Prima Esercitazione
Web Intelligence

Rosada Fabio 851772

December 10, 2016

Indice

1 Introduzione

L'esercitazione è stata svolta su più file, per consentirne una più facile comprensione. Per farla partire sarà sufficiente lanciare **main.py**, che a sua volta si occuperà di lanciare il resto.

Gli articoli sono salvati all'interno della cartella **articoli/** presente nella directory del progetto.

2 Parte A

La parte A è svolta nella sua interezza all'interno del file theverge_downloader.py

2.1 Generazione url archivio

Per lo svolgimento della parte A dell'esercitazione, il sito scelto è stato www.theverge.com. Il sito offre una struttura ad archivio, quindi per prima cosa ho generato gli url delle pagine dell'archivio da cui successivamente recuperare i link degli articoli. Questa prima parte è stata abbastanza semplice in quanto per generare gli url è bastato aggiungere **/[categoria]/archives/[numero pagina]** allo url principale del sito.

2.2 Recupero dei link degli articoli

Una volta fatto questo, aprendo le singole pagine tramite **urllib2** è bastato individurare la sezione contentente i link agli articoli tramite **BeautifulSoup** e recuperare il testo dei tag contenenti l'attributo **href**. Una volta recuperati i link, li scrivo in un file, così se il download dei 1000 articoli dovesse essere interrotto, al prossimo avvio basterebe recuperare i link dal file senza "scansionare" nuovamente tutte le pagine dell'archivio.

2.3 Salvataggio articoli

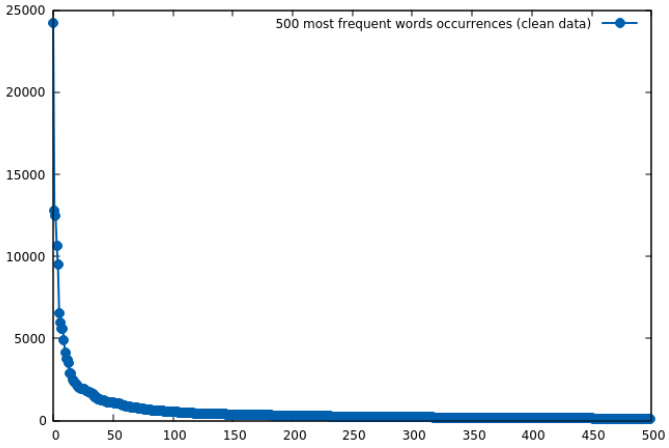
Analogamente, una volta recuperati i link dei 1000 articoli, tramite **BeautifulSoup** ho recuperato Titolo, Autore e Testo dell'articolo, e per ogni articolo ho creato un file che ha come prima riga il link dell'articolo, in modo da poterlo recuperare facilmente, come seconda riga il titolo, come terza l'autore e tutte le rimanenti sono dedicate al testo. Il nome del file è stato creato tramite la libreria **SHA** che mi consente di fare l'hash dello url, e di ottenere così un nome unico (per evitare sovrapposizioni).
Durante quest'ultima fase però mi sono accorto che il sito in questione, per alcuni articoli utilizza una struttura completamente differente del corpo html, per semplicità quindi, nella prima fase della parte A, i link recuperati dell'archivio sono superiori a 1000 per assicurare almeno 1000 file scaricati nonostante gli errori.

3 Parte B

La parte B è stata volta all'interno del file analyzer.py, appoggiandosi a item_reader.py per la lettura degli articoli (rirotna una lista contentente gli articoli già "puliti" dai caratteri che non ci interessano), e a gnuplot.py per la stampa dei grafici (in caso quest'ultima causare errori, in quanto utilizza gnuplot, basterà quindi commentarla).

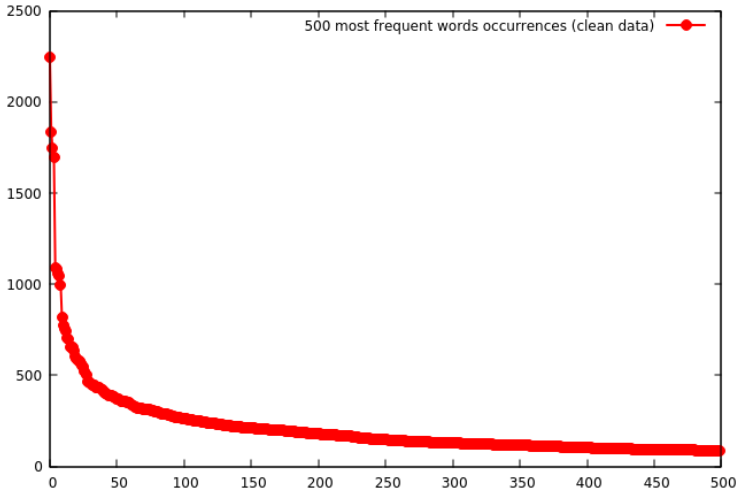
3.1 B.1 - analyzer.start_base

Questa parte consiste semplicemente nella creaione di un semplice dizionario delle occorrenze globali delle parole. Una volta creato questo dizionario, salvo in una lista le parole presenti nel dizionario, ordinate per occorrenze. In base a questa lista creo poi un file **.dat** utile per stampare poi il grafico tramite gnuplot. Dirante questa fase ho selezionato poi alcune parole molto frequenti e di poca rilevanza, e le ho inserite poi nel file **stopwords.txt**



3.2 B.2 - analyzer.start_advanced

Similmente a come avviene nella parte B.1, in questa parte si andranno a contare le occorrenze e poi a creare il file **.dat** per stampare il grafico. Questa volta però prima di contare le occorrenze, ci occuperemo di rimuovere le stopwords (leggendole dal file stopwords.txt), e tramite lemmatize trasformeremo le parole nei relativi lemmi (per esempio "he, she, it" vengono interpretati tutti come "he").



