Relazione
Esercitazione 5

TM - TV

Introduzione

La seguente esercitazione ha come consegna quella di sviluppare un sistema che, sfruttando la tecnica del Topic Modelling, parte da un testo e restituisce una lista di topics.

Nel caso di studio è stato considerato un approccio LDA (Latent Diritchlet Allocation), reso più semplice grazie all'uso della libreria `Gensim` che, appunto, ne fornisce una implementazione già pronta.

Il Topic Modelling affronta il problema di recuperare, a partire da una base documentale, il topic, ovvero una lista di parole riassuntive di ciò che si sta parlando. LDA considera ogni documento come una collezione di topic sotto una certa proporzione, e ogni topic è un insieme di parole chiave che, sotto una distribuzione di probabilità rimandano ad un tema.

Il corpus contenente sei documenti, è stato recuperato dalla libreria `NLTK`, nello specifico dal modulo `webtext` il quale è una raccolta di testi dal web che è stata inclusa come parte del pacchetto dati `NLTK`. Alcuni esempi di testi presenti nella raccolta di testi web includono sceneggiature di film, registri di chat e articoli di notizie.

Nello specifico i testi analizzati in questa esercitazione sono

- *firefox.txt*
- *grail.txt*
- *overheard.txt*
- *pirates.txt*
- *singles.txt*
- *wine.txt*

Struttura del codice ed implementazione

Il codice è presente nel file

- **`main.py`**
  si occupa di richiamare i metodi principali per l'esecuzione del task, ovvero

    - `retrive_coprus()`
      metodo per recuperare i documenti dal modulo `webtext` descritto in precedenza, ed unirli in un'unica lista `corpus_merge`

    - `clean_sentences(coprus)`
      metodo che si occupa di processare le frasi del corpus attraverso una pipeline, definendo così un *bag of words* delle frasi dei documenti

    - `extract_topics(bag_of_words)`
      metodo che si occupa di estrarre e visualizzare i topics grazie alla libreria **`gensim`**

Nello specifico, il task è stato realizzato eseguendo i seguenti step

- pre-processing dei dati

    - Include alcune funzioni per processare i dati rimuovendo i simboli di punteggiatura, stopwords, spazi bianchi multipli e infine per costruire il *bag of words* di ogni frase del corpus.

- creazione del dizionario e del corpus necessari per il topic modelling

    - Per la generazione dei topics, la libreria `gensim` richiede un dizionario e un corpus
    - Infatti, per analizzare il testo, la libreria `gensim` richiede che i tokens pre-processati siano convertiti associando loro degli identificativi univoci; quindi, per far ciò si avvale dei dizionari di python. Ogni token è mappato con un id unico. Viene eseguito tramite questa linea di codice

      ```
      dict_LoS = corpora.Dictionary(text)
      ```

    - Questo dizionario verrà usato come base di partenza per la costruzione del corpus che sostanzialmente è un mapping tra id univoci del dizionario e la frequenza della parola all'interno del documento originale. Questo ci permette di avere una rappresentazione sparsa delle occorrenze di ciascuna parola nel paragrafo.

      ```
      BoW_corpus = [dict_LoS.doc2bow(text) for text in text]
      ```

- costruzione dei topics

- Per quanto riguarda la costruzione dei topics, la libreria `gensim` offre la funzione `gensim.models.ldamodel.LdaModel()` che riceve in ingresso il corpus, il dizionario e il numero di topic che deve trovare.

