Università degli studi di
Milano-Bicocca

F1801Q104

Data Analytics

Amazon Reviews Analysis

Studenti:	*Matricole:*
Basso Matteo	807628
Ferri Marco	807130

Luglio 2019

Abstract

L'avvento dell'e-commerce sul Web ha portato con sé la possibilità di ricevere, per produttori e commercianti in generale, un'innumerevole quantità di recensioni spontanee da parte dei clienti. Se in certi contesti il grado di soddisfazione viene indicato tramite una valutazione numerica, si assiste anche alla presenza di recensioni di natura testuale che consento all'autore di esprimere un'opinione dettagliata in merito ad un determinato argomento o prodotto, in tutte le sue sfaccettature. Con l'aumentare dei portali espressamente dedicati a raccogliere recensioni dagli utenti, è nata la necessità di elaborare queste informazioni in maniera automatica.

La **sentiment analysis** è quindi la disciplina che applicata ad un insieme di dati che rappresentano opinioni ha l'obiettivo di comprendere il pensiero di una comunità riguardo un certo aspetto, ad esempio nell'ottica di ottimizzare le vendite di un prodotto. É questo il caso di Amazon, nota società di commercio elettronico che basa la sua forza proprio sull'opinione che gli utenti hanno nei confronti di quanto venduto dalla piattaforma e sulla fiducia riposta verso l'azienda stessa.

Analizzando le recensioni dei prodotti su Amazon è possibile non solo comprendere l'opinione degli utenti riguardo i prodotti specifici, ma anche sviluppare dei **sistemi di raccomandazione** basati sulle preferenze dei singoli con lo scopo di aumentare le vendite e la fidelizzazione dei propri clienti.

In questo elaborato si analizza un dataset di recensioni ufficiali Amazon per sperimentare diverse tecniche di analisi del sentiment e presentare un modello di raccomandazione dei prodotti agli utenti. Infine viene presentata una demo interattiva per la dimostrazione di alcune delle tecniche descritte.

Indice

Elenco delle figure

Elenco delle tabelle

1 Introduzione

Lo studio ha lo scopo di condurre diversi tipi di analisi sulle recensioni del noto portale e-commerce Amazon (2). In questa sezione viene presentata una breve introduzione al problema, il dataset utilizzato e gli strumenti che sono stati impiegati per portare a termini gli obiettivi prefissati.

1.1 Dominio di riferimento

Sempre maggiore è il numero di siti web che fanno delle recensioni il proprio principale business. Si pensi ai portali dedicati alla valutazione di località turistiche, film o ristoranti. Allo stesso modo, anche Amazon basa sulle recensioni parte della propria fidelizzazione clienti.

Trattandosi di dati testuali prodotti dagli utenti per valutare i prodotti acquistati, le recensioni esprimono attraverso il linguaggio naturale le impressioni dell'autore, le quali possono assumere un carattere di natura positiva o negativa. In questo contesto è inoltre pratica comune associare al proprio pensiero un punteggio che esprima una valutazione del prodotto su una scala numerica. Se questa informazione è fondamentale per i clienti della piattaforma, poiché permette di capire a colpo d'occhio quale possa essere la qualità dell'articolo che si sta considerando di acquistare, è anche vero che tale punteggio possa rappresentare un aiuto importante per riassumere in forma strutturata (e pertanto più facilmente comprensibile da un computer) l'opinione dell'autore riguardo un certo argomento. Pertanto, analizzare congiuntamente il testo di una recensione ed il punteggio ad essa associato è fondamentale per determinare una correlazione fra il linguaggio naturale e l'opinione dell'utente nei confronti del prodotto, anche detta **sentiment**. Inoltre, l'elaborazione del testo può considerare il piano morfologico del linguaggio per derivare l'opinione espressa riguardo le diverse caratteristiche del prodotto, dette **aspect**, espresse nella recensione. Ciò è particolarmente utile per migliorare la qualità dei propri prodotti e ottenere quindi un vantaggio sul piano commerciale.

Infine, l'analisi delle recensioni può essere utile anche per ottenere un'approssimativa profilazione di un utente; questa può rivelarsi particolarmente rilevante dal punto di vista del marketing, ad esempio per dare suggerimenti ad altri utenti che dimostrano di avere le medesime preferenze. Allo stesso modo, può essere costruito un sistema di suggerimenti basato su prodotti simili o solitamente venduti insieme. Tali tecniche vengono dette di **collaborative filtering**.

Il seguente elaborato si pone l'obiettivo di sperimentare con i concetti appena presentati per determinare quanto sia possibile ottenere attraverso l'analisi delle recensioni di un portale e-commerce come Amazon.

1.2 Dataset

Il dataset utilizzato per effettuare le analisi è fornito ufficialmente da Amazon e contiene recensioni redatte in lingua inglese fra il 1996 e il 2014, per diverse categorie di prodotti (4). Poiché contenente un gran numero di recensioni che coinvolgono altrettanti utenti e prodotti, talvolta poco partecipativi all'interno del portale e-commerce, si è scelto di utilizzare ai fini del progetto una versione rielaborata del suddetto dataset, reperibile qui: `http://jmcauley.ucsd.edu/data/amazon`. Julian McAuley (10), professore dell'Università di San Diego, ha estrapolato dal dataset originale solamente i record delle recensioni riguardanti prodotti e utenti con almeno cinque recensioni ciascuno; ciò viene definito in teoria dei grafi con il termine **k-core**, cioè un sottografo in cui tutti i nodi hanno un grado almeno pari a `k` (21).

1.3 Strumenti

Per effettuare le analisi mostrate in questo elaborato si è utilizzato prevalentemente il linguaggio di programmazione open source Python, attraverso l'utilizzo di Google Colab (8) per la creazione di Jupyter Notebook interattivi. Le specifiche librerie di volta in volta utilizzate saranno presentate contestualmente alle singole analisi qualora lo si ritenesse necessario ai fini di una migliore comprensione del problema.

Per l'analisi della rete di prodotti e utenti ricavata dal dataset si è utilizzato il software Cytoscape (6), dedicato appositamente all'integrazione e visualizzazione di grafi anche molto complessi. Nell'ultima parte del documento verrà infine presentata una demo Web-based per testare con mano le potenzialità dei concetti studiati.

2 Basic Analysis

Il dataset è stato reperito in formato JSON e successivamente letto attraverso Python per la memorizzazione in una struttura adatta ad essere elaborata efficientemente dal linguaggio. A tal fine si è scelto di utilizzare la liberia `pandas` (14), dedicata all'analisi di dati anche molto voluminosi. É interessante notare come la lettura del dataset, ed in particolare la conversione di quest'ultimo dal formato JSON, sia stata una delle operazioni computativamente più *time-consuming* fra tutte quelle effettuate, a dimostrazione del fatto che `pandas` sia successivamente in grado di elaborare molto velocemente le informazioni memorizzate all'interno dei cosiddetti `DataFrame`.

2.1 Schema

Il dataset può essere descritto molto velocemente, poiché rappresentabile attraverso una singola struttura tabellare composta dai campi illustrati in tabella 1.

Tabella 1: Schema originale del dataset

Campo	Descrizione
reviewerID	ID utente
reviewerName	Nome utente
asin	ID prodotto
reviewText	Testo della recensione
summary	Titolo della recensione
helpful	Utilità della recensione
overall	Punteggio
reviewTime	Timestamp in formato string
unixReviewTime	Timestamp in formato unix

Ogni record è la rappresentazione di una singola recensione, svolta da parte di un utente per un certo prodotto nella data indicata. Mentre per quanto riguarda l'utente si è in possesso sia dell'ID che del nome (che non verrà utilizzato), il prodotto è rappresentato nel dataset solamente attraverso un ID (`asin`); ulteriori dettagli sul prodotto possono essere ricavati contattando il creatore del dataset attraverso un'apposita richiesta, che non è stata effettuata poiché tali informazioni si sono rivelate ininfluenti ai fini del progetto in esame.

Per quanto riguarda i campi relativi alla recensione, è possibile visualizzare sia il titolo che il testo del corpo, oltre al punteggio espresso su una scala numerica da 1 a 5. Poiché ciò rappresenta il sentiment associato alla recensione, questo attributo

costituirà anche la variabile target per l'analisi e l'addestramento dei modelli di machine learning.

Il campo `helpful` presenta un dominio non particolarmente definito e non è stato considerato per l'analisi, mentre per il timestamp si è computato un attributo `date` di tipo *datetime*; infine, per semplificare il concetto associato ad ogni campo si è scelto di rinominarli. Al termine della trasformazione, il dataset risultante è il seguente:

Tabella 2: Schema modificato del dataset

Campo	Descrizione
userID	ID utente
productID	ID prodotto
text	Testo della recensione
summary	Titolo della recensione
rating	Punteggio
date	Timestamp in formato datetime

2.2 Dimensioni

Dalla fonte citata nell'introduzione, da cui è stato reperito il dataset, è possibile notare che siano presenti varie possibilità di download. Diverse categorie di prodotti sono state testate durante l'intero sviluppo dello studio, ma ai fini di redigere questo elaborato verrà considerato il dataset 5-core chiamato *Clothing, Shoes and Jewelry*, contenente recensioni relative al mercato dei vestiti, delle scarpe e dei gioielli.

Questo contiene un totale di **278.677 recensioni**, divise fra **39.387 utenti** e **23.033 prodotti**. Ciò significa una media di circa 7 recensioni ad utente e 12 recensioni per prodotto. Nel capitolo 3 verrà mostrata se tale media è anche effettivamente rappresentativa del dataset oppure diversi prodotti ed utenti presentano un numero di recensioni sbilanciato.

2.3 Distribuzione di `rating`

In figura 1 è possibile osservare la distribuzione del campo `rating`, che costituisce l'elemento fondamentale su cui costruire un modello supervisionato di sentiment analysis. Come è possibile osservare, questa variabile è fortemente sbilanciata sui valori 4 e 5, motivo che a fronte di alcune considerazioni future porterà il problema ad essere prima binarizzato e successivamente downsampled per ridurre il gap fra la classe positiva e quella negativa.

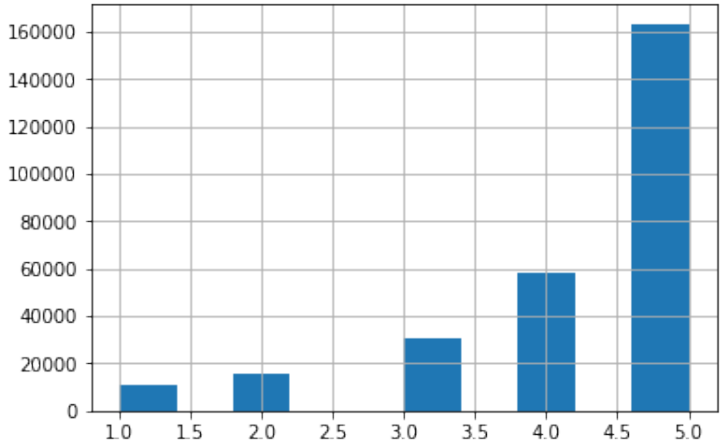


Figura 1: Distribuzione del campo `rating`

2.4 Analisi temporale business-oriented

Si era già accennato che la fonte fornisca recensioni per più di dieci anni di vendite. Più precisamente, il dataset qui considerato considera recensioni fra il marzo 2003 e giugno 2014 secondo la distribuzione indicata in figura 2. Pur essendo la maggior parte delle recensioni concentrata negli ultimi 3 anni, qualsiasi sia il filtro temporale applicato non c'è differenza nella distribuzione della variabile target.

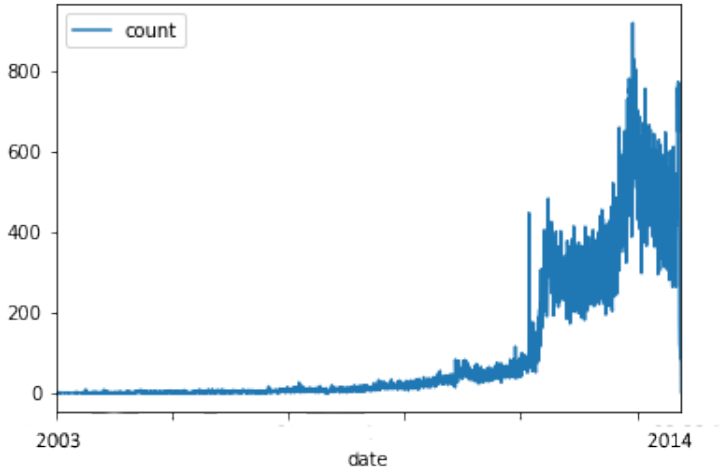


Figura 2: Distribuzione delle recensioni per data

Considerato il dominio in esame, potrebbe essere molto utile nell'ottica di prendere decisioni di business per i produttori analizzare come le recensioni si distribuiscono anche in un anno relativamente ai mesi e i giorni della settimana, per evidenziare eventuali periodi di maggiore attività su Amazon. A tal proposito sono pertanto stati prodotti i grafici in figura 3 e 4.

