Università degli studi di Milano-Bicocca

Text Mining and Search

Final Project

The 20 Newsgroups

Text Classification task

Authors:
Beatrice Fumagalli - 784549 - b.fumagalli9@campus.unimib.it
Matteo Porcino - 748876 - m.porcino1@campus.unimib.it

1. Introduzione

Il dataset “20 newsgroups” è una raccolta di circa 20.000 documenti, suddivisi in modo -quasi- uniforme in 20 diversi *newsgroup*. Per quanto ne sappiamo, è stato originariamente raccolto da *Ken Lang*, probabilmente per il suo documento ‘*Newsweeder: Learning to filter netnews’*, anche se non menziona esplicitamente questa collezione. La raccolta 20 newsgroups è diventata un set di dati diffuso per esperimenti in *Text applications* di *Machine Learning*, come la *Text Classification* e il *Text Clustering*.

2. Dataset

Il dataset iniziale “20_newsgroups” è una raccolta di 19997 documenti, suddivisi in 20 diversi *newsgroup* di 1000 documenti ciascuno, fatta eccezione per il soc.religion.christian *newsgroup* che ne presenta 997:

- alt.atheism
- comp.graphics
- comp.os.ms-windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x
- misc.forsale
- rec.autos
- rec.motorcycles
- rec.sport.baseball
- rec.sport.hockey
- sci.crypt
- sci.electronics
- sci.med
- sci.space
- soc.religion.christian
- talk.politics.guns
- talk.politics.mideast
- talk.politics.misc
- talk.religion.misc

Ogni *record* nel set di dati è in realtà un file di testo, scritto in inglese, che presenta la seguente struttura: metadati – intestazione – testo del documento.

Il *dataset* è stato splittato in *Training set* (70%) e *Test set* (30%) per l’applicazione dei modelli di *Text Classification*.

3. Text pre-processing

La fase di *pre-processing* del testo di ciascun documento dei 20 *newsgroups* è stata articolata in un’unica funzione chiamata *text_preprocessing* e successivamente applicata

sia al *Training set* che al *Test set*. Al suo interno sono state applicate a loro volta delle funzioni, precedentemente ed appositamente create, per svolgere i seguenti compiti:

- Eliminazione dei metadata
- Eliminazione delle contrazioni
- Rimozione di URLs e indirizzi *e-mail*
- *Tokenization*
- Rimozione di righe vuote e *tabs*
- Rimozione della punteggiatura ed eventuali ulteriori spazi vuoti
- Eliminazione delle stringhe numeriche
- Normalizzazione dei caratteri in minuscolo
- Rimozione delle *stop-words*
- Eliminazione dei caratteri non *unicode*
- Eliminazione della punteggiatura
- *Stemming* e *Lemmatization*
- Eliminazione dei *tokens* di lunghezza inferiore a 2

Tutta la fase di *pre-processing* è stata implementata servendosi principalmente della libreria **NLTK** messa a disposizione da *Python*.

4. Text Representation

La *Text Representation* è stata effettuata misurando i pesi assegnati a ciascun termine attraverso due euristiche:

- Term Frequencies (TF): la Term Frequency $tf_{t,d}$ del termine t nel documento d è definito come il numero di volte che t si verifica in d.
- Term Frequency times Inverse Document Frequency (TF-IDF): il peso tf-idf di un termine è il prodotto del suo peso tf e del suo peso idf.

Per entrambe le misurazioni è stata utilizzata la libreria scikit-learn messa a disposizione da Python, settando i parametri tokenizer e preprocessor a NULL, in quanto fasi già svolte precedentemente. Attraverso il metodo fit_transform viene prima adattato lo stimatore dei pesi ai dati e successivamente la count-matrix viene trasformata in una rappresentazione TF o TF-IDF.
Entrambe le euristiche restituiscono una matrice *Document-Term* di dimensioni 13997x119613, dove la prima rappresenta il numero di documenti presenti all'interno del training set mentre la seconda rappresenta il numero di termini all'interno del vocabolario generato dal corpus dei 20 *newsgroups*.

5. Text Classification

Per la *Text Classification* sono stati testati quattro modelli su entrambe le rappresentazioni precedentemente descritte, al fine di stabilire quale combinazione rappresentazione-modello fornisca la miglior classificazione.

Per l'implementazione di ciascuno dei modelli di classificazione è stata ri-utilizzata la libreria scikit-learn con le funzioni concernenti il modello preso in esame. Per quanto riguarda invece la valutazione dei risultati ottenuti, si è presa in considerazione l'a*ccuracy*, ossia la percentuale di classificazioni corrette.

