Università degli studi di
Milano Bicocca

Text Mining & Search
Amazon Reviews Analysis

Authors:
Davide Sangalli - 848013 - d.sangalli5@campus.unimib.it
Federico Signoretta - 847343 - f.signoretta@campus.unimib.it
Diana Tenca - 789651- d.tenca@campus.unimib.it

Sommario

Amazon è un colosso mondiale nell'ambito dell'e-commerce. Una delle ragioni del suo successo è la notorietà del nome e l'affidabilità che esso suscita nei potenziali clienti. Tale *awareness* è stata conquistata anche grazie al sistema di review che permette ai clienti di confrontarsi e di suggerire un prodotto basandosi sulla propria esperienza.
In questo progetto, partendo dalle recensioni, si è deciso di sviluppare tre task differenti: il primo incentrato sull'*opinion mining* ossia, atto a determinare l'opinione espressa dal cliente mentre, nel secondo si è provato ad implementare un modello di classificazione che determinasse la categoria di appartenenza del prodotto recensito. Per entrambi i task sono state condotte le trasformazioni di pre-processing ed è stata utilizzata la tecnica di *word embedding* per la *text representation*. Quest'ultima scelta è stata guidata, in particolare nella classificazione della categoria di appartenenza del prodotto, dall'importanza rivestita dal contesto semantico. Le categorie risultano abbastanza simili l'una all'altra e per questo molti termini ricorrono frequentemente nelle recensioni indipendentemente dal prodotto a cui esse fanno riferimento. Ciononostante, considerando il contesto e non solo i singoli termini, è stato possibile ottenere dei buoni risultati dai modelli utilizzati. Infine, come task aggiuntivo, è stato proposto un modello di clustering non supervisionato al fine di ottenere una suddivisione rispetto alla polarità delle recensioni.

1 Introduzione

Con un fatturato annuo di più di 232 milioni di dollari, *Amazon.com* è uno dei maggiori siti e-commerce a livello mondiale. Uno dei punti di forza del colosso americano è il sistema di review: ogni cliente può riportare la propria esperienza di acquisto ed esprimere la propria opinione riguardo al prodotto comprato. Ciò permette la creazione di una community di persone che si consigliano ed influenzano a vicenda.
I dati sono stati scaricati dal seguente link `https://nijianmo.github.io/amazon/index.html`: sono stati utilizzati i dati campionati presenti nella sezione *"Small" subsets for experimentation*.
Le feature presenti in ogni dataset importato sono:

- asin: ID del prodotto
- overall: rating assegnato.
- reviewText: testo della recensione (in lingua inglese)
- reviewTime: quando la recensione è stata scritta (formato raw)
- reviewerID: ID recensione
- reviewerName: nome del cliente
- summary: riassunto della recensione
- unixReviewTime: quando la recensione è stata scritta (formato unix)
- vote: quanto una recensione viene ritenuta utile dagli altri utenti
- category: categoria del prodotto recensito

Categories	Number of Reviews
Industrial and Scientific	800357
Patio Lawn and Garden	798415
Video Games	497577
Prime Pantry	494485
Digital Music	231392
Musical Instruments	169781
Arts Crafts and Sewing	137788
Office Products	77071

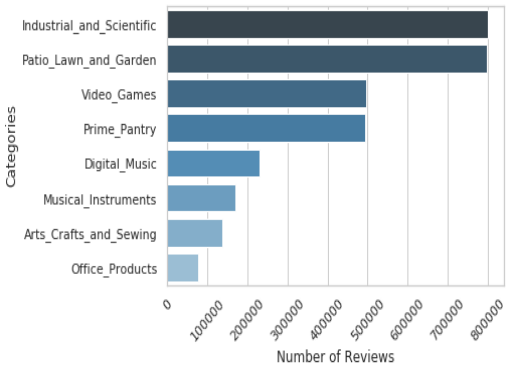
Tabella 1: Categorie selezionate per l'analisi

Le istanze totali - date dall'unione di tutte le categorie - sono 3 206 866.
Visto il grande numero di reviews, si è deciso di utilizzare un campione di sole 8 categorie in modo da evitare tempi computazionali eccessivi. Le categorie selezionate sono riportate nella Tabella 1.