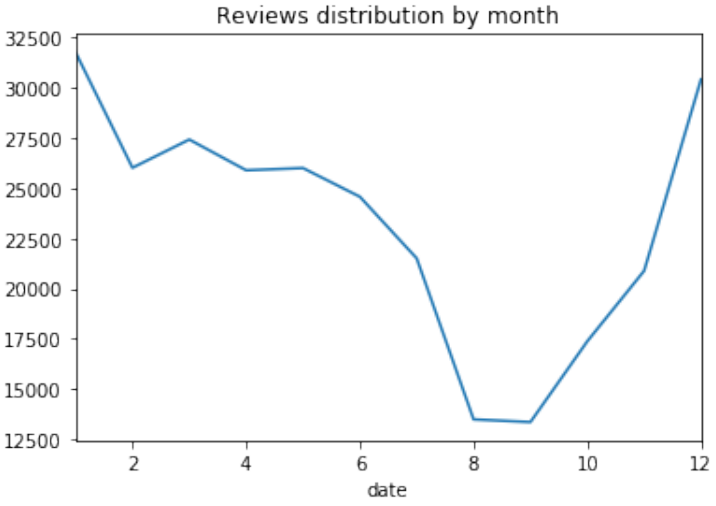


Figura 3: Distribuzione delle recensioni per mese

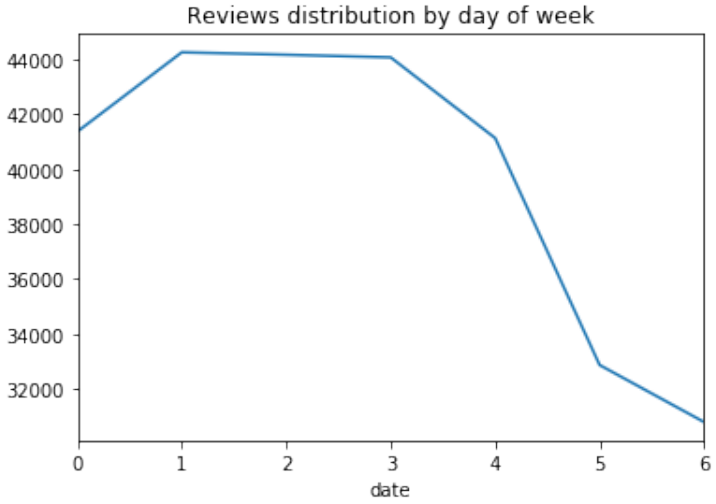


Figura 4: Distribuzione delle recensioni per giorno della settimana

Osservando la distribuzione per mese, è possibile notare come la maggiore concentrazione di recensioni si riscontri nel periodo compreso fra il black friday (Novembre) e tutte le vacanze natalizie, probabilmente proprio per via degli sconti e della necessità di comprare regali ai propri conoscenti. É durante questi mesi e quelli immediatamente precedenti che i produttori dovrebbero concentrare maggiormente le proprie campagne pubblicitarie. Tale attività decrementa gradualmente nei mesi successivi a Gennaio, per raggiungere livelli particolarmente bassi ad Agosto e Settembre.

Concentrandoci invece sui dati che riassumono l'andamento di acquisti durante la settimana, è chiaramente evidente come sabato e domenica costituiscano i giorni in cui gli utenti Amazon sono meno propensi a fare recensioni.

Analizzando quindi la distribuzione del `rating` suddiviso per mesi e giorni, emerge comunque che i risultati sono del tutto analoghi a quelli della figura 1. Da questa osservazione si potrebbe concludere che non vi sono particolari periodi temporali durante i quali i clienti sono più propensi a dare recensioni più positive o negative. Ciò su cui le aziende dovrebbero concentrarsi, da questo punto di vista, è solamente la necessità di raggiungere più persone possibili nei momenti di maggiore attività.

3 Network Analysis

Nonostante tale elaborato non si concentri sugli aspetti che sia possibile considerare attraverso la rete costituita da utenti, prodotti e recensioni, questo capitolo presenta comunque una breve analisi di tale rete poiché può essere utile ad analizzare il dataset da un punto di vista descrittivo. L'analisi della rete si mostrerà inoltre utile anche per il capitolo 7.

Con un dataset di recensioni è possibile costruire tre diverse tipologie di rete:

- prodotti e utenti nella stessa rete, collegati attraverso le recensioni
- rete di soli prodotti, collegati tramite similarità
- rete di soli utenti, collegati tramite similarità

Le due reti che considerano le similarità di prodotti o utenti non sono particolarmente semplici da costruire; esse richiedono che venga definito il concetto stesso di similarità. Ad esempio, prodotti simili potrebbero essere quelli che sono recensiti dai medesimi utenti ed analogamente accade per gli utenti che recensiscono gli stessi prodotti. Questo tipo di relazione potrebbe essere utilizzata per definire delle categorie di utenti o prodotti attraverso algoritmi di *clustering* che andrebbero successivamente interpretati sui metadati contenuti nei nodi, prodotti o utenti che siano. Poiché nel dataset utilizzato mancano questo tipo di informazioni, si è pensato potesse essere poco significativo costruire questi tipi di reti.

Al contrario, ci si è concentrati sulla prima alternativa che è direttamente estraibile dal dataset e consente di carpire se le recensioni coinvolgono omogeneamente diversi prodotti o si suddividono in diverse componenti connesse o cluster.

3.1 Struttura della rete

Per generare la rete ci si è affidati alla libreria Python `NetworkX` (12), che consente di generare, modificare e analizzare grafi anche di natura complessa. Per quanto riguarda la parte di visualizzazione si è invece utilizzato il software Cytoscape, che la libreria Python stessa suggerisce per la visualizzazione di reti molto grosse.

La rete è stata generata dal `pandas DataFrame`, considerando le colonne `userID` e `productID` per rappresentare rispettivamente i nodi sorgenti e i nodi target. Ciò significa che si è creata una rete i cui nodi rappresentassero alternativamente utenti o prodotti, nella quale i primi hanno sempre archi diretti verso i secondi: le recensioni. Si è quindi tratto un grafo orientato nel quale i nodi degli utenti hanno solamente archi uscenti e i nodi dei prodotti solo archi entranti. Ciò è particolarmente utile per l'analisi di *in* e *out degree* che si vedrà nella sezione successiva numero 3.2.

Si è ottenuta quindi una rete composta da **62.420 nodi** (utenti + prodotti) e **278.677 archi** (recensioni), quest'ultimi pesati sul `rating` delle recensioni stesse. Purtroppo una rete così grossa è particolarmente difficile da mostrare correttamente a video anche con Cytoscape, come è possibile notare nell'immagine 5.

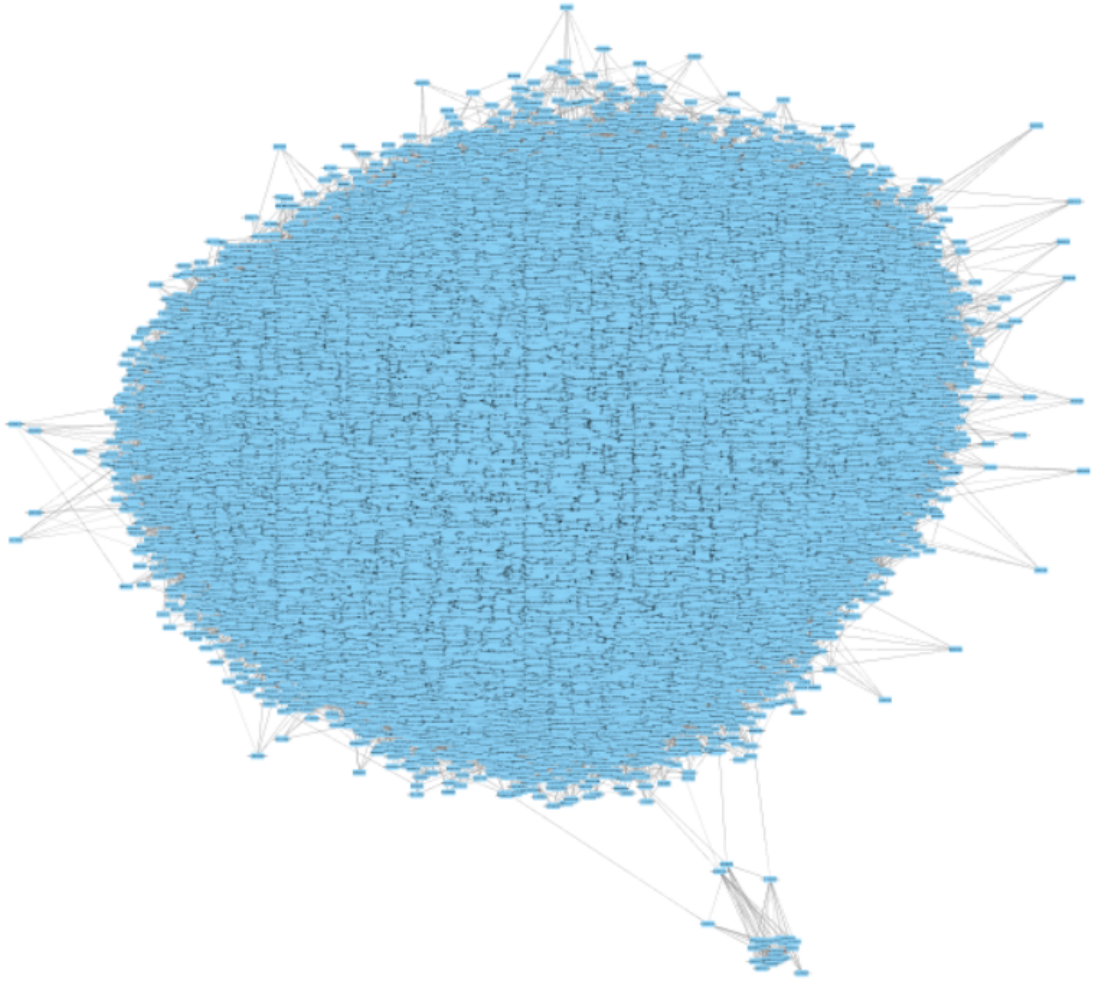


Figura 5: Rete di utenti e prodotti, collegati tramite recensioni

É facilmente possibile notare, per quanto sia complessa, che la rete è composta da un'unica grande componente connessa nella quale la maggior parte dei prodotti e degli utenti sono fortemente interconnessi senza apparenti raggruppamenti.

Tuttavia è interessante notare che dall'agglomerato centrale si sviluppano due fenomeni interessanti: alcuni nodi appaiono separati esternamente dal resto del gruppo, come fossero elementi con la voglia di distinguersi dal resto. É il caso ad esempio

di quanto accada in figura 6, che analizza la zona alta destra della rete. Queste due *spike* esterne alla rete sono rispettivamente un prodotto e un utente, come si può notare dalla direzionalità degli archi che li coinvolgono. Purtroppo non è dato sapere se questi vengano posizionati così da Cytoscape perché rappresentano effettivamente dei casi particolari, o solo per una pura casualità.

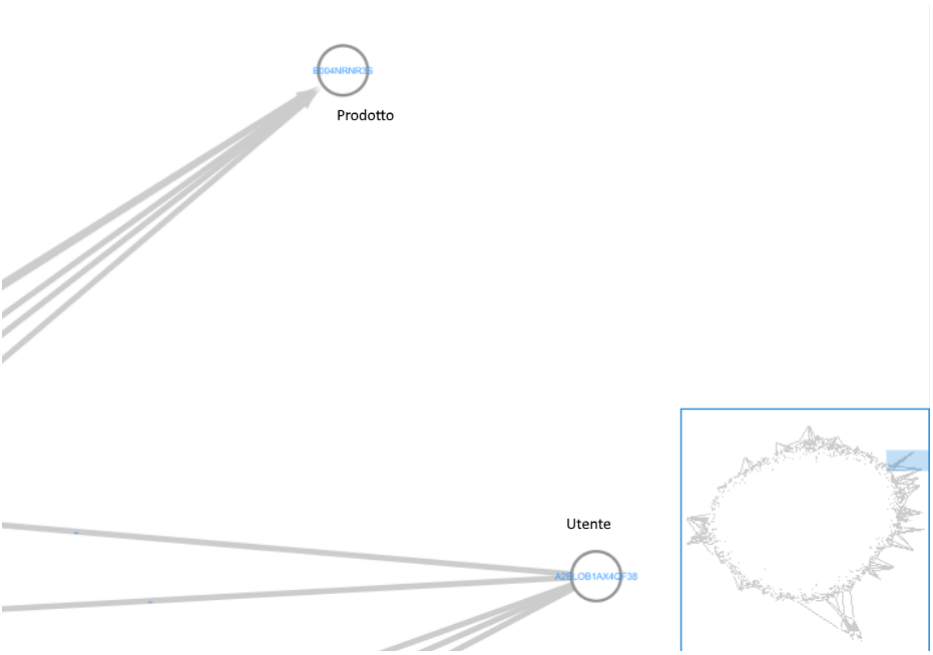


Figura 6: Esempio di nodi separati dal raggruppamento centrale

Di maggiore interesse appare invece la parte bassa della rete, rappresentata in figura 7. In questa zona sembra comparire una sorta di cluster vero e proprio, al suo interno fortemente connesso ma collegato alla rete "principale" solo attraverso pochi nodi che svolgono il ruolo di ponte: 3 utenti e 1 prodotto. Nonostante non ci è dato sapere cosa possa rappresentare tale cluster, vista la presenza di alcuni utenti e prodotti con un alto numero di recensioni ma isolati dal resto della rete si potrebbe dedurre si tratti del mercato di un paio d'articoli particolarmente di nicchia, probabilmente provenienti da mercati stranieri o destinati ad un tipo di vendita fra gruppi ristretti di persone con particolari accordi.