5.1 Multinomial Naive Bayes

Il classificatore *Multinomial Naive Bayes* è un classificatore *bayesiano* con un modello di probabilità sottostante che si basa sull'ipotesi di indipendenza delle *feature*, ovvero assume che la presenza o l'assenza di una particolare *feature* in un documento testuale non sia correlata alla presenza o assenza di altre *feature*.
Applicando il modello sulle due rappresentazioni si ottengono i seguenti risultati:

- TF: accuracy = 0.812
- TF-IDF: accuracy = 0.848

Pertanto, il modello *Naive Bayes* fitta meglio sulla rappresentazione TF-IDF.

5.2 Logistic Regression

La regressione logistica misura la relazione tra la variabile dipendente categoriale e una o più variabili indipendenti stimando le probabilità usando una funzione logistico/sigmoide.
Applicando il classificatore *Logistic Regression* ad entrambe le rappresentazioni si ottengono i seguenti valori di accuratezza:

- TF: accuracy = 0.809
- TF-IDF: accuracy = 0.823

Pertanto, il modello fitta meglio sulla rappresentazione TF-IDF, come il modello *Naive Bayes*.

5.3 Random Forest

Il *Random Forest* è un algoritmo di apprendimento supervisionato. Un classificatore *Random Forest* calcola la media di più alberi decisionali in base a campioni casuali del database. Fornisce anche un buon indicatore dell'importanza della *feature*.
Applicando tale modello alle due rappresentazioni si ottengono i risultati seguenti:

- TF: accuracy = 0.619
- TF-IDF: accuracy = 0.626

Anche il modello *Random Forest* fitta meglio sulla rappresentazione TF-IDF.

5.4 Support Vector Machine

Il *Support Vector Machine* è un classificatore binario, ossia divide gli oggetti in due classi. In *Classification*, ciò significa che determina se l'oggetto appartiene o meno alla classe. Per fare ciò, i valori sono mappati in un iperpiano. Una funzione lineare è usata per determinare un confine che divide gli oggetti in due classi. Il limite si basa sulla distanza degli oggetti più vicini in entrambe le classi. Questa distanza deve essere massimizzata. Gli oggetti più vicini sono chiamati “vettori di supporto”, i cosiddetti *Support Vectors*.
Applicando quest’ultimo modello ai due differenti metodi di *Text* Representation si ottengono i seguenti valori:

- TF: accuracy = 0.754
- TF-IDF: accuracy = 0.840

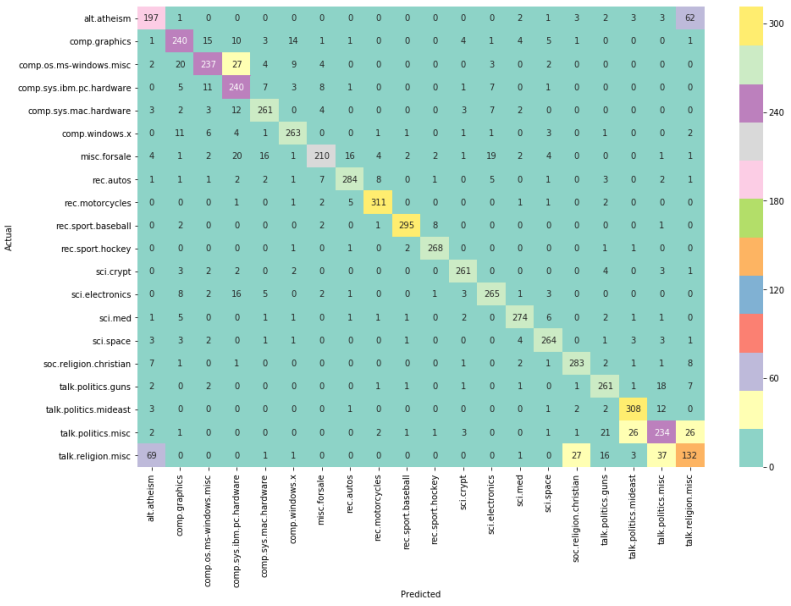
Il modello SVM fitta meglio sulla rappresentazione TF-IDF.

6. Conclusioni

Con una leggera differenza di accuratezza ma stessa rappresentazione (TF-IDF) , i modelli di classificazione migliori risultano essere il modello *Naive Bayes* e il modello *Support Vector Machine*. Pertanto si sono analizzate tre ulteriori misure: *recall, precision e f1-score.* La recall è calcolata come il numero di volte in cui il classificatore assegna un'etichetta di una categoria specifica tra tutte le etichette esistenti di quella categoria. La *precision* viene calcolata come il numero di volte in cui un'etichetta di una categoria specifica viene assegnata correttamente. L’*f1-score* è un compromesso tra queste due ultime misure.

