Vysoké učení technické v Brně

# Dokumentácia

Projekt k predmetu ISJ

Tibor Dudlák, xdudla00

# Obsah

# Úloha 1. : Sťahovanie dát z twitteru

Prvou úlohou bolo napísať v Pythone alebo Ruby skript, ktorý bude sledovať aktivitu vybraného účtu na twitteri a ukladať pri zverejnení odkazov v tweete aj ich obsahy. Vybral som si jazyk python a pri vývoji som pracoval s verziou 2.7.6 .

**Použité moduly:**

Pri riešení tejto úlohy som pracoval s modulom tretej strany a to *tweepy* viac informácii nájdete v tomto odkaze. Ďalej som použil moduly *os , csv, sys, urllib3* a *deque*.

**Riešenie:**

Pre prácu s modulom tweepy je potrebné získať tokeny, ktoré budú vygenerované hneď po registrácii aplikácie a následne ich použiť pri autentifikácii na začiatku skriptu, respektíve pred tým, ako začneme používať API.

Hneď pri spúšťaní skriptu je potrebné zadať parameter, ktorý prezentuje meno účtu twitteru, nakoľko je riešenie univerzálne pre akýkoľvek účet a práca s rôznymi menami nieje vďaka API žiaden problém. Teda nasledovný syntax spustenia vyzerá v mojom prípade:

```
./tweets.py foxnews
```

resp.: `python tweets.py foxnews`

Po korektnom spustení , teda ak účet so zadaným menom existuje, skript zisťuje či sa mu podarí v zložke v ktorej sa nachádza nájsť súbor, ktorý obsahuje tweety meno súboru som zvolil jednoducho `A_tweets.csv` kde A je meno účtu, naopak vypíše chybu.

Ak súbor neexistuje, skript sa spustí v režime štahovania posledných 50 tweetov v histórii účtu a vytvorí tento súbor, kde následne zapíše vo vhodnom formáte 50 stiahnutých tweetov. Pri každom tweete si do zoznamu ukladám jeho ID, dátum a čas vytvorenia, text a zároveň linky na externé odkazy. Po uložení tweetov pokračuje skript v sťahovaní externých odkazov z tohto zoznamu do zložky ktorú si vopred vytvorí ak už náhodou neexistuje. Jediný problém vznikol tu, nakoľko tweet môže obsahovať aj viac externých odkazov a ja som ukladal ich obsahy podľa dátumu a času pridania, takže ak tam bolo viac odkazov prepisoval sa stále ten istý súbor. Vyriešil som to pridaním k názvu súboru pár posledných znakov za posledným znakom `'/'` v odkaze.

Ak súbor existuje, tak je skript spustený v režime update. Z posledného riadku súboru kde je aj najnovší tweet si prečíta jeho ID a nesleduje stiahnutie a uloženie maximálne `200` tweetov, ktoré nasledujú hneď po tweete, ktrorý je evidovaný ako najnovší v súbore. Ak je veľkosť zoznamu stiahnutých tweetov `'0'` teda je prázdny program skončí. Sťahovanie externých odkazov funguje rovnako ako v režime sťahovania tweetov.

Doba sťahovania skriptom sa líši počtom externých odkazov a ich obsahom.

# Úloha 2. : Sťahovanie príspevkov z fóra.

Ďalšou úlohou bolo napísať skript, ktorý stiahne všetky doterajšie príspevky z vybraného diskusného fóra a ukladá ich do súboru spolu s relevantnými informáciami. Skript by mal dokázať zistiť nové príspevky od posledného stiahnutia a spustiť ich sťahovanie.ň

**Použité moduly:**

Pri riešení tejto úlohy som použil moduly *re*, *os*, *csv*, *shutil*, *zipfile*, *urllib3*, *requests* a *BeautifulSoup* čo je modul tretej strany. Viac informácií v tomto odkaze.

**Riešenie:**

Moje riešenie neobsahuje funkciu update ktorá by mala zistiť nové príspevky a začať ich sťahovanie do odpovedajúcich súborov. Spôsob riešenia, ktorý som implementoval zahrňuje postupné spracovávanie linkov na témy a vlákna a následne prechádzanie obsahu príspevkov, čo by bolo potrebné spraviť aj vo funkcii update, teda iterovať nad celým fórom. Taktiež spôsob ukladania príspevkov do jednotlivých súborov neobsahuje dostatok informácií na to, aby bol update časovo efektívny a nezaistil by spracovávanie v prípade, ak by bol príspevok vymazaný.

Hneď po spustení skriptu si pomocou css selectorov vyberiem z hlavnej stránky linky na všetky témy a ich názvy. Problém vznikal že v zozname boli aj odkazy na facebookovu stránku a na označenie tém ako prečítané a podobne. Vyriešil som to tak, že som si spravil list `bugs[]`, ktorý obsahoval názvy k linkom, ktoré neboli potrebné. Pri iterovaní v tomto liste som preskakoval sťahovanie obsahu týchto linkov, aby nevznikali chyby jednoduchým :

```
if (topic[name] in bugs):
    continue
```

čo zabezpečilo to, že for pokračoval v ďalšej iterácii bez toho, aby vykonal to čo je v jeho tele.

Podobný problém nastal pri opätovnom sťahovaní vlákien, ktoré boli na každej strane témy pripnuté. Vyriešil som to podobne, po stiahnutí obsahu vlákna som opätovne uložil jeho odkaz do zoznamu `downloaded` a v ďalšej iterácii zisťoval, či bolo vlákno už stiahnuté :

```
if (thread[link] in downloaded):
    pass
```

kde sa po nesplnení podmienky vykonávalo telo else.

Nakoniec som pomocou `Soup.find_all()` stiahol zo stránky vlákna dátum a čas pridania príspevku, samotný príspevok a samozrejme autorov login do listov nad ktorými som iteroval a postupne spájal do výsledného listu. Informácie v tomto liste som ukladal spolu s názvom témy do súboru typu `.csv` podobne ako v prvej úlohe.

**Detail riešenia:**

Skript si vytvorí dočasnú zložku `Forum_content`, v ktorej si postupne vytvára ďalšie zložky s názvami tém a do nich postupne ukladá súbory. Tieto súbory prezentujú celý obsah vlákna. Sú pomenované menami vlákien a sú v zložke ktorá presne zodpovedá téme v ktorej sa dané vlákna nachádzajú.

Po skončení sťahovania dát z diskusného fóra postupne zabalím obsah tejto zložky do archívu `Forum_content.zip` a hneď potom zmažem zložku s celým obsahom. Ak teda skript pracoval presne podľa popísania zmazal dočasnú zložku a výsledkom je iba archív v zložke v ktorej bol skript spustený.

Skript funguje lineárne čo znamená aj pomalšiu činnosť, zrýchliť by sa to dalo zavedením vlákna napríklad pre každú tému. Doba spustenia skriptu na mojom systéme bola takmer dve hodiny presnejšie *123 minút a 12 sekúnd* ( pri spustení `time ./forum.py` ) .