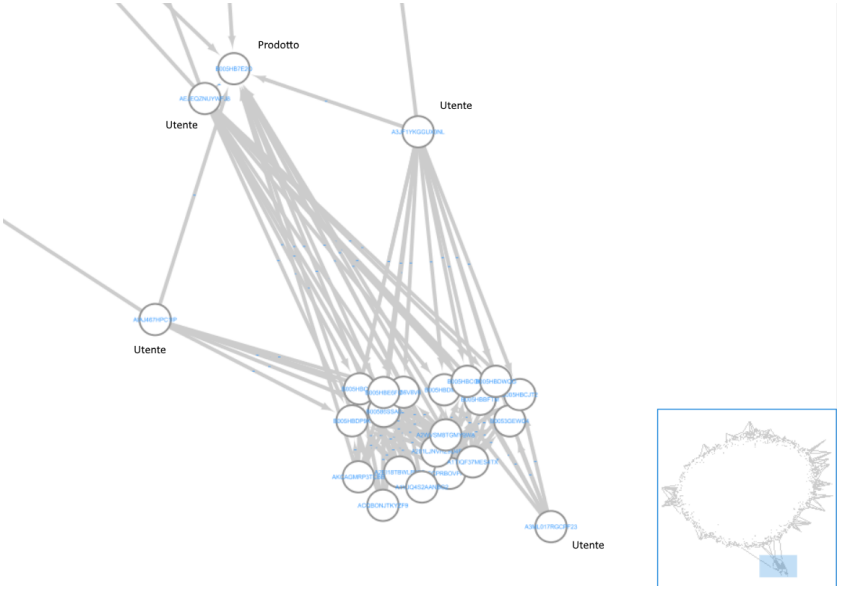


Figura 7: Cluster inferiore della rete

3.2 Grado dei nodi

Oltre alla visualizzazione della struttura, una delle analisi fondamentali che si può fare sulla rete è l'analisi del grado dei nodi. Per come essa è stata costruita, l'*out-degree* di un nodo rappresenta le recensioni fatte da un utente, mentre l'*in-degree* costituisce quelle ricevute da un prodotto. É chiaro che ogni nodo avrà sempre uno dei due gradi uguale a zero.

Particolarmente rilevante è quindi la distribuzione dei due gradi, al fine di comprendere se all'interno del dataset compaiano utenti e prodotti con un eguale distribuzione di recensioni, piuttosto che invece utenti particolarmente attivi o prodotti molto popolari. Si fa inoltre notare che, come accennato nella sezione 1.2, il grado minimo di ciascun nodo è pari a cinque poiché il dataset è ottenuto attraverso l'estrazione di un sottografo 5-induttivo da una rete molto più grande.

Osservando in figura 8 ed escludendo dall'analisi i nodi con *out-degree* pari a zero (cioè i prodotti), è possibile notare che nonostante la media di recensioni per utente sia pari a 7 (vedere sezione 2.2), in realtà il grado per ciascun nodo è distribuito in maniera piuttosto varia con una frequenza che diminuisce all'aumentare del grado. Solo sei sono gli utenti con più di 60 recensioni, di cui uno ne conta ben 136. Eccoli:

```
[('A2J4XMWKR8PPD0', 136),('A2GA55P7WGHJCP', 76),('A2KBV88FL48CFS', 69),
 ('AENH50GW3OKDA', 68),('A2V5R832QCSOMX', 62),('AVUJP7Z6BNT11', 61)]
```

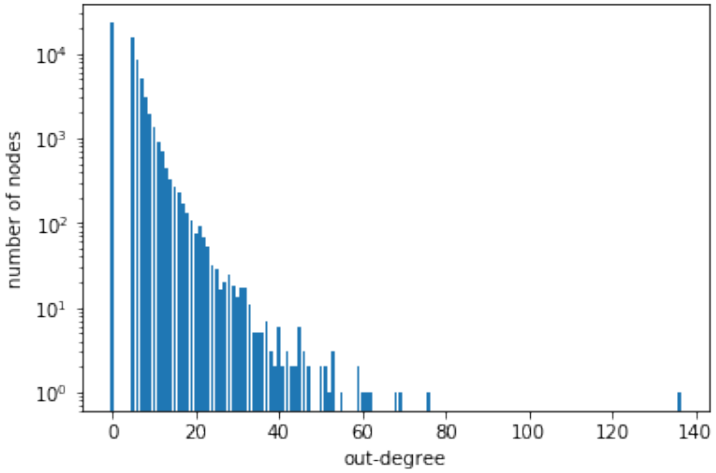


Figura 8: Distribuzione dell'*out-degree* per gli utenti

Anche considerando il grafico della distribuzione degli *in-degree* (figura 9) si nota che molti prodotti si discostano dalla media di 12 recensioni precedentemente calcolata e addirittura esistono all'interno del dataset prodotti particolarmente popolari con più di 200 recensioni. Sebbene sia un numero elevato, probabilmente non è necessario affinché questi possano definirsi *hub*, vista la presenza nella rete di ben 23.033 prodotti per più di 250.000 recensioni. Di seguito i quattro prodotti più popolari:

```
[('B005LERHD8', 441),('B005GYGD7O', 286),('B008WYDP1C', 249),
 ('B0058XIMMM', 241),('B00CKGB85I', 225),('B007RD9DS8', 217)]
```

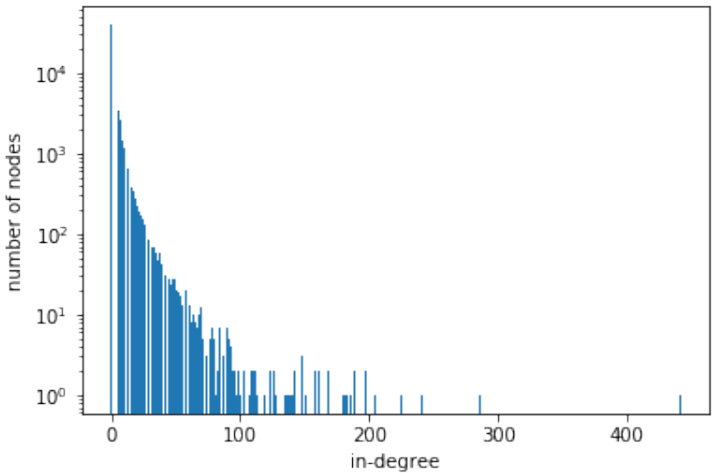


Figura 9: Distribuzione dell'*in-degree* per i prodotti

4 Sentiment Analysis

La sentiment analysis è il processo che, attraverso l'elaborazione del linguaggio naturale, consente di identificare e estrarre da un testo un'opinione: positiva, negativa o neutrale. Ci sono varie tecniche che è possibile adottare per svolgere sentiment analysis, che ha inizio dal pre-processing del testo al fine di identificare le parole utilizzate e successivamente classificarne il sentiment.

La polarità espressa da ciascuna parola, che successivamente può contribuire all'identificazione di un sentiment per l'intera frase, può essere reperita in un vocabolario linguistico piuttosto che attraverso un dizionario appreso dai dati a disposizione. Quest'ultimo approccio è realizzabile qualora si abbia un elemento nel dataset da usare come target per la classificazione durante l'apprendimento supervisionato dei termini associati a sentimenti positivi o negativi.

In questo capitolo viene presentata solamente l'analisi preliminare effettuata sul testo delle recensioni nel dataset. Tale analisi sarà seguita nei capitoli successivi da due diversi approcci per l'identificazione del sentiment.

4.1 Assunzioni

A causa dell'elevato sbilanciamento della variabile target `rating`, già emerso nel capitolo 2.3, si è considerato che la maggior parte degli algoritmi di apprendimento supervisionato di sentiment analysis avrebbero performato male sul dataset originale e non filtrato. Per questo motivo, in alcune fasi della sentiment analysis si è deciso di mettere in atto delle contromisure per arginare il problema, e a posteriori è possibile dire che tali scelte si sono rivelate corrette per ottenere dei modelli di predizione (capitolo 5) con prestazioni migliori.

4.1.1 Binarizzazione

La prima scelta effettuata per ribilanciare le classi del target è stata quella di binarizzare l'outcome. Se il `rating` è espresso nel dataset su un dominio discreto da 1 a 5 (figura 1), considerata l'elevata percentuale di recensioni con punteggio 4 o 5 si è pensato di raggruppare quest'ultime in un'unica classe che esprimesse un sentimento positivo; i valori minori o uguali a 3 sono invece stati raggruppati per rappresentare la classe negativa. In funzione del ridotto numero di recensioni negative, si è preferito in questa sede escludere la considerazione di un'eventuale classe per il sentiment neutrale, anche in funzione della difficoltà di classificazione.

4.1.2 Undersampling

A seguito della binarizzazione si sono quindi ottenute 221.597 e 57.080 recensioni con sentiment rispettivamente positivo e negativo. Essendo ancora insufficiente per ottenere buoni modelli di machine learning si è quindi proceduto effettuando undersampling della classe di maggioranza, attraverso una funzione che ha portato alla generazione di un dataset con 57.080 recensioni per entrambi i sentiment considerati.

4.2 Elaborazione del testo

Affinché qualsiasi operazione di analisi testuale possa essere portata a termine con successo è necessario che i testi considerati vengano sottoposti a una fase di preprocessing davvero fondamentale. Questa si compone di diversi step, alcuni dei quali facoltativi, che hanno l'obiettivo di suddividere il testo in parole e successivamente normalizzarle sulla base di scelte operative o regole linguistiche.

Vengono qui presentate le fasi generiche di elaborazione di un testo e quindi descritte le modalità attraverso le quali sono state implementate. Si consideri che si sta considerando testo scritto in lingua inglese, pertanto non soggetto alla presenza di caratteri che richiedono trattazioni particolari.

La libreria utilizzata per portare a termine queste operazioni è `NLTK` (Natural Language Toolkit) (11), ampiamente utilizzata dalla community.

4.2.1 Tokenizer

La prima fase di elaborazione è sempre quella di tokenizzazione, cioè la suddivisione del testo in *token*, solitamente parole. Per questa operazione è possibile sfruttare spazi e segni di interpunzione, ma anche caratteri speciali, numeri o l'alternarsi di lettere maiuscole e minuscole.

Diverse tecniche sono state utilizzate per questo elaborato.
In prima fase si è scelto di effettuare un'analisi relativa alle parole più comuni, quindi si è scelto di costruire un tokenizer basato su una espressione regolare personalizzata: `[^\d\W][\w\']*`. Questa consente di considerare come *token* tutte le parole, ma non i numeri, e soprattutto di preservare eventuali contrazioni tipiche della lingua inglese per i verbi o il genitivo sassone.
Per le fasi successive della sentiment analysis si è invece optato per un tokenizer più standard, solitamente fornito di default dalla libreria in uso.

4.2.2 Normalizer

Ottenuto l'elenco dei *token* costituenti un testo è necessario che essi vengano normalizzati. In questa fase si possono eseguire diverse operazioni, come ad esempio la rimozione o modifica di alcuni caratteri (se non precedentemente fatto direttamente sul testo), ma soprattutto e molto più frequentemente la conversione a caratteri minuscoli, come nel caso in esame.

4.2.3 Stopwords

Prettamente dipendente dalla lingua che si sta considerando, la rimozione delle stopwords consiste nell'eliminazione dall'elenco di *token* delle parole altamente ricorrenti in una lingua: articoli, preposizioni, alcuni aggettivi, ... É importante rimuovere le stopwords poiché non forniscono informazione alcuna e potrebbero anche degradare le performance delle analisi successivamente applicate al testo.

Per ogni lingua è solitamente a disposizione delle diverse librerie una lista di stopwords. Per questo progetto si è scelto di ottenere la lista di stopwords inglesi ma successivamente modificarla per evitare la rimozione di alcune parole che si sono ritenute potenzialmente utili per la valutazione delle recensioni: "not" e "but" sono due esempi particolarmente rilevanti.