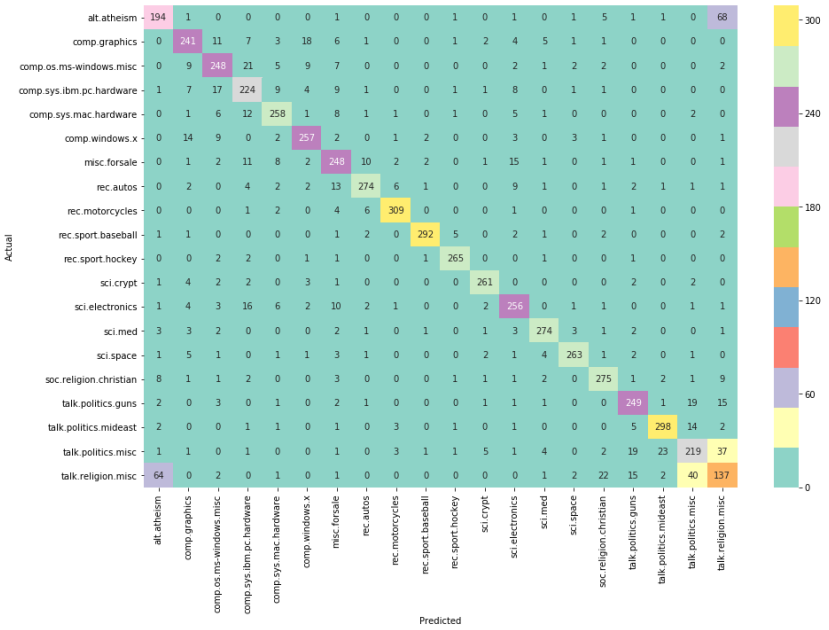
- **Naive Bayes, TF-IDF:**

	precision	recall	f1-score	support
alt.atheism	0.67	0.72	0.69	274
comp.graphics	0.79	0.80	0.79	301
comp.os.ms-windows.misc	0.84	0.77	0.80	308
comp.sys.ibm.pc.hardware	0.72	0.85	0.78	284
comp.sys.mac.hardware	0.86	0.88	0.87	297
comp.windows.x	0.88	0.89	0.89	295
misc.forsale	0.88	0.69	0.77	306
rec.autos	0.91	0.89	0.90	320
rec.motorcycles	0.95	0.96	0.95	324
rec.sport.baseball	0.97	0.95	0.96	309
rec.sport.hockey	0.95	0.98	0.97	274
sci.crypt	0.93	0.94	0.93	278
sci.electronics	0.86	0.86	0.86	307
sci.med	0.93	0.92	0.93	297
sci.space	0.90	0.92	0.91	287
soc.religion.christian	0.89	0.92	0.90	308
talk.politics.guns	0.82	0.88	0.85	296
talk.politics.mideast	0.89	0.94	0.91	329
talk.politics.misc	0.74	0.73	0.74	319
talk.religion.misc	0.55	0.46	0.50	287
micro avg	0.85	0.85	0.85	6000
macro avg	0.85	0.85	0.85	6000
weighted avg	0.85	0.85	0.85	6000



- **Support Vector Machine, TF-IDF:**

	precision	recall	f1-score	support
alt.atheism	0.70	0.71	0.70	274
comp.graphics	0.82	0.80	0.81	301
comp.os.ms-windows.misc	0.80	0.81	0.80	308
comp.sys.ibm.pc.hardware	0.74	0.79	0.76	284
comp.sys.mac.hardware	0.86	0.87	0.87	297
comp.windows.x	0.86	0.87	0.86	295
misc.forsale	0.77	0.81	0.79	306
rec.autos	0.91	0.86	0.88	320
rec.motorcycles	0.95	0.95	0.95	324
rec.sport.baseball	0.97	0.94	0.96	309
rec.sport.hockey	0.96	0.97	0.96	274
sci.crypt	0.94	0.94	0.94	278
sci.electronics	0.82	0.83	0.82	307
sci.med	0.92	0.92	0.92	297
sci.space	0.95	0.92	0.93	287
soc.religion.christian	0.87	0.89	0.88	308
talk.politics.guns	0.83	0.84	0.83	296
talk.politics.mideast	0.91	0.91	0.91	329
talk.politics.misc	0.73	0.69	0.71	319
talk.religion.misc	0.49	0.48	0.49	287
micro avg	0.84	0.84	0.84	6000
macro avg	0.84	0.84	0.84	6000
weighted avg	0.84	0.84	0.84	6000



Analizzando gli output ottenuti, il modello migliore risulta essere il *Naive Bayes*, in termini di accuratezza, *recall* ed *f1-score*.