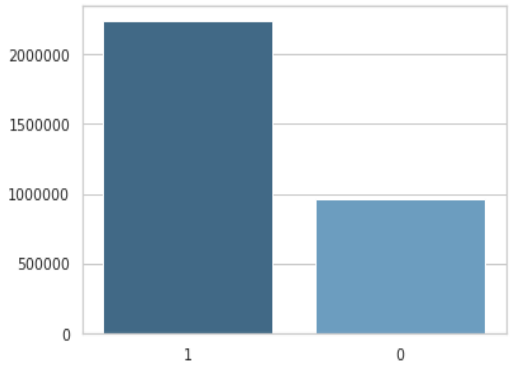
Uno studio condotto presso la *The Ohio State University* ha dimostrato che la reattività umana risulta maggiore se soggetta a stimoli negativi. Per tale motivo, si potrebbe supporre che le recensioni negative hanno un maggior impatto sui clienti rispetto a quelle positive. Risulta quindi evidente l'importanza di cogliere immediatamente il malcontento al fine di evitare che i consumatori insoddisfatti decidano di compiere i loro futuri acquisti su altre piattaforme.
Pertanto, si è deciso di effettuare una classificazione basata sul *opinion* delle review, attraverso l'utilizzo di una *Recurrent Neural Network*. Peraltro, vista l'importanza della corretta identificazione dell'opinione espressa in una recensione, si è deciso di utilizzare - oltre ai modelli di classificazione - un algoritmo di clustering non supervisionato. Queste ultime due metodologie, basate su diversi criteri per la determinazione della positività/negatività della recensione, non vogliono essere confrontate, bensì fornire due differenti strumenti validi all'identificazione dei giudizi degli utenti.
Oltre a questi primi task, si è deciso di condurre una seconda analisi pensata per un cliente diverso da Amazon: supponendo che il lessico usato per le recensioni in Amazon sia analogo a quello dei clienti di altri e-commerce fornitori di prodotti simili, è stata implementata una architettura in grado di classificare opportunamente le recensioni in diverse categorie, basandosi su quelle di Amazon (attraverso l'utilizzo di una *Recurrent Neural Network*). Per questa analisi sono state estratte solo le caratteristiche effettivamente utili ai fini dell'analisi: **overall**,**category** e **reviewText**.

La prima cosa effettuata, è stata osservare i dati: si è da subito notato un forte sbilanciamento fra le classi, (Figura 1), sia per il task incentrato sulla definizione della categoria che quello inerente all'opinion mining (nella figura sono rappresentate le review positive/negative in base al criterio utilizzato con il modello di classificazione). Al fine di risolvere questo problema, è stato inizialmente realizzato un *undersampling* ma, poiché i risultati dei modelli non erano soddisfacenti, si è deciso di bilanciare le classi nella fase di addestramento dei modelli (i pesi sono stati assegnati automaticamente attraverso la funzione *compute_weight()* servita dal pacchetto *sklearn*).

Terminata la fase di osservazione del dataset si è proceduto con le fasi di preprocessing.



(a) Reviews per Categoria di Prodotto



(b) Reviews Positive e Negative (definizione per il task di classificazione)

Figura 1: Sbilanciamento Classi

2 Preprocessing

Al fine di effettuare delle buone analisi sui dati testuali le fasi di preprocessing rivestono un ruolo fondamentale. Tuttavia, la loro corretta implementazione non è semplice. In questo progetto sono state condotte le operazioni di normalizzazione, lemmatization e tokenization.

2.0.1 Normalizzazione

Con il termine normalizzazione ci si riferisce al processo di adeguamento del testo al fine di evitare che la presenza di simboli, accenti o maiuscole possa compromettere il riconoscimento di una parola. A tal scopo, sono state utilizzate le espressioni regolari così da eliminare eventuali simboli, numeri, sostituire le vocali accentate con le rispettive lettere non accentate e, con la funzione *.lower*, sono state rimosse anche quelle maiuscole.
In seguito, sono stati eliminati i duplicati e i valori nulli così da considerare le sole recensioni in grado di fornire nuove informazioni al modello. La fase di normalizzazione si é conclusa con la sostituzione delle forme contratte con le rispettive scritture estese (es. la forma *don't* è stata trasformata in *do not*).