4.2.4 Stemmer

Anche la fase di stemmer (come la successiva qui presentata) è fortemente dipendente dalla lingua che si sta analizzando, questa volta derivato dalle regole grammaticali che costituiscono la fine delle parole. Si pensi ad esempio ai plurali o i participi in inglese, costituiti con regole ben precise: l'applicazione di uno stemmer riduce queste parole alla propria forma base contraendo le ultime lettere, pratica particolarmente utile per l'identificazione di concetti ricorrenti. Fra quelle fin'ora descritte lo stemming è l'operazione più dispendiosa in termini computazionali, quindi deve essere applicato di volta in volta effettuando una valutazione di costi e benefici.

4.2.5 Lemmatizer

Simile allo stemming, la lemmatization è un tipo di elaborazione linguistica ancora più potente. Anziché limitarsi a troncare la fine di alcune parole come fa uno stemmer, il lemmatizer converte ciascuna nel proprio lemma linguistico: i verbi all'infinito, per esempio. Ciò consente di ottenere una lista di *token* in cui parole simili che riguardano lo stesso concetto semantico vengano ricondotte al medesimo lemma. È sicuramente la fase più dispendiosa fra tutte e sia per motivi computazionali che di reperibilità dei dizionari è quella in genere meno utilizzata.

4.2.6 Un esempio

A dimostrazione di quanto appena esplicato, sia per chiarificare i concetti presentati che per fornire un esempio di elaborazione applicata al testo per la sezione successiva, viene qui mostrato una frase esemplificativa per dimostrare il pre-processing testuale applicatovi.

```
Testo:
"Hi! This... isn't a beautiful sentence with some interesting
$70 and $5,50 features like people's names and Mr.  Fox thoughts
for number such as 23, 4 and 7 or peer2peer and wi-fi with
snake_case but not kebab-case."

Tokenization e lowercasing:
['hi', 'this', "isn't", 'a', 'beautiful', 'sentence', 'with',
'some', 'interesting', 'and', 'features', 'like', "people's",
'names', 'and', 'mr', 'fox', 'thoughts', 'for', 'number',
'such', 'as', 'and', 'or', 'peer2peer', 'and', 'wi', 'fi',
'with', 'snake_case', 'but', 'not', 'kebab', 'case']

Rimozione stopwords:
['hi', "isn't", 'beautiful', 'sentence', 'interesting', 'features',
'like', "people's", 'names', 'mr', 'fox', 'thoughts', 'number',
'peer2peer', 'wi', 'fi', 'snake_case', 'but', 'not', 'kebab', 'case']

Stemming:
['hi', "isn't", 'beauti', 'sentenc', 'interest', 'featur', 'like',
'peopl', 'name', 'mr', 'fox', 'thought', 'number', 'peer2peer',
'wi', 'fi', 'snake_cas', 'but', 'not', 'kebab', 'case']
```

4.3 Parole più comuni

A seguito dell'elaborazione testuale apportata sui campi `text` e `summary` del dataset, come descritto nella precedente sezione, si sono quindi potute calcolare le parole più ricorrenti all'interno del dataset. Per ottenere questo risultato e soprattutto visualizzarlo in maniera accattivante si è scelto di utilizzare la libreria `WordCloud` (22), appositamente pensata per ottenere questo tipo di risultato.

Nelle figure 10, 11, 12 vengono quindi riportate le parole più comuni all'interno del campo `text` del dataset, prima considerando tutte le recensioni e successivamente solo quelle positive e negative.

Figura 10: Parole più comuni in TUTTE le recensioni

Figura 11: Parole più comuni nelle recensioni POSITIVE

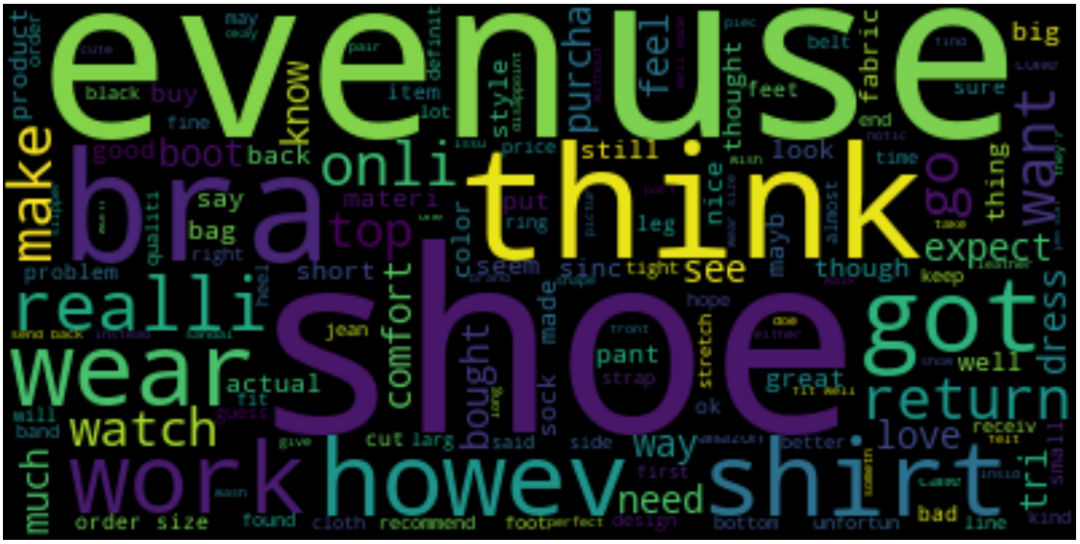


Figura 12: Parole più comuni nelle recensioni NEGATIVE

4.3.1 Alcune considerazioni

Come si evince dalle WordCloud, è possibile notare che alcune parole siano ricorrenti per entrambi i sentiment. Questo comportamento è del tutto normale per alcune parole tipiche del linguaggio che si sta considerando, utilizzate spesso in tutte le recensioni. Anche quelle negative per esempio potrebbero contenere parole di natura positiva, ma spesso negate attraverso l'utilizzo di avverbi di negazione.

Questo è il motivo per cui molte parole, seppur sembrerebbe logico inserirle nella lista di stopwords affinché non vengano impropriamente considerate, possono comunque rivelarsi utili sia per task di sentiment prediction (capitolo 5) che di aspect based sentiment analysis (capitolo 6). Questa affermazione è confermata da alcuni test eseguiti proprio inserendo alcune parole comune ad entrambi i sentiment all'interno della lista di stopwords: le performance dei risultati subivano un peggiormento.

5 Sentiment Prediction

Quando si ha a che fare con un dataset di testi su cui si vuole fare sentiment analysis, uno degli aspetti più interessanti è la possibilità di predire il sentiment di nuove recensioni attraverso l'analisi del linguaggio naturale. Ciò è possibile attraverso la classificazione delle parole utilizzate in sentiment positivo e negativo; tale operazione può essere eseguita attraverso l'utilizzo di un vocabolario linguistico, oppure tramite apprendimento di un dizionario dal testo purché il dataset metta a disposizione una variabile target su cui fare apprendimento supervisionato.

Quest'ultimo caso rappresenta perfettamente la situazione che si sta tentando di analizzare, ed è per questo che si è scelto di verificare se un algoritmo di Machine Learning di apprendimento supervisionato sia in grado di ottenere buoni risultati in termini di analisi del linguaggio naturale.

In questo capitolo vengono quindi descritte le fasi generiche necessarie ad effettuare supervised sentiment prediction, implementate poi attraverso l'utilizzo della libreria `scikit-learn` (17), facilmente utilizzabile e ampiamente diffusa per la fruizione di svariati modelli di Machine Learning.

5.1 Bag of words

In ogni occasione in cui si lavora con il testo, che sia essa per apprendimento o anche semplicemente per effettuare una ricerca, è necessario che questo venga memorizzato in una modalità che lo renda facilmente rappresentabile e recuperabile.

Uno dei modelli più utilizzati per ottenere questo obiettivo è il cosiddetto bag of words, implementato da `scikit-learn` attraverso la classe `CountVectorizer`. Ciò consiste nella rappresentazione del testo in parole, ad ognuna delle quali viene associato un identificativo; successivamente, di ciascuna parola viene salvata la lista di **occorrenze** nei documenti che costituiscono il dataset (i record, ad esempio). Così facendo è possibile ottenere di fatto una matrice molto grande e di natura sparsa, contenente le informazioni necessarie a rappresentare il testo per parole, d'ora in poi chiamate più correttamente **termini**.

5.2 Pesatura dei termini (TF-IDF)

Suddividendo il testo originale in quello che è stata definita bag of words, si ottiene di fatto un **dizionario** rappresentativo della base di dati testuale. Nonostante sia stato precedentemente detto che per ogni termine si memorizzino le occorrenze nei documenti del dataset, sarebbe più corretto dire che di questi dovrebbero essere salvate le **frequenze**, opportunamente normalizzate, che rappresentano una più accurata modalità di rappresentazione dell'importanza di ciascuna parola nel testo.

Un modello molto famoso e utilizzato oggi, specialmente nell'ambito dell'Information Retrieval, è conosciuto con il nome di **TF-IDF (term frequency - inverse document frequency)**. Tale statistica è in grado di computare l'importanza di un termine per ciascun documento presente in un corpus (collezione di testi), sia facendo riferimento alle occorrenze nel documento stesso che alla popolarità della parola nell'intera collezione. Il peso di un termine i nel documento j si esprime con la formula sottostante, in cui il primo fattore del prodotto è la TF (normalizzata) e il secondo invece la IDF (N è il numero di documenti, mentre d_{fi} rappresenta il numero di documenti che contengono il termine considerato).

$$w_{ij} = \frac{tf_i}{\max tf_j} \times log\frac{N}{d_{fi}}$$

Il concetto di TF-IDF è facilmente utilizzabile da `scikit-learn` attraverso la classe `TfidfTransformer` per ricavare dalla semplice bag of words una matrice computata secondo TF-IDF. Questo è successivamente utilizzato da un modello di classificazione a piacere per effettuare l'apprendimento supervisionato.

La libreria permette inoltre di accorpare le funzionalità di `CountVectorizer` e `TfidfTransformer` nell'utilizzo di un'unica classe: `TfidfVectorizer`. Tale classe è inoltre in grado di effettuare la pre-elaborazione del testo esplicata nel capitolo 4.2, previa configurazione con appositi parametri dall'utilizzo estremamente facile.

5.2.1 Termini più rilevanti

Applicando i concetti appena descritti al dataset in esame si ottiene il peso di ciascun termine presente nel testo delle recensioni. Per completezza è interessante notare quali sono i termini che TF-IDF considera più rilevanti dopo la computazione del dizionario. Questi sono mostrati nella tabella 3

Tabella 3: Termini più rilevanti secondo TF-IDF

ID	Termine	Peso
4121	but	0.040886
19248	not	0.037178
25748	size	0.033456
31509	veri	0.032994
10467	fit	0.032224
16499	look	0.030117
16151	like	0.029839
32213	wear	0.029403
25314	shoe	0.027903
16656	love	0.025427

5.3 Modelli di predizione

Appreso il dizionario del testo, è possibile utilizzarlo come input di un classificatore insieme alla colonna del target per apprendere il modello di Machine Learning. Lo scopo è trovare una correlazione fra il testo di una recensione e il suo sentiment.
É possibile utilizzare lo stesso dizionario per svariati modelli, ampiamente configurabili. Ai fini dell'esperimento ne sono stati provati tre: Random Forest, Naive Bayes e SVM. Vengono quindi presentate le performance ottenute da ciascuno, considerando una separazione fra training e test set del 70-30% con rimescolamento.

Verranno riportate per ciascun modello tutte le metriche valutate. Mentre la matrice di confusione, così come ROC e AUROC sono ottenute da una singola esecuzione, le metriche di accuracy, recall, precision e F1 sono invece ricavate attraverso l'esecuzione di una 10-fold cross validation.