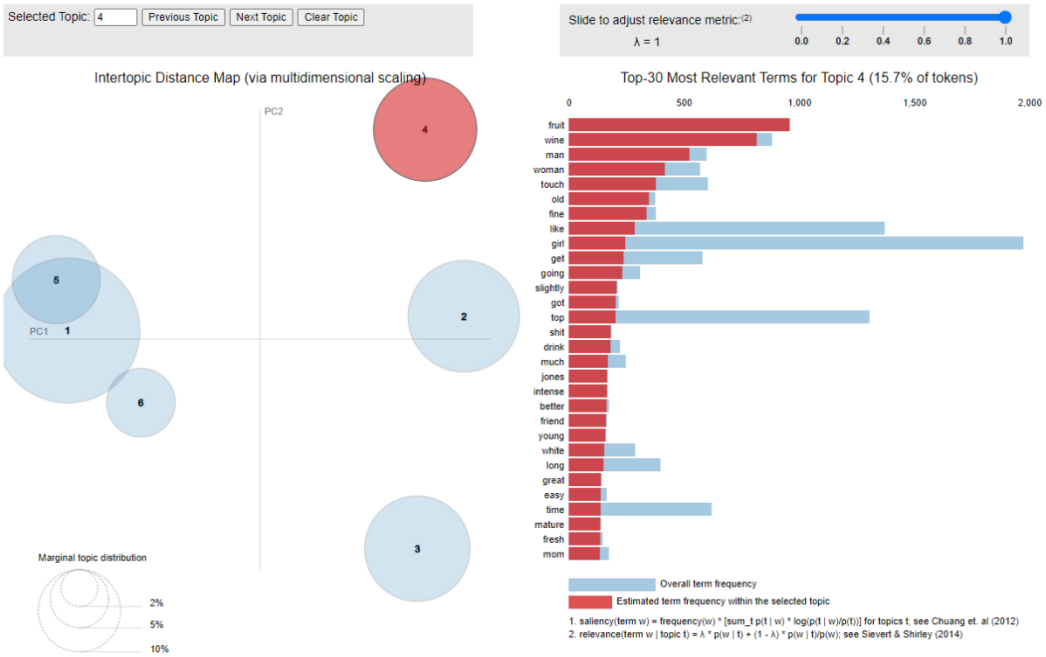
```
lda_model = LdaModel(corpus=BoW_corpus,id2word=dict_LoS,num_topics=N_TOPICS)
```

- Un modo semplice per visualizzare i topics costruiti è usare la libreria `pyLDAvis`. Quest'ultima offre una serie di primitive che consentono di rappresentare graficamente il risultato della computazione dell'algoritmo LDA. In particolare è stata invocata la funzione `pyLDAvis.gensim_models.prepare` e i risultati sono stati salvati in un file `html`.

```
vis = pyLDAvis.gensim_models.prepare(lda_model, BoW_corpus, dict_LoS)
pyLDAvis.save_html(vis, 'LDA_Visualization.html')
```

Risultati

Di seguito viene riportata una schermata del grafico costruito attraverso il modello `pyLDAvis`, relativo al quarto topic (*wine.txt*)



Di seguito una rappresentazione dei topics, salvata in una lista che contiene le parole e le probabilità di ciascun termine per ogni contesto trovato nel modello LDA

```
[(0,
  '0.038*"quite" + 0.024*"dry" + 0.023*"nice" + 0.021*"really" + 0.020*"like" '
  '+ 0.017*"rather" + 0.016*"bottle" + 0.015*"girl" + 0.011*"well" + '
  '0.009*"long"'),
 (1,
  '0.060*"wine" + 0.030*"bare" + 0.028*"lady" + 0.018*"know" + '
  '0.014*"slightly" + 0.013*"might" + 0.008*"tourist" + 0.008*"like" + '
  '0.007*"glass" + 0.006*"think"'),
 (2,
  '0.077*"guy" + 0.056*"jack" + 0.023*"year" + 0.018*"drinking" + '
  '0.013*"nicely" + 0.011*"ship" + 0.011*"finish" + 0.011*"sparrow" + '
  '0.009*"vintage" + 0.008*"age"'),
 (3,
  '0.035*"good" + 0.030*"fruit" + 0.021*"bit" + 0.017*"woman" + 0.015*"man" + '
  '0.015*"lovely" + 0.014*"wa" + 0.013*"nose" + 0.012*"like" + 0.011*"yeah"'),
 (4,
  '0.061*"girl" + 0.015*"boy" + 0.015*"little" + 0.014*"teen" + '
  '0.014*"perhaps" + 0.011*"ha" + 0.009*"pretty" + 0.009*"complex" + '
  '0.009*"shit" + 0.008*"oh"'),
 (5,
  '0.034*"top" + 0.027*"touch" + 0.011*"elizabeth" + 0.010*"weight" + '
  '0.009*"palate" + 0.008*"open" + 0.008*"forward" + 0.007*"swann" + '
  '0.007*"style" + 0.007*"classic"')]
```

Conclusioni

I topics sono stati individuati con un certo grado di precisione.

Una considerazione da fare è relativa a due documenti *singles.txt overheard.txt* infatti dalla figura si nota che è presente una forte sovrapposizione dei cluster, questo è dovuto principalmente al fatto che il primo documento parla di annunci di persone single, il secondo riguardi conversazioni (chat) tra uomini e donne, per cui è lecito che i due cluster risultati siano per buona parte sovrapposti.