Da questo punto il preprocessing dei due task viene concluso in due modi differenti:

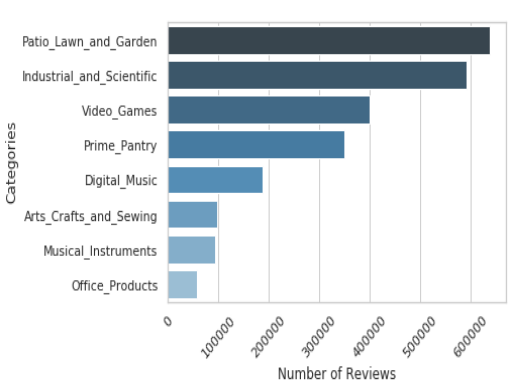
- **Task 1** (Classificazione delle categorie): nell'identificazione del prodotto recensito le stop word possono anzi, secondo la teoria di Luhn, devono essere rimosse. É stata utilizzata la libreria *nltk* ed in particolare il dizionario delle stop word inglesi in essa contenuto.
- **Task 2** (classificazione dell'opinion mining): in questo secondo caso, non volendo utilizzare un dizionario per la determinazione dell'opinione associata alla recensione, si è dovuta trovare una soluzione che tenesse conto della presenza di negazioni. Per questo motivo, è stata utilizzata

un'espressione regolare che legasse la negazione alla parola consecutiva così da tenerne traccia. Si noti che in realtà questa fase è stata condotta a seguito della lemmatizzazione di cui parleremo successivamente.

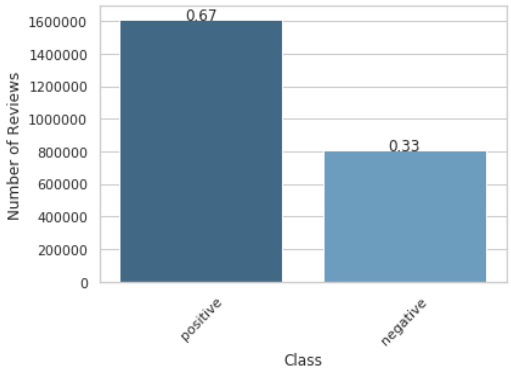
- **Task 2** (clusterizzazione dell'opinion mining): in questo caso le stop word sono state mantenute poichè la libreria *analyzer* era in grado di gestirle autonomamente.

2.0.2 Lemmatization e Tokenization

Nella fase conclusiva del preprocessing sono state utilizzate le tecniche di *lemmatization* e *tokenization* (attraverso l'utilizzo della libreria **nltk**). La scelta di effettuare la lemmatization - invece di applicare un algoritmo di stemming - deriva dalla grande somiglianza dei termini utilizzati nelle recensioni che, riportando le parole alla sola radice, sarebbe stata ulteriormente accentuata rischiando di impedire al modello di cogliere al meglio le relazioni tra i termini e quindi di peggiorare le performance.
A seguito di numerose prove si è deciso di assegnare dei tag ai verbi, agli aggettivi e agli avverbi che spesso, a causa dell'ambiguità, non venivano correttamente modificati dalla funzione di lemmatizzazione.
Terminate le operazioni di preparazione dei dati, sono stati ricreati i grafici presentati in figura 1 al fine di cogliere eventuali modifiche. Si noti che lo sbilanciamento delle classi si è leggermente ridotto a seguito dell'eliminazione di quasi un milione di recensioni. Ciò nonostante, le classi risultano ancora fortemente sbilanciate.



(a) Reviews per Categoria di Prodotto



(b) Reviews Positive e Negative (definizione per il task di classificazione)

Figura 2: Sbilanciamento Classi a seguito del Preprocessing

3 Text Representation

Questa parte conclusiva di preparazione dei dati risulta essere cruciale ai fini delle performance dei modelli. Dopo aver valutato le varie possibilità, la scelta migliore è sembrata il **word embedding**. Utilizzando questa metodologia rappresentativa è possibile sfruttare non solo la frequenza delle parole all'interno del testo ma anche il contesto in cui esse compaiono. Questo risulta cruciale

sia per l'identificazione del prodotto in quanto molte parole vengono specializzate da quelle che le circondano e, a maggior ragione, nell'opinion mining in cui la co-occorrenza di due termini può modificare radicalmente il senso della frase.
Per prima cosa, si è suddiviso il dataset in training (75%)e test(15%). In seguito, è stato effettuato il word embedding: tale fase è stata specificata come layer della rete neurale, attraverso la funzione *Embedding*, servita da *Keras*.
A seguito di varie ricerche condotte, si è visto che l'utilizzo di un numero di parole superiore alle 100,000 più frequenti non modifica i risultati dei modelli di previsione, per questo motivo sono state considerate solo le prime 100,000 parole.