5.3.1 Random Forest

Random Forest è dei tre il modello con le peggiori performance, qui riportate. Bisongna considerare che rispetto agli altri modelli è anche decisamente più lento nell'apprendimento (nell'ordine di un paio di minuti, contro una decina di secondi per NB e SVM).

```
RandomForestClassifier(
   bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features='auto', max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=5, warm_start=False)
```

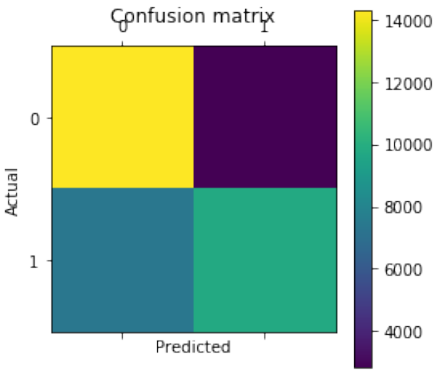


Figura 13: Random Forest - Matrice di confusione

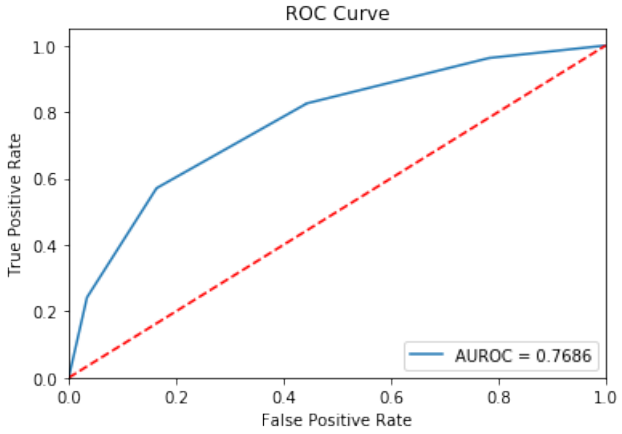


Figura 14: Random Forest - ROC e AUC

```
ACCURACY     10-FOLD CROSS VALIDATION:     0.7327 (std dev 0.005981)
PRECISION    10-FOLD CROSS VALIDATION:     0.7348 (std dev 0.009141)
RECALL       10-FOLD CROSS VALIDATION:     0.7348 (std dev 0.008359)
F1           10-FOLD CROSS VALIDATION:     0.7279 (std dev 0.003274)
```

5.3.2 Naive Bayes

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

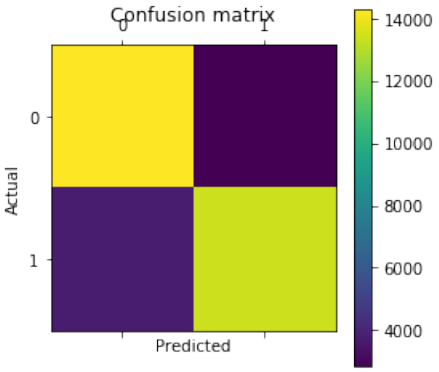


Figura 15: Naive Bayes - Matrice di confusione

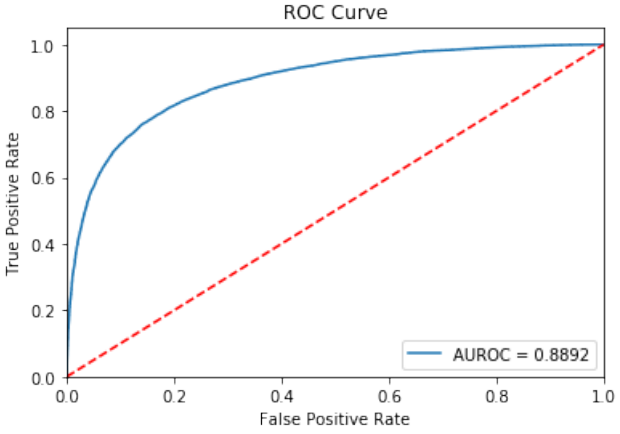


Figura 16: Naive Bayes - ROC e AUC

```
ACCURACY     10-FOLD CROSS VALIDATION:     0.8015 (std dev 0.007751)
PRECISION    10-FOLD CROSS VALIDATION:     0.8116 (std dev 0.014807)
RECALL       10-FOLD CROSS VALIDATION:     0.7858 (std dev 0.007262)
F1           10-FOLD CROSS VALIDATION:     0.7984 (std dev 0.006062)
```

5.3.3 Support Vector Machines

```
SGDClassifier(
   alpha=0.0001, average=False, class_weight=None,
   early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True,
   l1_ratio=0.15, learning_rate='optimal', loss='log', max_iter=1000,
   n_iter_no_change=5, n_jobs=None, penalty='l2', power_t=0.5,
   random_state=None, shuffle=True, tol=0.001,
   validation_fraction=0.1, verbose=0, warm_start=False)
```

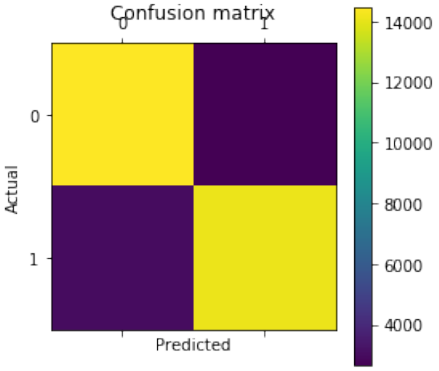


Figura 17: SVM - Matrice di confusione

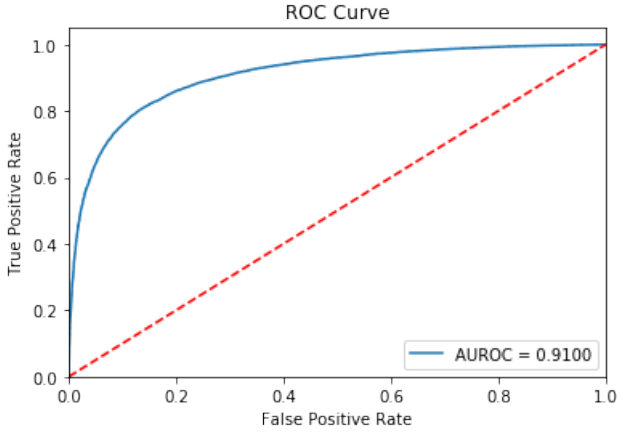


Figura 18: SVM - ROC e AUC

```
ACCURACY     10-FOLD CROSS VALIDATION:     0.8290 (std dev 0.005217)
PRECISION    10-FOLD CROSS VALIDATION:     0.8371 (std dev 0.009706)
RECALL       10-FOLD CROSS VALIDATION:     0.8176 (std dev 0.008412)
F1           10-FOLD CROSS VALIDATION:     0.8273 (std dev 0.004608)
```

5.3.4 Scelta del modello

Come mostrato, Naive Bayes e SVM ottengono performance molto simili. Il primo è più rapido nell'apprendimento, ma il secondo ottiene un paio di punti percentuali in più su tutte le metriche. Per questo motivo, si è scelto di utilizzare SVM per le predizioni future.

5.4 Pipeline

Scelta la modalità di elaborazione del testo e il classificatore più adeguato per il dominio di riferimento, `scikit-learn` consente di effettuare predizioni sui nuovi input. Chiaramente non è necessario che vengano effettuate ogni volta le operazioni preliminari di apprendimento sull'intero dataset, ma è sufficiente sfruttare il modello appreso a cui fornire il nuovo input, adeguatamente pre-processato. Per agevolare il processo, la classe `Pipeline` fornisce al programmatore la possibilità di definire una sequenza di operazioni necessaria sia per l'apprendimento che per la predizione su nuovi input.

É sufficiente quindi salvare il modello su un file persistente per poterlo riutilizzare successivamente. Questa è anche la tecnica adottata per il funzionamento della Web demo che verrà presentata nel capitolo 8. Per completezza, si riporta qui anche il codice della pipeline scelta per effettuare i nostri task di sentiment prediction.

```
text_clf = Pipeline([
     ('vect', TfidfVectorizer(
                  use_idf = True,
                  strip_accents = 'ascii',
                  stop_words = stopset,
                  lowercase = True)),
     ('tfidf', TfidfTransformer()),
     ('clf', SGDClassifier(loss='log')),
])
```

6 Aspect Based Sentiment Analysis

Se nel capitolo precedente è stata presentata una forma di sentiment analysis basata su un modello di apprendimento supervisionato, non è sempre ovvio che nel dataset da analizzare sia presente il target su cui effettuare la classificazione. Inoltre, l'avanzamento della ricerca nell'ambito del Natural Language Processing (NLP) ha permesso ai ricercatori di addestrare svariati modelli negli anni e quindi apprendere e pubblicare dei vocabolari standard che permettono, sia attraverso un'analisi morfologica che semantica, di valutare il sentiment di un insieme di parole.

In questo contesto nasce l'aspect based sentiment analysis (ABSA) il cui scopo è l'estrazione degli **aspetti** da una base testuale, cioè dei concetti descritti al suo interno, e la loro conseguente classificazione come positivamente o negativamente valutati dal testo attraverso l'utilizzo di vocabolari consolidati nel tempo. Ciò consente a chi analizza un corpus di ottenere una visione d'insieme non solo relativa all'opinione generica dei propri utenti, ma anche riguardo specifici aspetti (caratteristiche del prodotto, ad esempio). Questo permette alle aziende di migliorare ulteriormente il proprio servizio e porsi ovviamente su una più alta fascia di mercato.

6.1 Strumenti

L'aspect based sentiment analysis è un argomento meno approfondito allo stato dell'arte attuale rispetto agli argomenti precedenti trattati. Sebbene esistano numerosi articoli che ne parlano e anche alcuni servizi a pagamento che consentono di fare ABSA, è molto più complicato - se non impossibile - trovare delle librerie open source che permettano di eseguire rapidamente l'intera pipeline necessaria a questo tipo di analisi, come invece è possibile fare per il sentiment prediction supervisionato.

In funzione di queste considerazioni, si è deciso di valutare l'opportunità di riprendere un progetto open source scritto in Python e già avviato su GitHub, per ri-adattarlo alle esigenze di questo studio (9). Dopo qualche esperimento, l'utilizzo di tale progetto si è rivelato la scelta adatta a soddisfare i requisiti richiesti per implementare correttamente aspect based sentiment analysis.

6.2 Elaborazione del testo

Nonostante sia stato accennato che questa tecnica di analisi sia fondamentalmente diversa dalla precedente, bisogna notare che la trattazione preliminare dei testi avviene in modo totalmente analogo. I principi spiegati nel capitolo 4.2 rimangono validi e devono solamente essere riadattati per adeguarsi alle fasi successive.

Nello specifico, le operazioni che si è scelto di effettuare sull'input sono le seguenti:

- rimozione stop word
- suddivisione in frasi
- rimozione delle domande: poco significative o sarcastiche
- POS tagging (approfondito nella sezione successiva)
- lemmatization, con il supporto del precedente POS tagging

6.2.1 POS tagging

Acronimo di *part-of-speech tagging*, consiste nell'assegnazione di un tag per ciascuna parola del testo, al fine di identificare la categoria grammaticale a cui appartiene. Linguisticamente parlando, il termine **parte del discorso** definisce le categorie di parole atte a formare le frasi di una lingua, definite dalla disciplina chiamata **morfologia**. Nella lingua inglese sono distinte in: nomi, aggettivi, verbi, proposizioni, avverbi, pronomi, congiunzioni e interiezioni.

Il processo di POS tagging, quindi, analizza una frase e si occupa di assegnare il ruolo di nome, aggettivo, verbo, etc. ad ogni parola che la compone. Tale operazione è svolta nel codice attraverso l'utilizzo della libreria `NLTK`, precedentemente utilizzata anche per le altre fasi dell'elaborazione testuale.

6.3 Estrazione degli aspetti

Lo step successivo al tagging consiste nell'identificazione degli aspetti da tenere in considerazione per l'analisi. Per comprendere come avviene tale estrazione è semplicemente necessario fornire la definizione di aspetto secondo l'implementazione adottata: un aspetto è la **congiunzione di uno o più sostantivi contigui** derivanti dal POS tagging. Appare dunque chiaro che questa fase non fa altro che iterare i tag e associare gli aspect di conseguenza.

Una peculiarità di questa fase è il parametro di *threshold*, configurabile a piacimento in base all'utilizzo. Spesso infatti non risulta conveniente considerare tutti gli aspetti che vengono estratti ma solamente quelli che occorrono un determinato numero di volte. Questo fa si che eventuali errori di valutazione o battitura, per esempio, vengano ignorati piuttosto che essere considerati come aspetti nonostante siano di fatto errati o di poca rilevanza. Questa tipologia di filtraggio permette inoltre di estrarre, per dataset molto grandi, non tutti gli aspetti trattati dagli utenti ma solo quelli più ricorrenti e conseguentemente anche più importanti.

6.4 Identificazione del sentiment

Avendo a disposizione gli aspetti e le frasi che compongono il corpus è necessario finalmente stimare il sentiment associato a ciascun aspetto. Attraverso la libreria `TextBlob` (20) è possibile ottenere la **polarità** di una parola, cioè un punteggio che rappresenta il sentiment della parola su scala numerica, in grado di esprimere il livello di negatività, neutralità o positività del termine stesso.

Iterando sugli aspetti precedentemente estratti ed identificando nel dataset le frasi che li contengono, è possibile utilizzare la libreria per ottenerne la polarità. Si è scelto inoltre di salvare per ciascun aspect anche la lista di aggettivi che lo descrivono. Data la classe associata ad un aspect, viene per completezza infine computata anche la percentuale di frasi che hanno portato all'identificazione di tale classe, sul totale delle frasi inerenti quel determinato aspetto; ciò è utile per valutare lo sbilanciamento fra diverse classi che è possibile constatare per il medesimo argomento.

6.5 Risultati

A seguito dell'esecuzione degli step descritti, è stato possibile trarre alcune conclusioni riguardo gli aspetti identificati dall'algoritmo.

