Ingegneria dei dati: Homework 2

Davide Gattini

Ottobre 2022

1 Analyzers

Per il campo ***filename*** si è scelto l'uso di un *StandardAnalyzer* così caratterizzato:

- Tokenizer
  - StandardTokenizer
- Token filter
  - LowerCaseFilter

La scelta deriva dalla volontà di permettere all'utente la ricerca delle canzoni tramite il loro titolo, ignorando che quest'ultimo possa contenere lettere maiuscole e/o una punteggiature, infatti colui che ricerca i titoli delle canzoni si suppone lo faccia senza pensare a questi dettagli.

Mentre per il campo ***content*** viene applicato un *CustomAnalyzer*:

- Tokenizer
  - StandardTokenizer
- Token filter
  - LowerCaseFilter
  - StopFilter (stop words = {*of*, *an*, *a*, *the*, *for*})
  - EnglishPossessiveFilter

L'adozione di un analyzer di questo tipo per il lyrics delle canzoni nel corpus, permette di evitare l'indicizzazione di termini molto comuni che saranno quindi ignorati in fase di ricerca, poiché si punta ad offrire all'utente il maggior numero possibile di documenti, garantendo efficienza nell'indicizzazione. Inoltre, se ne è fatto uso solo per il contenuto poiché quest'ultimo, essendo più articolato del titolo, presenterà molti di questi termini, in questo modo si è evitato che gran parte dei documenti fossero recuperati inutilmente.

In più sono state ignorate le maiuscole, i vari delimitatori ed anche la punteggiatura, per evitare che ne sia richiesta la conoscenza a chi esegue la ricerca.

L'ultima considerazione riguarda la presenza di un filtro per il genitivo sassone, poiché è stato considerato come un dettaglio da poter trascurare e quindi non indicizzare.

Infine per la ***query***, poiché si è usato il query parser, è stato necessario utilizzare un ulteriore analyzer e la scelta è ricaduta sempre su di un *StandardAnalyzer*. Da sottolineare che non sono state definite delle stop word, così da permettere all'utente di inserire una phrase query contenente questi termini (qualora si ricordasse il titolo della canzone di cui vuole leggere il testo): si suppone che il titolo cercato sia più facile da ricordare rispetto al contenuto. Da notare che, grazie all'analyzer del contenuto, sebbene questi termini siano indicizzati per la query, non troveranno una corrispondenza nel testo indicizzato di una canzone (se appartenenti all'insieme di stop word definito sopra), soddisfacendo l'obiettivo definito precedentemente, cioè quello di permettere il retrieval di più documenti possibili.

2 File indicizzati e tempo di indicizzazione

I file che sono stati indicizzati sono stati recuperati tramite l'uso dell'API *lyricsGenius*, la quale, tramite uno script python, ha permesso l'estrazione e il salvataggio dei testi delle 100 canzoni più popolari dei Coldplay. Di conseguenza, il nome di ogni file corrisponde al nome della canzone (a cui sono stati eliminati i caratteri speciali) e nel programma se ne fa riferimento con il campo *filename*, mentre il lyrics associato è stato salvato in formato testuale e assegnato al campo *content*.

Tempo d'indicizzazione per i-esima esecuzione									
1°	2°	3°	4°	5°	6°	7°	8°	9°	10°
$49ms$	$50ms$	$47ms$	$46ms$	$43ms$	$44ms$	$41ms$	$39ms$	$42ms$	$45ms$

Table 1: in questa tabella sono stati riportati i tempi impiegati dal programma per indicizzare i documenti nel corpus: si ha una media di $\sim 45ms$.

3 Query testate

Di seguito saranno elencate le query testate, suddivise per il campo su cui sono state effettuate. Prima di farlo è utile sottolineare che le query, lette da riga di comando, sono processate dal programma tramite il *QueryParser*, che riesce a comprendere se l'utente ha inserito una *PhraseQuery* e, più in generale, ad eseguire il parsing della stringa in input (permettendo, inoltre, l'utilizzo di alcuni simboli (e.g. `+` e `-`, vedi 3.1.3).