4 Task 1

4.1 Classificazione per categoria

Come detto in precedenza, si è deciso di implementare un modello di classificazione che permettesse di stabile la classe di appartenenza dell'oggetto recensito.
A seguito di diverse ricerche, si è presa la decisione di utilizzare una rete neurale per la classificazione della categoria. Più nel dettaglio, è stata implementata una *Recurrent Neural Network di tipo LSTM* utilizzando la libreria *Keras*. Di seguito sono riportate le specifiche della rete:

1. Layer di embedding: dimensione del vocabolario pari 100,000 e lunghezza del vettore pari a 100

2. Layer LSTM: costituito da 128 neuroni, con funzione di attivazione **relu** e come *'recurrent activation function'* **tanh**. Inoltre sono stati inseriti i parametri *'dropout'* e *'recurrent_dropout'* pari a 0.5, in modo da limitare l'overfitting.

3. Layer output: layer dense costituito da 8 neuroni, uno per categoria, ed anch'esso con funzione di attivazione **softmax**.

Per il *fitting* della rete si è scelto di utilizzare l'ottimizzatore **adam**, la funzione di perdita **categorical crossentropy**.
Come precedentemente accennato, il problema dello sbilanciamento delle classi è stato gestito utilizzando la funzione *compute_sample_weight* di *sklearn*. Una volta definite tutte le specifiche si è proceduto ad allenare la rete utilizzando 6 epoche ed un batch size di 2,048. I risultati ottenuti sono riportati in figura 4. Per valutare le performance si è deciso di tenere in considerazione la misura *f1-score*, siccome ci troviamo difronte ad un problema di classi sbilanciate. É possibile notare che in generale il modello ottiene dei buoni risultati, ottenendo l' 80% di f1-score (osservando la "macro avg"). Entrando più nel dettaglio, invece, è possibile notare che la categoria "Office Products" ottiene un f1-score pari al 50%: il motivo è chiaramente imputabile alle poche osservazioni disponibili rispetto alle altre categorie.
I tempi computazionali per le fasi di *training* e di *prediction* del modello sono:

- training time: *34min 4s*
- prediction time: *17min 55s*

	precision	recall	f1-score	support
Video_Games	0.94	0.91	0.92	60054
Patio_Lawn_and_Garden	0.79	0.88	0.84	95987
Office_Products	0.72	0.39	0.51	8652
Arts_Crafts_and_Sewing	0.88	0.74	0.80	14702
Musical_Instruments	0.94	0.89	0.92	13993
Digital_Music	0.90	0.78	0.84	28320
Prime_Pantry	0.74	0.74	0.74	52519
Industrial_and_Scientific	0.79	0.81	0.80	88582
accuracy			0.82	362809
macro avg	0.84	0.77	0.80	362809
weighted avg	0.83	0.82	0.82	362809

Figura 3: Risultati della rete sul test set

5 Task 2

5.1 Classificazione Opinion Mining

Volendo classificare l'opinione espressa da ciascun utente ma, non avendo a disposizione una variabile da utilizzare come target, si è reso necessario costruirne una. Si è deciso di utilizzare la variabile "overall" per stabilire se la recensione fosse positiva o meno ovviamente, supponendo che gli utenti fossero coerenti nell'assegnare le stelle ai prodotti.
La recensione è stata ritenuta positiva solo se associata ad un rating di 5 stelle ($x = 5$) mentre, in tutti gli altri casi ($x \leq 4$) è stata ritenuta negativa. Questa scelta è stata guidata dal seguente ragionamento: se il cliente non ha assegnato il massimo significa che non si ritiene pienamente soddisfatto del prodotto o del servizio ricevuto e che quindi un qualche aspetto debba essere migliorato. Inoltre, si è osservato che anche adottando questa strategia, la classe positiva risulta molto più numerosa dell'altra.

Anche in questo caso è stata utilizzata una *Recurrent Neural Network (LSTM)* con le stesse specifiche di quella utilizzata precedentemente, ma con layer di output composto da 2 unità (pari al numero di classi: positiva e negativa).

	precision	recall	f1-score	support
negative	0.80	0.62	0.70	121433
positive	0.83	0.92	0.87	241376
accuracy			0.82	362809
macro avg	0.82	0.77	0.79	362809
weighted avg	0.82	0.82	0.82	362809

Figura 4: Risultati della rete sul test set

Come era immaginabile, la rete classifica meglio gli elementi della classe maggioritaria (f1-score pari a 87%) rispetto a quella minoritaria (f1-score pari a 70%) nonostante sia stato effettuato un bilanciamento dei pesi con la funzione **sklearn** come nel task precedente. Pertanto, il modello ottiene risultati convincenti ottenendo un f1-score complessivo pari a 79%.