I dati estrapolati dal dataset sono agevolmente navigabili mediante l'interfaccia web definita nel capitolo 8.2. In linea generale è possibile asserire che l'algoritmo sia in grado di identificare correttamente la maggior parte degli aspetti con il relativo sentiment. Talvolta tuttavia, sono identificati come aspetti delle parole non propriamente congrue rispetto quanto ci si aspetti dalla frase. Vengono per esempio selezionati alcun aggettivi erroneamente classificati come sostantivi, anche a causa dell'elevato numero di occorrenze. Sulle singole frasi poi è possibile osservare che non sempre la divisione in frasi per punteggiatura funziona correttamente, andando così a ignorare qualche aspetto che potrebbe risultare fondamentale.

Data la complessità del task, in ogni caso la presenza di qualche errore rimane accettabile. Un miglioramento potrebbe essere introdotto con l'introduzione di uno passaggio per identificare la dipendenza fra i token all'interno della frase, come spiegato nel paper *The Representation of Sentence Meaning* (16).

L'utilizzo di un sistema di questo genere quindi sembrerebbe essere maggiormente utile se applicato a un grosso insieme di recensioni dalle quali estrarre gli aspetti fondamentali. Un impiego mirato sulla singola recensione risulta poco accurato nel complesso e quindi poco utile. Un'azienda dovrebbe dunque effettuare queste analisi su tutti i dati a disposizione, per identificare gli aspetti critici e i punti di forza dei servizi che sta offrendo, al fine di correggere eventuali problematiche e fornire sempre il meglio per il cliente.

7 Collaborative Filtering

É stato mostrato come attraverso l'analisi del linguaggio naturale sia possibile ricavare l'opinione di un utente nei confronti di un prodotto o di svariati aspetti legati al business che si sta analizzando. Attraverso l'analisi del sentiment di un gruppo di utenti, è inoltre possibile costruire un modello di raccomandazione su una collezione di prodotti. Ciò è implementabile attraverso due possibili approcci:

- **content-based**: sfruttando i metadati di un insieme di prodotti, come ad esempio il genere o la categoria di prezzo, è possibile identificare degli elementi chiave in grado di rappresentare i gusti di uno o più utenti, a cui conseguentemente suggerire prodotti che rispecchino tali preferenze
- **collaborative-filtering**: senza considerare alcun tipo di metadato, è possibile inferire dallo storico degli acquisti e delle preferenze espresse dagli utenti una similarità fra prodotti o utenti stessi, secondo l'assunzione per cui, ad esempio, due persone che hanno valutato equamente un gruppo di prodotti abbiano effettivamente dei gusti simili; è possibile quindi sfruttare questa considerazione per suggerire al secondo utente anche altri acquisti effettuati dal primo

Poiché, come precedentemente accennato, il dataset utilizzato per lo studio è privo di metadati, si è scelto di provare a implementare un sistema di raccomandazione basato su collaborative filtering. Quest'ultimo può rappresentare un netto vantaggio per la piattaforma e-commerce, in funzione della necessità di fidalizzare i propri clienti e spingerli ad effettuare nuovi acquisti.

7.1 Funzionamento

Il concetto di collaborative filtering è utile nel contesto in cui, data una lista di persone e articoli, si conoscono le valutazioni assegnate da ciascun utente nei confronti di alcuni articoli sul totale.

Una conoscenza di questo tipo, rappresentata solitamente attraverso la matrice in figura 19, è utile per effettuare predizioni in merito alle valutazioni mancanti sulla base degli utenti con preferenze simili.

Ciò è reso possibile attraverso l'applicazione di calcoli matriciali in grado di estrarre dalla matrice opportuni descrittori per ciascun utente e prodotto al fine di effettuare una successiva comparazione.

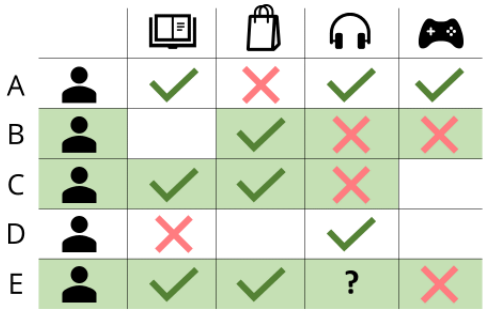


Figura 19: Matrice di utenti e prodotti con valutazioni personali

7.1.1 Embedding

Come appena accennato, la prima fase di realizzazione di un collaborative filtering prevede l'impiego di una modellazione dei dati che sia in grado di descrivere e successivamente comparare efficacemente le componenti del dominio. L'estrazione di tali descrittori dalla matrice in figura 19 è detta embedding, e può essere svolta con molteplici tecniche. La più utilizzata consiste nel proiettare ciascun prodotto e utente in uno spazio vettoriale, andando quindi a rappresentare ciascuno degli elementi che si vuole descrivere tramite un vettore.

L'operazione effettuata sulla matrice è chiamata **fattorizzazione**, ed avviene analizzando singolarmente le righe e le colonne della matrice. Al termine della fattorizzazione si otterranno tanti embedding quanti sono gli utenti e i prodotti, ognuno dei quali rappresentato da un vettore con un arbitrario numero di elementi scelto a priori. Ciascun elemento dei vettori di embedding rappresenta di fatto una caratteristica dell'utente o del prodotto, inferita dalla matrice delle valutazioni, senza tuttavia che sia dato modo sapere quale sia il suo vero significato semantico.

Il calcolo dei vettori di embedding non è univoco, pertanto quest'ultimi sono sottoposti a cicli di aggiornamento durante la fase di apprendimento del modello di collaborative filtering con lo scopo di ottimizzare il risultato delle predizioni.

7.1.2 Bias

Oltre al vettore di embedding, per ogni elemento del dominio viene calcolato anche un ulteriore parametro detto bias, che ha l'obiettivo di normalizzare utenti e prodotti rispetto all'insieme delle valutazioni ricevute. Tale parametro è ad esempio in grado di trovare un compromesso per la descrizione di utenti particolarmente critici o prodotti molto popolari. Il bias per ciascun prodotto e utente ha chiaramente un'influenza sui risultati delle predizioni, e anch'esso come l'embedding è soggetto ad un processo di regressione svolto durante l'apprendimento.

7.1.3 Predizione

Premesso che embedding e bias sono appresi attraverso l'applicazione iterativa di specifici modelli di machine learning, è importante considerare quale sia la metodologia da utilizzare per effettuare il confronto fra utenti e prodotti. Il metodo scelto determina anche la modalità di esecuzione della predizione, che dev'essere svolta per ogni possibile coppia di utenti e prodotti.

La fase di predizione è pertanto dipendente dal concetto di similarità fra elementi, che può essere calcolata con diverse tecniche. Fra le più utilizzate si trovano ad esempio l'applicazione del **prodotto scalare**, la **distanza euclidea** o la **cosine**

similarity. Considerando ad esempio una similarità basata su prodotto scalare, la predizione del `rating` per l'utente u e il prodotto p è calcolata come segue:

$$rating_{up} = dot\_product(user, product) + bias_u + bias_p$$

Il risultato produce un numero decimale compreso fra 1 e 5, che rappresenta il rating predetto. Applicando la formula ad ogni coppia di utenti e prodotti e verificando quando tali predizioni si discostino dai valori reali è possibile calcolare la *loss* al fine di valutare il modello, inteso come combinazione di embedding e bias. L'apprendimento avviene quindi tramite l'applicazione di algoritmi di ottimizzazione con lo scopo di ridurre al minimo l'errore generale commesso per le predizioni.

7.2 Implementazione

Per sperimentare l'utilizzo di un modello di collaborative filtering sul dataset considerato, si è quindi scelto di provare la libreria Python `fastai` (7), pensata per l'applicazione di algoritmi di deep learning tramite reti neurali e l'utilizzo della *GPU* per ridurre i tempi di computazione.

Tale libreria fornisce un pacchetto `collab` appositamente pensato per implementare facilmente un modello di collaborative filtering. Attraverso la costruzione di una struttura dati ottimizzata, la libreria è in grado di apprendere gli embedding attraverso la tecnica del prodotto scalare o l'utilizzo di una rete neurale. Appresi i parametri del modello è quindi possibile svolgere predizioni fornendo gli identificativi di utenti e prodotti per cui si vuole tentare di predire il `rating`.

7.3 Risultati

Nonostante la semplicità di utilizzo di `fastai` per la creazione del modello, la libreria sembra essere ancora in fase di sviluppo e manca di funzionalità avanzate per la visualizzazione di molte metriche di valutazione. Pertanto, non è particolarmente facile ottenere metriche aggiuntive rispetto alla *loss* fornita dalla libreria in seguito all'apprendimento con politica 1-cycle (1).

A scopo puramente esemplificativo, viene innanzitutto mostrato sul piano cartesiano in figura 20 il posizionamento degli embedding relativi ad alcuni prodotti, per comprendere meglio cosa significhi rappresentare un'entità in uno spazio vettoriale. Prodotti spazialmente vicini dovrebbero essere classificati dal modello di collaborative filtering come fossero prodotti simili.

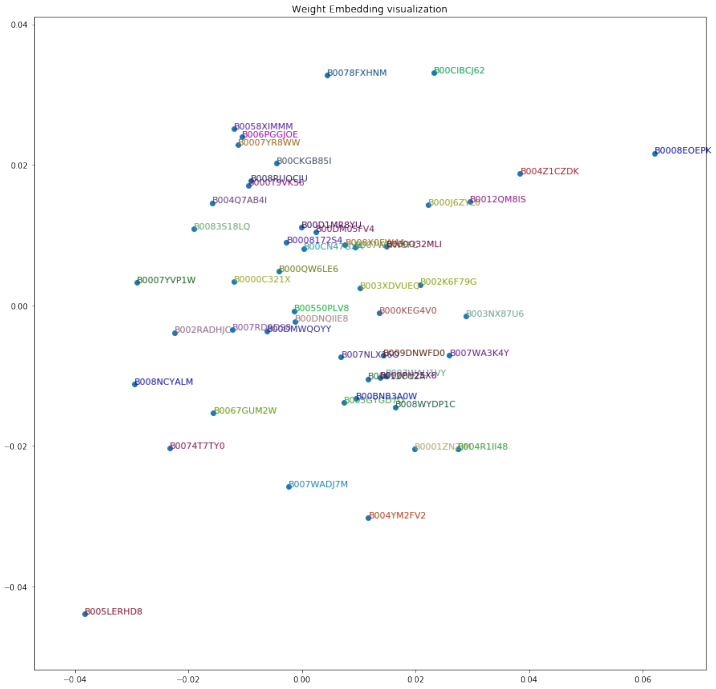


Figura 20: Embedding di alcuni prodotti

7.3.1 Performance

Per quanto riguarda invece la parte di valutazione delle performance, i valori rilevati per il modello di apprendimento sono riportati nella tabella 4. Questi si riferiscono alla fase di apprendimento svolta su 5 cicli con il metodo del prodotto scalare per il calcolo della similarità. Gli embedding considerati sono di 20 elementi ciascuno e la divisione fra training e test set è di 70-30%.

La colonna `train_loss` mostra la *loss* per il training set mentre la colonna `valid_loss` corrisponde a quella del test set, entrambe calcolate come errore quadratico medio (MSE). La figura 21 mostra i medesimi dati in formato grafico.

Tabella 4: Collaborative filtering - MSE, prodotto scalare

epoch	train_loss	valid_loss	time
0	2.717241	2.697971	00:36
1	2.539666	2.543803	00:37
2	2.429567	2.443485	00:36
3	2.344157	2.401317	00:36
4	2.330287	2.394638	00:36

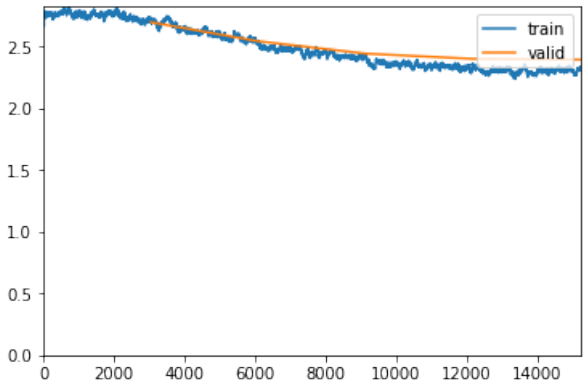


Figura 21: Collaborative filtering - MSE, prodotto scalare

É possibile notare che la *loss* sia coerente fra training e test set, pertanto si potrebbe dedurre che l'apprendimento sia svolto correttamente. Tuttavia, i corrispondenti valori discendono molto gradualmente fra un ciclo e l'altro, come se il modello non riuscisse a migliorarsi molto dopo la prima epoca. Per tentare di migliorare l'apprendimento si sono provate svariate combinazioni di parametri, sia relativamente al numero di elementi costituenti gli embedding e sia per cercare di ottimizzare la regressione, variando i valori di *learning rate* e *weight decay*. Tutti i tentativi hanno prodotto risultati del tutto paragonabili e si è evinto che lo scarto quadratico medio convergesse a 2.07 dopo il quindicesimo ciclo.

