Tom and Huck  
For Oldies, For Best.



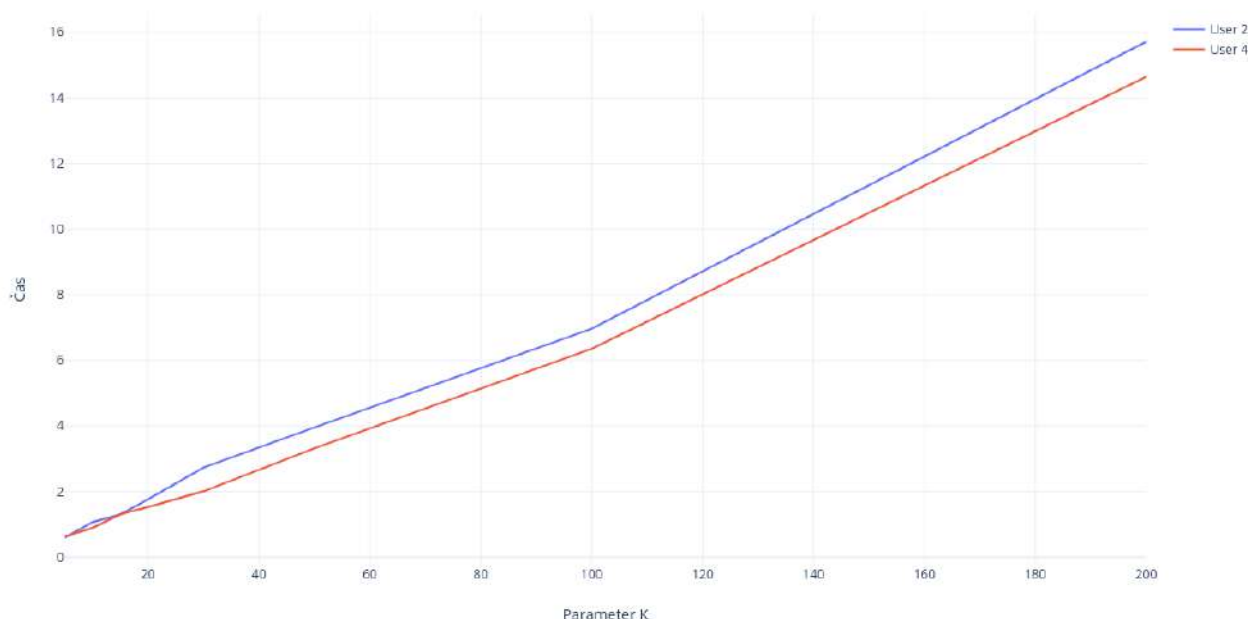
Sudden Death  
TERROR GOES INTO OVERTIME



GOLDENEYE 007

## Experimentálna sekcia

Z dôvodu, že dáta o užívateľoch sú uložené v datasete a API TheMovieDatabase nebola natolko rýchla, aby sa ju dalo využiť na sťahovanie dát v reálnom čase, rozhodol som sa ponechať dáta pasívne, a tým pádom som sa sústredil na čas, ktorý je potrebný na odporúčenie filmov v závislosti od počtu najbližších susedov (K parameter)



To, že čas potrebný na výpočet pri väčšom parametre K, je pochopiteľné, pretože čím viac ľudí je užívateľovi podobných, tým sa zväčšuje pole filmov, ktoré ešte nevidel a musí sa pre ne vypočítať skóre.

Avšak čas, ktorý je potrebný na výpočet odporúčení závisí aj od počtu filmov, ktoré daný užívateľ videl - User 2 videl iba 31 filmov, pričom User 4 ich videl 216. Čas výpočtu sa pri väčších číslach K líši síce len o 1 sekundu, pretože pole filmov, ktoré užívateľ nevidel, sa vyrovná (a nezáleží, koľko filmov videl), no pri malých číslach K je rozdiel viditeľnejší: pri default hodnote  $K = 30$  trvá výpočet pre Usera 2 2.75 sekúnd, pričom len 1.9 sekúnd pre Usera 4 - takmer 50% rozdiel

## Diskusia

Využívanie SQLAlchemy pre uskladňovanie dát je citelné najmä pri načítavaní úvodnej stránky - trvá to niekoľko sekúnd prázdneho bufferovania, kde človek nevie, či je problém na jeho strane, alebo na strane serveru. To isté sa dá povedať aj o používaní Pandas dataFramu - práca s tabuľkami s desaťtisícmi záznamov nie je ideálna.

## Záver

Práca na tomto projekte mi dala veľa, nie iba vedomosti z problematiky algoritmov pre spracovanie dát, ale aj samotnej tvorby webových aplikácií a zistenie, že pracovať sám nie je vždy ideálne.