Proprio perché la definizione della variabile target è stata imposta dal programmatore, è stato implementato un algoritmo di clustering non supervisionato che permettesse di suddividere, in maniera indipendente dall'interpretazione umana, le recensioni.
I tempi computazionali per le fasi di *training* e di *prediction* del modello sono:

- training time: *33min 21s*
- prediction time: *17min 33s*

5.2 Clustering

Si è scelto l'algoritmo **K-Means** (servito dalla libreria **sklearn**), per la realizzazione del clustering non supervisionato delle recensioni. É bene specificare che il clustering viene effettuato sulle polarità delle reviews calcolate utilizzando *vaderSentiment*: è un lexicon e rule-based sentiment analysis tool pensato per valutare le opinioni espresse sui social. Per tale ragione, vaderSentiment, è sembrato adatto al calcolo della polarità anche in questo contesto. La polarità, p, varia nell'intervallo $p \in [-1, 1]$, dove 1 indica una recensione totalmente positiva. In questo modo è possibile conoscere la "percentuale" di positività della review (ed analogamente per la negatività).
In linea con l'idea espressa precedentemente - ossia che definire neutra una recensione non permette di sfruttarne le informazioni contenute - si è deciso di imporre il numero di cluster $k = 2$.

Il testo, opportunamente pre-processato, viene passato direttamente alla funzione **calculate sentiment analyser** che ne calcola la polarità e la salva nella colonna *sentiment* del dataframe così da tener traccia della review a cui è associata.
Infine, tali polarità vengono passate all'algoritmo K-Means, il quale calcola iterativamente i centroidi utilizzando la distanza euclidea.
I gruppi ottenuti confermano che la maggior parte delle recensioni risulti positiva, si veda la figura 5.

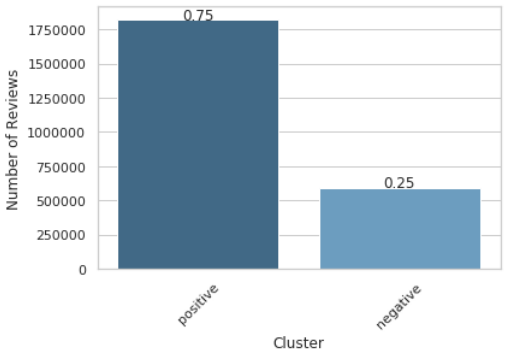


Figura 5: Suddivisione recensioni positive/negative con la definizione cluster

É stato utilizzato il coefficiente di *Davies–Bouldin* per la valutazione della bontà dei cluster trovati. Esso risulta pari a *0.48* si potrebbe provare ad aumentare il numero di cluster considerati al fine di migliorarne la qualità.

I tempi di computazione nella fase di ricerca delle polarità e di clustering sono:

- polarity assignments time: *1h 3min 3s*
- clustering time: *6.2 s*

6 Conclusioni

Sono stati condotti tre task, due dei quali di classificazione di testo. Nonostante i risultati siano soddisfacenti si potrebbe cercare di migliorare le performance dei modelli, sia ottimizzando i parametri delle reti che utilizzando delle classi più bilanciate. É da ricordare che la classificazione del topic, condotta nel primo task, dipende strettamente dall'interpretazione del testo e quindi sarà soggetta ad errori non completamente eliminabili.
La classificazione rispetto al rating (dove potrebbero esserci dei mismatch tra stelle attribuite e reale soddisfazione) ed il clustering delle polarità (il quale è sicuramente affetto da bias) forniscono due strumenti utili al fine di comprendere la soddisfazione/insoddisfazione degli utenti nella fase di post-vendita dei prodotti.

Riferimenti bibliografici

[1] *Appunti del corso di Text mining and Search aa 2019-2020*, Gabriella Pasi & Marco Viviani

[2] *Autonomic, neuroendocrine, and immune responses to psychological stress: The reactivity hypothesis*, JT Cacioppo, GG Berntson, WB Malarkey,