Si fa inoltre notare che, a differenza di quanto operato per la parte di sentiment analysis, i modelli di collaborative filtering non ottengono particolari miglioramenti lavorando su problemi binari, pertanto si è deciso di addestrare il modello sul problema multiclasse. Mostrando graficamente le predizioni effettuate su un campione del dataset a seguito del training, si è scoperto tuttavia un comportamento molto strano; come mostrato in figura 22, si è evinto che il modello classificasse il `rating` intorno al valore tre per qualsiasi coppia di utente e prodotto considerata.

Si è quindi scelto di effettuare un'operazione di downsampling sul dataset per tentare di bilanciare la classe target, ma la situazione si è quindi rivelata addirittura peggiore: la *loss* calcolata era minore, ma le predizioni giacevano tutte in un intorno ancor più ristretto del valore tre.

Per risalire alla causa del problema si è pensato che l'errore potesse risidere nelle modalità di apprendimento del modello, quindi si è deciso di testare come controprova il medesimo codice sorgente per l'implementazione di collaborative filtering su un dataset spesso utilizzato come esempio: quello di Movielens, contenente un totale di 100.000 recensioni di film. Il risultato, mostrato in figura 23, presenta una distri-

buzione delle predizioni nettamente migliore, sull'intera scala di valori ammissibili, riuscendo pertanto a dimostrare la sensatezza del modello implementato.

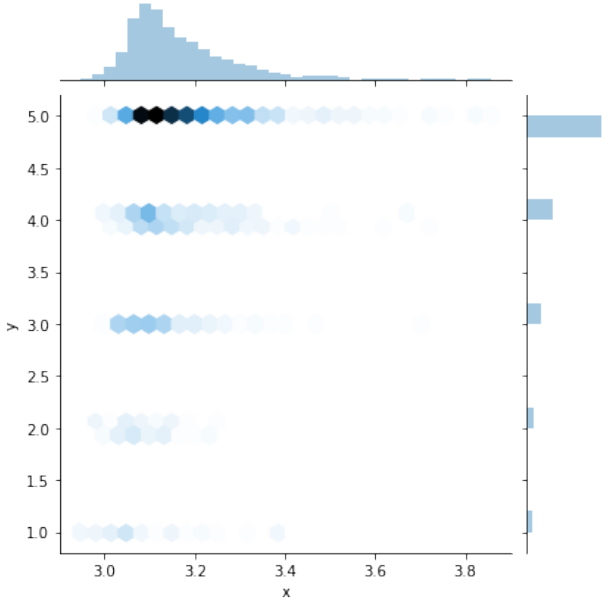


Figura 22: Predizioni del modello, tutte molto vicine a 3

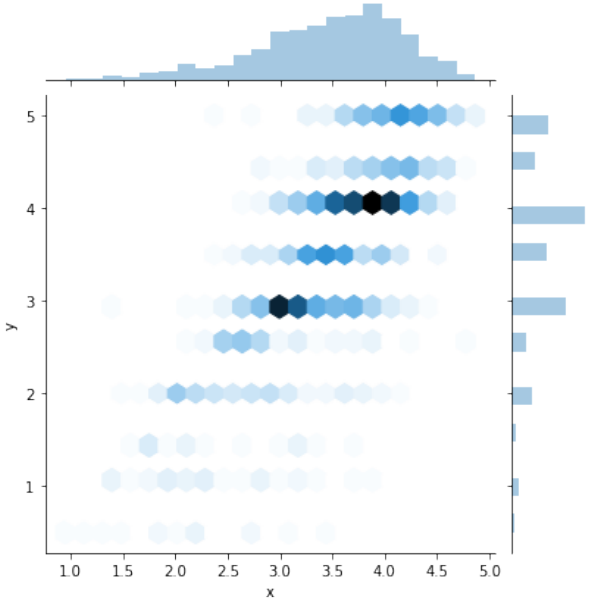


Figura 23: Predizioni sul dataset di Movielens, ben distribute

7.3.2 Coefficiente topologico

In funzione di queste considerazioni si è pensato che il problema potesse risiedere nel dataset, e non nel modello. Si è quindi deciso di effettuare un'analisi di comparazione fra il dataset di Amazon e quello di Movielens, sfruttando la rete creata nel capitolo 3. Considerata la distribuzione delle predizioni intorno a tre, che rappresenta anche il valore medio della variabile `rating`, si è pensato che il problema potesse essere rappresentato la mancanza nel dataset di un sufficiente numero di utenti o prodotti che condividessero recensioni. Considerate ad esempio tutte le possibili coppie di utenti, è fondamentale che alcune di queste abbiano in comune un certo numero di recensioni sui medesimi prodotti, affinché collaborative filtering sia in grado di considerare due utenti simili e pertanto suggerire ad alcuni di essi nuovi prodotti da acquistare. Se questa assunzione viene a mancare all'interno del dataset, allora tale modello di raccomandazione non può algoritmicamente funzionare.

Per analizzare la correlazione effettivamente presente fra utenti nel dataset si è scelto di valutare sulla rete una metrica che prende il nome di **coefficiente topologico** (5). Tale parametro è calcolato per ogni nodo n con la formula che segue, e in un dominio [0, 1] esprime una misura relativa alla quantità di vicini che ciascun nodo n condivide con gli altri nodi del grafo.

$$T_n = \frac{avg(J(n,m))}{k_n}$$

Mentre k_n è il grado del nodo (il numero dei suoi vicini), $J(n,m)$ rappresenta il numero di vicini in comune fra i nodi n ed m, ed è calcolato su tutti i nodi m che condividono almeno un vicino con n.

Nel caso in esame è pertanto utile per considerare il numero di recensioni ai medesimi prodotti che condividono diversi utenti; in modo analogo, il ragionamento applicato ad un prodotto determina se quest'ultimo condivida un buon numero di recensori con altri articoli nel dataset. Se il coefficiente topologico di un nodo è alto, significa che tale utente o prodotto è un buon candidato per l'algoritmo di collaborative filtering poiché sarà possibile effettuare sul nodo delle ottime predizioni; viceversa, sarebbe impossibile poter effettuare una qualsiasi predizione corretta e pertanto fornire un suggerimento valido.

Analizzando la distribuzione dei coefficienti topologici sul grafo ottenuto dal dataset delle recensioni Amazon (figura 5), si evince quindi il motivo per cui collaborative filtering non funziona su tale insieme di recensioni. Come si osserva in figura 24, la maggior parte dei nodi della rete (utenti o prodotti che siano) possiede un coefficiente inferiore a 0.25. Ciò significa che quasi tutti i nodi non condividono i propri vicini

con un buon numero di altri nodi nel grafo; infatti, solamente 11 elementi sul totale presentano un coefficiente topologico superiore a 0.40, ed è inoltre possibile notare che vi è un singolo nodo che condivide con altri tutti i propri vicini.

A dimostrazione del fatto che il coefficiente topologico sia fondamentale affinché collaborative filtering funzioni, si mostra in figura 25 la distribuzione del medesimo coefficiente sul grafo relativo al dataset di Movielens, che si è evinto performare meglio (come mostrato già in figura 23). Come si può facilmente notare dal grafico, in questo caso i coefficienti sono mediamente ben più alti, con valori che partono da 0.65 e subiscono un decremento molto graduale. É pertanto chiara la motivazione che porta collaborative filtering ad ottenere risultati accettabili su quest'ultimo dataset, con una *loss* (MSE) di 0.73 al quinto ciclo, comparata a quella superiore a 2 del dataset di Amazon.

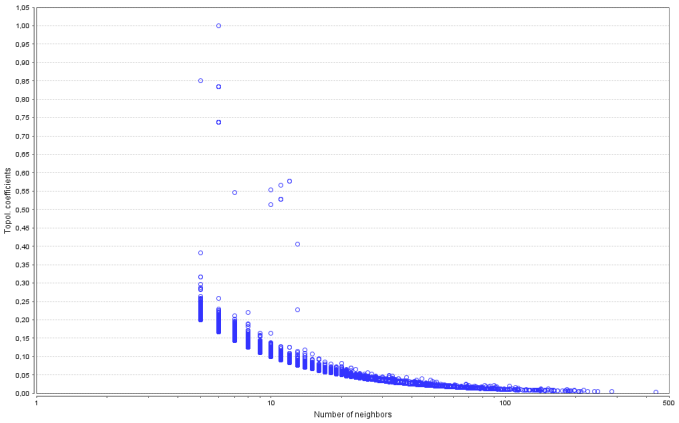


Figura 24: Coefficiente topologico dei nodi sul grafo di Amazon

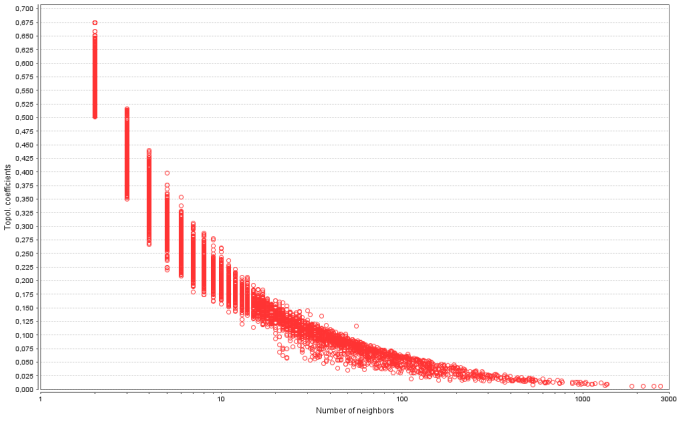


Figura 25: Coefficiente topologico dei nodi sul grafo di Movielens

7.4 Il caso Spotify

Come si è appena dimostrato, in alcuni casi collaborative filtering non è sufficiente a costituire un buon metodo di raccomandazione. Questo può accadere qualora all'interno del dataset non vi siano utenti o prodotti con recensioni in comune o, ancora più frequentemente, nel momento in cui all'interno del catalogo di prodotti si introducono nuovi articoli, pertanto privi di recensioni. In questi caso, tale approccio di raccomandazione non può matematicamente suggerire i nuovi arrivati; al contrario potrebbe essere causa dell'isolamento dei suddetti prodotti, qualora gli utenti continuassero a ricevere suggerimenti sui soli elementi già ampiamente recensiti.

Per questo motivo, un sistema di raccomandazione non dovrebbe essere implementato con il solo collaborative filtering, ma è bisognoso di alcune contromisure che impediscano l'insorgere di situazioni spiacevoli. Agire sul piano pubblicitario potrebbe essere la mossa più intuitiva per risolvere il problema dei nuovi prodotti, ma oltre ad essere lenta e costosa non ha alcun effetto nei casi in cui - come nel dataset considerato per questo studio - non vi siano sufficienti recensioni per considerare simili un gruppo di utenti o prodotti. Alla luce di queste considerazioni, è fortemente incoraggiata l'integrazione di diversi modelli per l'effettiva implementazione del sistema di raccomandazione.

Un portavoce di questa tecnica è sicuramente Spotify(19), famoso servizio per lo streaming di musica on-demand, che a collaborative filtering non affianca solamente un sistema di raccomandazione di tipo content-based, già citati nell'introduzione di questo capitolo, ma anche un modello basato sull'analisi dello spettro audio delle canzoni tramite deep learning (23). É così che nasce una delle feature che pongono la piattaforma di streaming su un piano molto competitivo rispetto ai concorrenti: **Discover Weekly**(18).

Attraverso l'elaborazione del linguaggio naturale nei testi delle canzoni, l'analisi sugli ascolti pregressi in combinazione con quella sui metadati (es: il genere musicale) e una sorta di regressione sulle componenti audio ricorrenti nelle canzoni che si preferiscono, Spotify è in grado di suggerire **ogni settimana a tutti i suoi utenti** una lista **personalizzata** di canzoni inedite che, con alta probabilità, saranno apprezzate dall'utente a cui vengono suggerite. Il sistema permette a chiunque di conoscere canzoni che altrimenti non si sarebbero scoperte, senza di fatto presentare particolari bias né per gli artisti emergenti o comunità di nicchia, né per quanto riguarda canzoni con la tendenza a divenire sempre più impropriamente popolari.

8 Web Demo

Al fine di dimostrare l'utilizzo della sentiment analysis in un applicativo software, viene qui illustrato lo sviluppo e il funzionamento di un'applicazione web che permette la predizione del sentiment di una recensione, sia sul piano globale che a livello dei singoli aspetti identificabili. Viene inoltre mostrata a scopo esemplificativo un'interfaccia per la visualizzazione degli aspetti più ricorrenti nel dataset.

8.1 Architettura

L'interfaccia web è stata sviluppata utilizzando l'architettura client-server, con la separazione tra front-end e back-end.