Si noti che qualora la query corrisponda ad una stringa vuota, verrà processata senza riportare documenti (questo comportamente è permesso dall'uso della funzione *MatchNoDocsQuery()*).

3.1 Campo: *filename*

3.1.1 Query 1

```
filename: "A Sky Full Of Stars"
```

Questa *PhraseQuery* è stata utilizzata per testare che le maiuscole vengano correttamente ignorate, poiché sia per il filename che per il content, i vari termini sono stati indicizzati con tutte le lettere minuscole.

	DocID	Documenti recuperati
1°	38	A Sky Full of Stars

3.1.2 Query 2

```
filename: "a sky full of stars"
```

In questo caso si è testato, tramite una *PhraseQuery* con solo minuscole, che il risultato della query precedente fosse uguale alla query in questione.

	DocID	Documenti recuperati
1°	38	A Sky Full of Stars

3.1.3 Query 3

```
filename: -sky +full
```

La *TermQuery* corrente è caratterizzata da simboli usati come prefissi dei singoli termini e, in questo caso, la query, grazie al *QueryParser*, viene riconosciuta per eseguire una ricerca dei file cui titolo presenta necessariamente "full", ma che non deve contenere "sky".

	DocID	Documenti recuperati
1°	4	A Head Full of Dreams

3.1.4 Query 4

```
filename: LOVE
```

Si è testata l'indipendenza dalle maiuscole anche quando viene passato un solo termine, quindi per verificare che i documenti trovati siano relativi a file cui nome contiene la stessa parola in qualsiasi formato.

	DocID	Documenti recuperati
1°	26	True Love
2°	37	Lovers in Japan Reign of Love

3.1.5 Query 5

```
filename: The Scie
```

Per verificare la correttezza del sistema di ranking e di ricerca, è stato controllato che, passato un termine troncato, vengano comunque trovati i file cui nome include parole "vicine" al termine cercato. Si noti che, avendo indicizzato parole comuni come "The", vengono anche riportati tanti altri documenti, ma grazie al ranking vengono valutati meno del file ricercato, cioè *The Scientist.txt*

	DocID	Documenti recuperati
1°	44	The Scientist
2°	30	The Hardest Part
3°	23	Hymn for the Weekend
4°	52	People of the Pride
5°	70	Champion of the World
6°	72	X Marks the Spot
7°	86	Us Against the World
8°	87	Swallowed in the Sea
9°	83	Hymn for the Weekend Seeb Remix
10°	61	Hymn for the Weekend Alan Walker Remix

3.2 Campo: *content*

Successivamente sono stati verificati gli stessi concetti di cui sopra (3.1) anche per il campo *content*. Per cui non verranno ripetute le motivazioni per cui sono state eseguite le seguenti query, piuttosto saranno riportati solamente i risultati relativi.

3.2.1 Query 6

```
content: "love you"
```

	DocID	Documenti recuperati
1°	3	True Love
2°	82	Yellow
3°	16	Flags
4°	9	X Y
5°	37	Lovers in Japan Reign of Love
6°	65	A Message
7°	68	Biutyful

3.2.2 Query 7

```
content: fix
```

	DocID	Documenti recuperati
1°	39	Fix You
2°	9	X Y
3°	8	Politik

3.2.3 Query 8

```
content: "Everything you want"
```

	DocID	Documenti recuperati
1°	28	Adventure of a Lifetime
2°	68	Biutyful

3.2.4 Query 9

```
content: I could not stop
```

	DocID	Documenti recuperati
1°	18	Gravity
2°	25	Clocks
3°	87	Swallowed in the Sea
4°	30	The Hardest Part
5°	12	Speed of Sound
6°	49	Everythings Not Lost
7°	39	ix You
8°	62	Arabesque
9°	16	Flags
10°	52	People of the Pride

3.2.5 Query 10

```
content: want's dream
```

	DocID	Documenti recuperati
1°	28	Adventure of a Lifetime
2°	11	Paradise
3°	66	Eko
4°	70	Champion of the World
5°	16	Flags

4 Github

Link del progetto github: `https://github.com/Gatto99/DataEngineering`