3.3 B.3 - analyzer.content_recommender

Questa parte utilizza in gran parte le librerie di **Gensim**, infatti per prima cosa creiamo un dizionario di tutte le parole presenti nei testi. Successivamente andremo a creare il corpus, ovvero una lista dove ogni elemento rappresenta un articolo, e questi elementi sono composti di una lista di tuple che rappresentano parola-occorrenze, tramite **tf-idf** andremo poi a dare un "peso" alle parole.

Dato che viene richiesto di dare i suggerimenti in base a N articoli, andremo ad inserire in corpus un nuovo articolo fittizio, che rappresenterà la "media" degli articoli scelti. Per fare ciò basta fare la media delle occorrenze nei vari articoli.

A questo punto non resta che creare la matrice di similarità (sempre tramite librerie di gensim) e in base a quella estrarre i suggerimenti.

```
Articoli di partenza

Outfit your entire family with 360-degree cameras - The Verge
AMD Radeon RX480 review roundup: VR graphics for the masses - The Verge
Canon finally gets serious about mirrorless with the new M5 - The Verge
Virtual reality documentary Nomads arrives on Gear VR - The Verge
Sprint will support next-gen texting on Android next year - The Verge
Google’s Daydream VR headset is coming November 10th - The Verge
The HTC Vive now lets iPhone users take phone calls in VR - The Verge
Google used virtual poker dogs to find a way to stop VR trolls - The Verge
Apple's Services division is the one bright spot in a down fiscal 2016 - The Verge
CBS joins YouTube's streaming TV service, set to launch in early 2017 - The Verge
Here's everything you can do with the new MacBook Touch Bar - The Verge
Google's Daydream View VR headset goes on sale next month for $79 - The Verge
Snapchat releases a new web tool for making custom geofilters - The Verge
Microsoft's canceled Band 3 wearable leaked in new images - The Verge
Apple has been ordered to pay VirnetX $302.4 million in patent lawsuit - The Verge
Tour Yosemite in virtual reality with Barack Obama - The Verge
Apple now sells refurbished iPhones - The Verge
Virtual reality theme park The Void opening its first outpost in Times Square - The Verge
Poll: How's that VR headset working out for you? - The Verge
The Martian VR Experience is coming to PSVR and Vive tomorrow - The Verge


Articoli suggeriti

1       similarita': 34.54 %    >  Google Daydream View review: mobile VR done mostly right - The Verge
2       similarita': 31.69 %    >  Google Daydream View is the coziest VR headset - The Verge
3       similarita': 29.40 %    >  Daydream is Google’s Android-powered VR platform - The Verge
4       similarita': 28.64 %    >  Google Daydream is a quixotic quest to make VR normal - The Verge
5       similarita': 28.57 %    >  Google reveals plans for new VR headset and motion controller - The Verge
6       similarita': 27.09 %    >  Daydream Nation - The Verge
7       similarita': 26.92 %    >  Google Pixel review: Home run - The Verge
8       similarita': 26.06 %    >  Here's where you can buy Google's new Daydream VR headset - The Verge
9       similarita': 25.72 %    >  Google launches Daydream with a new Harry Potter game - The Verge
10      similarita': 25.40 %    >  Google Daydream VR will reportedly launch in ‘weeks' - The Verge
11      similarita': 24.99 %    >  Google's VR platform includes partnerships with HBO, Ubisoft, the NBA, and more - The Verge
12      similarita': 24.32 %    >  YouTube VR is Daydream's killer app - The Verge
13      similarita': 23.89 %    >  Here's why that 'leaked' Gear VR controller makes no sense - The Verge
14      similarita': 23.73 %    >  Canon’s 5D Mark IV has built-in Wi-Fi and shoots 4K video - The Verge
15      similarita': 23.72 %    >  The 5 biggest announcements from Google's Pixel event - The Verge
16      similarita': 22.32 %    >  Smartphone camera shootout: Google Pixel takes on the iPhone 7 and Galaxy S7 Edge - The Verge
17      similarita': 22.31 %    >  PlayStation VR review: When good enough is great - The Verge
18      similarita': 22.12 %    >  If you preorder a Pixel phone you'll also get a free Daydream View headset 'while supplies last'
19      similarita': 22.09 %    >  Mossberg: Google's first phone is first rate - The Verge
20      similarita': 21.51 %    >  Will virtual reality kill the YouTube comment? - The Verge
```

I risultati possono essere considerati validi in quanto gli articoli suggeriti risultano essere simili a quelli più frequenti negli articoli di partenza, che come possiamo vedere, se pur con qualche eccezzione, risultano avere degli argomenti abbastanza omogenei.

4 Parte C - analyzer.topic_finder

La parte C è molto simile alla parte b (infatti si basa sui dati creati in precedenza) con la differenza che modifica il corpus in base alla riduzione dimensionale (numero di topic) che scegliamo. Una volta fatto questo, topic_finder utilizza le stesse funzioni di supporto utilizzare da B.3 per trovare e stampare i suggerimenti. Come prevedibile, per un numero molto basso di topic, e quindi per un numero molto basso di dimensioni, gli articoli tendono ad avere una somiglianza molto alta (in alcuni casi pari al 100%), mentre per un numero molto alto (tendente al numero di articoli presenti) si tenderà ad avere un risultato simile a quello di partenza, in quanto la riduzione dimensionale sarà quasi inesistente.

Figure 1: con 1 solo topic gli articoli risultano essere tutti identici tra di loro

Figure 2: con 950 topic, gli articoli presentano una similarità molto simile a quella di partenza