

# Analisi del sentimento

Marco Frassinetti - 6301351

## Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Dataset . . . . .	1
1.2	Codice . . . . .	1
<b>2</b>	<b>Risultati</b>	<b>2</b>
<b>3</b>	<b>Uso del codice</b>	<b>2</b>
<b>4</b>	<b>Riferimenti</b>	<b>3</b>

## 1 Introduzione

In questo esperimento si è cercato di riprodurre i risultati ottenuti da [Gräßer et al. 2018](#) per l'analisi del sentimento su recensioni di farmaci. Sono state utilizzate le implementazioni di Naive Bayes Bernoulli e Multinomiale fornite da `scikit-learn` in python per la classificazione delle recensioni secondo il metodo descritto nell'articolo. I modelli sono valutati misurando *accuratezza* e *Kappa di Cohen*.

### 1.1 Dataset

Si è utilizzato il dataset reperibile su [UCI](#) per allenare e testare i classificatori. Il dataset comprende 215063 campioni composti da una recensione testuale, una valutazione da 1 a 10 data dall'utente e altri attributi che verranno ignorati in questo esperimento. In base alla valutazione dell'utente si andranno ad identificare tre classi usate per la classificazione con Naive Bayes. Il dataset é già suddiviso in partizioni di *train* (75%) e *test* (25%).

### 1.2 Codice

Il codice, scritto in python, fa uso delle librerie `panda` e `scikit-learn` ed esegue le seguenti azioni:

1. Apre i file del dataset (*train* e *test*) e ne estrae: le recensioni, dalle quali sono rimossi tutti i caratteri non alfabetici i quali vengono trasformati in

minuscolo, e il voto da 1 a 10, che viene trasformato nella label della classe, "1" se il voto è  $\geq 7$ , "-1" se è  $\leq 4$ , altrimenti "0" per quelle comprese tra 4 e 7. (vedi funzione `load_dataset`)

2. Estrae le *features* dalle recensioni utilizzando la classe `CountVectorized` di `scikit-learn`, come descritto nell'articolo sono state utilizzate parole, bigram, e trigram (usando il parametro `ngram_range`) e sono state ignorate le *features* che appaiono con frequenza maggiore dell'80% (tramite il parametro `max_df`).  
Così facendo si ottiene la *document-term matrix* che indica le frequenze con cui appaiono le varie *features* nelle recensioni.
3. Esegue una 5-fold cross-validation per determinare il miglior valore per l'iperparametro  $\alpha$  di Naive Bayes (parametro di *additive smoothing* per casi estremi in cui determinate parole non si presentano nel train set, se  $\alpha = 1$  si ha *Laplace Smoothing* altrimenti si ha *Lidstone smoothing*).
4. Crea il classificatore, `BernoulliNB` o `MultinomialNB`, passando l'iperparametro ottenuto precedentemente e lo allena passando la *document-term matrix* e le *label* del dataset di allenamento, successivamente prova a predire il dataset di test e ne calcola *accuratezza* e *Kappa di Cohen*.

## 2 Risultati

Di seguito sono riportati i risultati ottenuti in questo esperimento e i dati ottenuti da *Gräßer et al. 2018* che usa *Logistic Regression* per la classificazione. Per l'iperparametro i valori migliori trovati con la 5-fold cross-validation sono:

- $\alpha = 0.4$  per Bernoulli Naive Bayes
- $\alpha = 0.6$  per Multinomial Naive Bayes

Per quanto riguarda accuratezza e Kappa di Cohen si può osservare come il modello multinomiale dia risultati migliori rispetto al modello di Bernoulli, ma entrambi risultino inferiori se confrontati con i risultati ottenuti nell'articolo citato.

	Accuratezza	Kappa di Cohen
BernoulliNB	87.46	71.87
MultinomialNB	89.90	77.98
Gräßer et al. 2018	92.24	83.99

## 3 Uso del codice

Per riprodurre i risultati ottenuti basta eseguire lo script `main.py` con python3, assicurandosi che nella stessa directory siano presenti i due file del dataset (`drugsComTrain_raw.tsv` e `drugsComTest_raw.tsv`), in alternativa è possibile specificare i file modificando le righe 75 e 78 di `main.py`. I risultati verranno mostrati come output su console.

## 4 Riferimenti

1. F. Gräßer, S. Kallumadi, H. Malberg, and S. Zaunseder. (2018). Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning. <https://dl.acm.org/doi/10.1145/3194658.3194677>
2. scikit-learn: [https://scikit-learn.org/stable/modules/naive\\_bayes.html](https://scikit-learn.org/stable/modules/naive_bayes.html)
3. panda: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/index.html](https://pandas.pydata.org/pandas-docs/stable/user_guide/index.html)