Per lo sviluppo del back-end si è scelto di utilizzare l'engine Javascript attraverso il progetto Node.js (13), in grado di agire come middleware tra il front-end e la fonte dei dati al fine di separare al meglio le logiche di manipolazione del dato stesso dalla sua presentazione. É inoltre incaricato di invocare gli script Python per le analisi descritte nei capitoli precedenti.

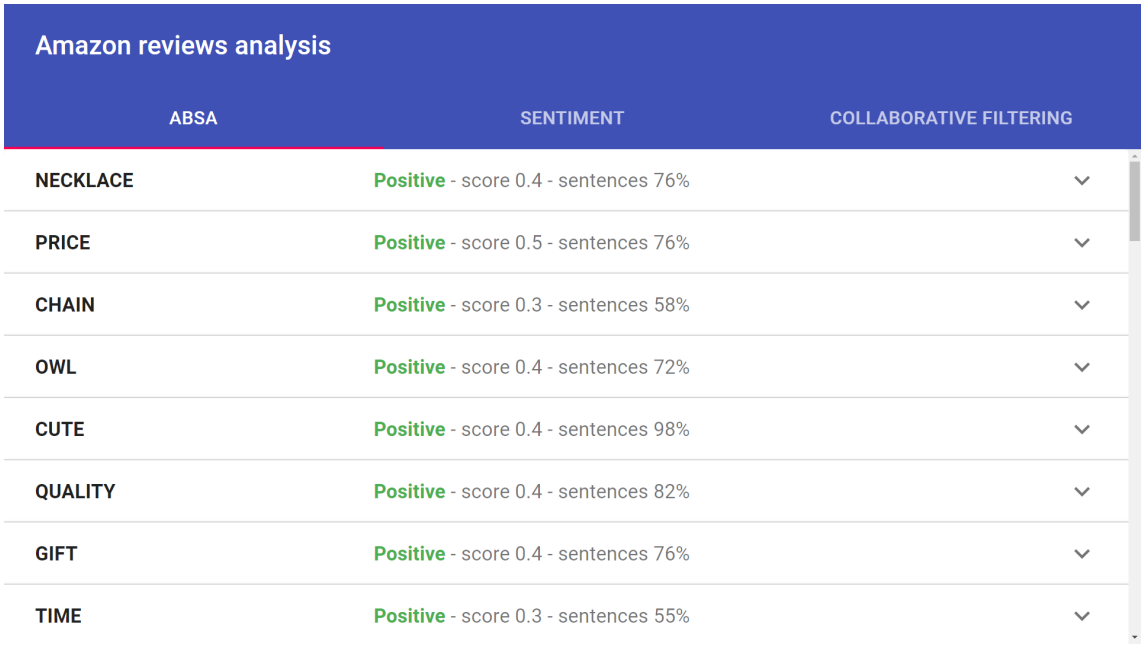
Il frontend è invece sviluppato utilizzando la libreria Javascript React.js (15) ed è dedicato all'interazione dell'utente con i modelli di sentiment analysis sviluppati tramite la visualizzazione dei risultati ottenuti con quest'ultimi.

8.2 Aspect Based Sentiment Analysis

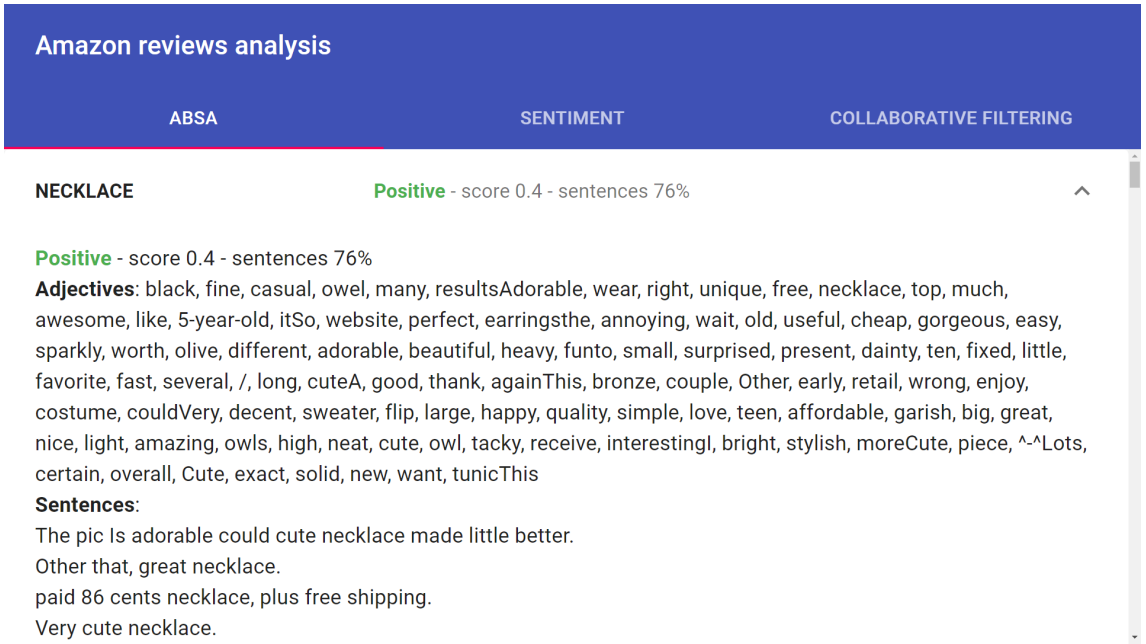
All'avvio dell'app tramite web browser viene presentata la schermata principale con due tab di selezione: *ABSA* e *Sentiment*.

Selezionando il primo dei due, compare la videata rappresentata in figura 26 (a) che contiene una lista di aspect, ognuno dei quali associato al sentiment predominante. Tale lista rappresenta le informazioni estratte dal dataset in riferimento al solo prodotto più recensito. Ciascuna riga mostra l'aspect sul lato sinistro, mentre nella parte destra vengono mostrate la polarità con il relativo score e la percentuale di frasi con tale polarità, in relazione a tutte quelle che lo riguardano.

Facendo click su una delle righe è possibile espandere i dettagli e visualizzare, per ogni classe (positiva, negativa o neutrale), le frasi e gli aggettivi ad essa associati, così come mostrato in figura (b).



(a) *Lista degli aspetti*



(b) *Dettaglio del singolo aspect*

Figura 26: Aspect based sentiment analysis UI

8.3 Sentiment Prediction

Tramite il secondo tab, posto nella parte superiore dello schermo, è possibile accedere ad una schermata che permette l'inserimento da parte dell'utente di una nuova recensione; di questa ne viene valutato sia il sentiment complessivo attraverso il modello di machine learning esplicato nel capitolo 5.4, sia estratta la lista degli aspect con associati i relativi sentiment.

La figura 27 mostra un'anteprima dell'interfaccia. É dunque possibile inserire un testo nell'apposita area di input sul lato sinistro e premere il pulsante sottostante per avviare l'analisi. Il risultato verrà mostrato sul lato destro dello schermo, con un'interfaccia analoga a quella mostrata precedentemente per la parte relativa all'aspect based sentiment analysis.

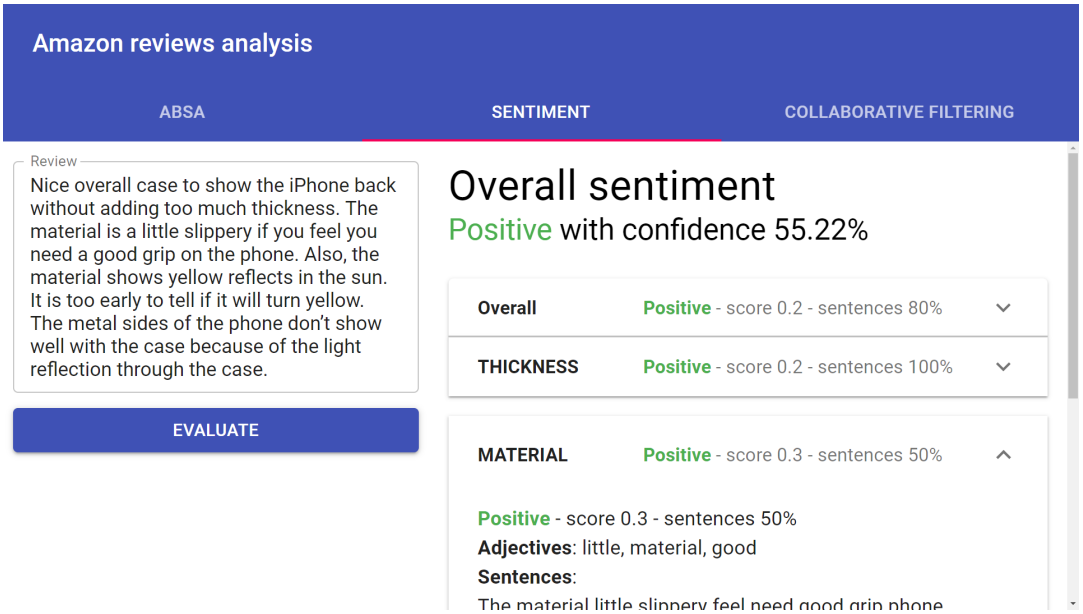


Figura 27: Sentiment prediction UI

9 Conclusioni

Lo studio è riuscito a dimostrare le innumerevoli possibilità di effettuare analisi sul dominio delle recensioni. Considerando sia l'elaborazione del linguaggio naturale che l'analisi sulla rete è possibile ottenere risultati molto importanti e anche piuttosto precisi sotto diversi aspetti, che possono sicuramente rappresentare un contributo fondamentale al miglioramento del business di un'azienda.

In particolare, l'analisi delle recensioni di Amazon può essere utile sia ai produttori che al portale di e-commerce stesso per migliorare i propri servizi o sapere come ampliare l'offerta commerciale, curando magari gli aspetti più carenti. Con assoluta certezza si potrebbe appurare che parte del successo di Amazon è stato acquisito anche grazie ad analisi di questo tipo.

I medesimi modelli possono essere facilmente riadattati a qualsiasi altro dominio preveda l'utilizzo di recensioni e sarebbe sicuramente interessante approfondire quali dei principali colossi tech odierni abbiano sfruttato metodologie di sentiment analysis e collaborative filtering per scalare le vette di mercato. Si è visto ad esempio nel capitolo 7.4 come Spotify sia in grado di fornire ai propri utenti un modello di raccomandazione della musica sfruttando svariati approcci: content e metadata analysis, collaborative filtering e deep learning sullo spettro audio delle canzoni.

Considerata la precisione dei risultati ottenuti si può dedurre che il metodo migliore per ottimizzare l'utilizzo dei dati raccolti sia quello di combinare molteplici tecniche di analisi e modelli di machine learning in grado di influenzarsi a vicenda, nell'ottica di ottenere risultati più completi, precisi e utili sia per gli utenti che per il mercato.

Riferimenti bibliografici

[1] 1cycle policy for learning. URL: https://sgugger.github.io/the-1cycle-policy.html.

[2] Amazon. URL: https://www.amazon.it/.

[3] Amazon reviews 5-core dataset source. URL: http://jmcauley.ucsd.edu/data/amazon/.

[4] Amazon reviews original dataset source. URL: https://s3.amazonaws.com/amazon-reviews-pds/readme.html.

[5] Coefficiente topologico di un grafo. URL: http://www.centiserver.org/?q1=centrality&q2=Topological_Coefficient.

[6] Cytoscape for network integration, analysis and visualization. URL: https://cytoscape.org/.

[7] Fastai, python library for deep learning. URL: https://docs.fast.ai.

[8] Google colab. URL: https://colab.research.google.com/.

[9] jonm01/absa: Aspect based sentiment analysis on amazon reviews. URL: https://github.com/jonm01/absa.

[10] Julian mcauley. URL: http://jmcauley.ucsd.edu/data/amazon/.

[11] Natural language toolkit. URL: https://www.nltk.org/.

[12] Networkx python package for networks management. URL: https://networkx.github.io/documentation/stable/.

[13] Node.js. URL: https://nodejs.org/it/.

[14] Pandas: Python data analysis library. URL: https://pandas.pydata.org/.

[15] React. URL: https://reactjs.org/.

[16] The representation of sentence meaning. URL: https://web.stanford.edu/~jurafsky/slp3/14.pdf.

[17] Scikit-learn, machine learning in python. URL: https://scikit-learn.org/stable/.

[18] Spotify discover weekly: la playlist creata apposta per te, ogni lunedì. URL: https://www.spotify.com/it/discoverweekly/.

[19] Spotify, la musica è per tutti. URL: https://www.spotify.com/it/.

[20] Textblob. URL: https://textblob.readthedocs.io/en/dev/.

[21] Wikipedia - degeneracy (graph theory). URL: https://en.wikipedia.org/wiki/Degeneracy_(graph_theory).

[22] Wordcloud. URL: https://amueller.github.io/word_cloud/.

[23] Sophia Ciocca. How does spotify know you so well? URL: https://medium.com/s/story/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe