Metody i narzędzia analizy dużych zbiorów danych

**Projekt 1**

*Statystyczna analiza danych z wykorzystaniem Hadoop MapReduce*

Wykonali:

*Paweł Suchanicz*

*Anna Zybek*

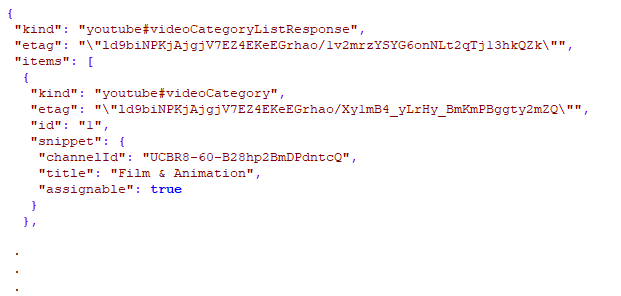
1. **Temat projektu**

Tematem projektu była implementacja aplikacji importującej dane z dowolnego statycznego źródła do systemu HDFS. Aplikacja ta miała za zadanie dokonać statystycznej analizy danych przy użyciu platformy Hadoop. Aplikacja wykorzystuje MapReduce do wyliczenia średniej ilości interakcji w filmach na platformie youtube w zależności od kategorii filmów.

1. **Opis danych**

Wybrane przez nas dane opisują najpopularniejsze filmy na platformie YouTube. Dane zawarte są w plikach CSV i w plikach JSON. Zestaw danych zawiera dane z kilku miesięcy pewnego okresu na temat popularnych filmów z YouTube. Do pobrania danych wykorzystane było API youtube pobierające informacje na temat filmów z karty “na czasie”. Dane uwzględniają różne regiony, dla każdego regionu dane zapisane są w osobnym pliku. Dane zawierają takie informacje, jak: tytuł filmu, tytuł kanału, czas publikacji, tagi, wyświetlenia, polubienia i nielubienia, opis i liczbę komentarzy. Dane zawierają również pole category\_id, które różni się w zależności od regionu. Aby pobrać kategorię dla konkretnego filmu, należy odszukać go w powiązanym pliku JSON.

**Struktura pliku JSON**



**Struktura pliku CSV**

|  |  |  |
| --- | --- | --- |
| Kolumna | Opis | Przykład |
| video\_id | Id filmu | jVpp2BUtMoY |
| trending\_date | Data pojawienia się na karcie “trending” | 17.14.11 |
| title | Tytuł filmu | So We Started A Fire... |
| channel\_title | Nazwa kanału | Philly D |
| category\_id | Id kategorii (id są różne dla różnych krajów) | 22 |
| publish\_time | Data opublikowania filmu | 2017-11-13T17:00:04.000Z |
| tags | Tagi przypisane do filmu | "PhillyD"|"Philly D"|"Vloggity"|"The Philip DeFranco Show"|"Philip DeFranco Show"|"DeFrancoElite" |
| views | Liczba wyświetleń filmu | 92280 |
| likes | Liczba kliknięć w przycisk “To mi się podoba” pod filmem | 11045 |
| dislikes | Liczba kliknięć w przycisk “To mi się nie podoba” pod filmem | 182 |
| comment\_count | Liczba komentarzy | 1412 |
| thumbnail\_link | Link do obrazka będącego miniaturką filmu | https://i.ytimg.com/vi/jVpp2BUtMoY/default.jpg |
| comments\_disabled | Informacja czy film ma wyłączone komentarze - boolean | False |
| ratings\_disabled | Informacja czy film ma wyłączone oceny - boolean | False |
| description | Opis umieszczony przez autora pod filmem | Happy World Kindness Day! ... |

Link do pobrania danych: <https://www.kaggle.com/datasnaek/youtube-new>

Rozmiar danych: 514 MB (380327 rekordów).

Dane w zbiorze były zebrane przy pomocy publicznego API udostępnianego przez youtube (w powyższym linku załączono skrypt wykorzystujący API do pobierania danych).

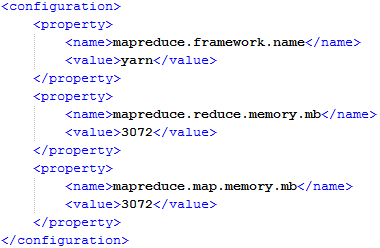
1. **Przeprowadzona statystyka**

W naszym projekcie przeprowadzona została następująca statystyka: dla każdej kategorii obliczono średnią wszystkich interakcji użytkowników. Jako interakcje brane są pod uwagę kliknięcia „lubię to”, „nie lubię” oraz liczba komentarzy dla danego filmu, należącego do danej kategorii filmów. Stąd można wnioskować, jakiej kategorii filmy na platformie YouTube najbardziej angażują widownię.

1. **Przygotowanie środowiska**

Do stworzenia środowiska w którym uruchamiana jest aplikacja została wykorzystana platforma Docker. Jest to narzędzie pozwalające na wirtualizację systemu operacyjnego w tzw. kontenerach. Wykorzystano gotowy obraz (Debian 8.3.0-6) z zainstalowanym Hadoopem w wersji 2.7.1 dostępny na platformie Dockerhub (<https://hub.docker.com/r/sequenceiq/hadoop-docker/>). Hadoop uruchamiany w trybie pseudo-distributed. Obraz został zmodyfikowany na potrzeby naszej implementacji aplikacji MapReduce. W tym celu w pliku *Dockerfile* dodano linie dodające do obrazu pliki z danymi, implementację mappera i reducera w pythonie oraz skrypt startujący.Rozwiązane zostały także napotkane problemy związane z budowaniem obrazu (m.in. inne kodowanie znaków końca linii w różnych systemach operacyjnych - CRLF dla Windows, LF dla Linux, wymiana wersji javy na openjdk 1.8, czyszczenie cache po użyciu *yum*).

Ze względu na to że wykorzystany zbiór danych był dość duży domyślna ilość pamięci dla zadań map oraz reduce (1GB) okazała się niewystarczająca. Limit pamięci został więc zwiększony do 3GB poprzez dodanie odpowiednich property w pliku *mapred-site.xml*:



1. **Implementacja**

Do wykonania projektu wykorzystano wspomniany już Docker oraz język Python w wersji 2.6.6.

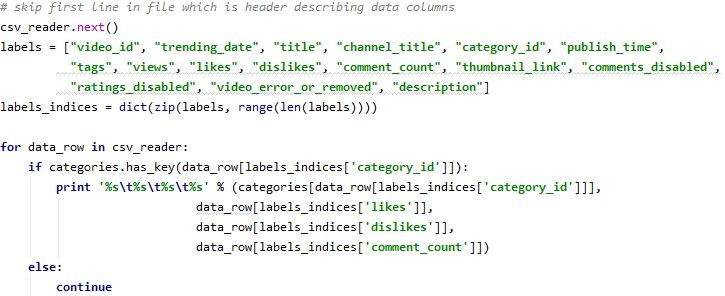
1. Mapper

Zadaniem mappera było wczytanie danych z plików i wydobycie z nich istotnych dla naszego problemu informacji czyli kategorii filmu, liczby kliknięć „podoba ”, liczby kliknięć „ nie podoba mi się” oraz liczbę komentarzy. Należało uwzględnić różne wartości category\_id tej samej kategorii w różnych regionach.



*Rys. 1 Wczytanie danych z plików*

Do wypisania wyników dla każdej kategorii filmu, użyty został słownik, przechowujący klucz-wartość, gdzie klucz to cechy opisujące dany film a wartość to odpowiednia wartość (np. liczba polubień).

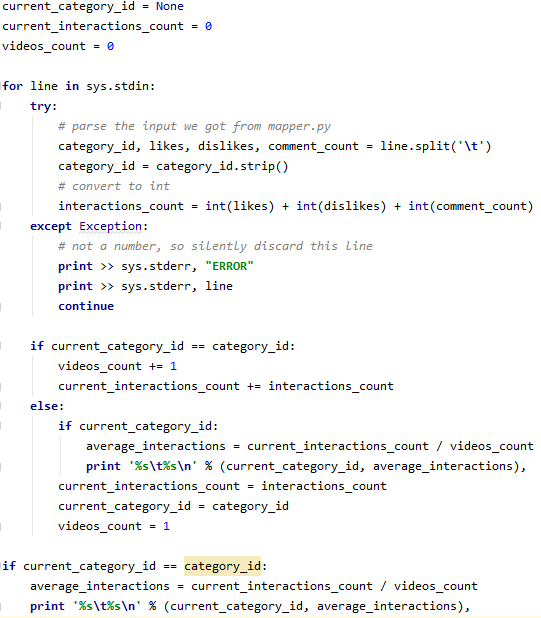


*Rys. 2 Wynikiem mappera są rekordy przechowujące informacje o filmie:kategoria filmu, liczba kliknięć ‘Lubię to”, liczba kliknięć „nie lubię”, liczba komentarzy*

1. Reducer

Reducer na wejściu przyjmuje wyniki otrzymane z mappera, przegląda wszystkie pojedyncze rekordy posortowane wg kategorii, w których znajdują się liczby: kliknięć ‘lubię to’, kliknięć ‘nie lubię’ oraz komentarz. Dla każdej kategorii oblicza średnią wszystkich interakcji użytkowników.

Wynikiem Reducera jest nazwa kategorii i średnia wszystkich interakcji użytkownika dla danej kategorii filmu.



*Rys. 3 Reducer zwracający średnią wszystkich interakcji użytkowników dla danej kategorii filmów*

1. Skrypt uruchamiający



*Rys. 4 Skrypt uruchamiający aplikację MapReduce*

Powyższy kod wykorzystuje funkcję *check\_call* z modułu *subprocess* do uruchamiania komend w linii poleceń. Funkcja czeka na wykonanie komendy podanej jako argument i w razie niepowodzenia rzuca wyjątek *CalledProcessorError*. Wywołujemy tutaj następujące komendy:

* utworzenie katalogu z danymi wejściowymi w hdfs
* skopiowanie danych do systemu hdfs
* uruchomienie zadań mappera i reducera z wykorzystaniem hadoop-streaming
* skopiowanie wyników z hdfs do lokalnego systemu

1. **Uruchomienie**

**Pobranie danych:**

* Pobrać zbiór danych <https://www.kaggle.com/datasnaek/youtube-new>
* Wypakować archiwum
* Umieścić pliki csv w katalogu /data/youtube-statistics/trending-statistics
* Umieścić pliki json w katalogu /data/youtube-statistics/categories

**Budowanie obrazu:**

docker build -t my-hadoop-docker .

**Uruchomienie kontenera na Docker:**

docker run -it -p 8088:8088 -p 50070:50070 my-hadoop-docker /etc/bootstrap.sh -bash

**Uruchomienie aplikacji MapReduce:**

./mapreduce/map-reduce.py

1. **Wynik działania**

Autos & Vehicles 11404

Comedy 45686

Education 17300

Entertainment 32233

Film & Animation 32512

Gaming 31894

Howto & Style 21705

Movies 30874

Music 191510

News & Politics 6373

Nonprofits & Activism 402365

People & Blogs 15726

Pets & Animals 13614

Science & Technology 38200

Shows 6052

Sports 19978

Trailers 41

Travel & Events 9276