Università di Pisa
Dipartimento di Informatica

Corso di Laurea Magistrale in
Data Science and Business Informatics

Progetto per il corso di
Text Analytics

Analisi del dataset “Amazon cell phones reviews”

A cura di:
Mario Bianchi, 616658
Marco Dalla Zanna, 603549

Anno accademico 2021/2022

1 Data Understanding

1.1 Introduzione

Il dataset *Amazon Cell Phones Reviews*[1] è stato ottenuto con la tecnica dello *scraping* dal sito di e-commerce *Amazon.com*. In particolare, il dataset è composto da informazioni relative all'acquisto di telefoni cellulari dal 2003 al 2019, ed è composto da due file:

- *items.csv*, composto da 720 osservazioni, in cui ogni osservazione descrive un telefono venduto sul sito;
- *reviews.csv*; composto da 67986 osservazioni, in cui ogni osservazione corrisponde ad una recensione relativa ad un acquisto, e comprende informazioni quali il testo della recensione, la valutazione data al prodotto e la data della recensione.

Dato lo scopo dell'analisi, ossia l'applicazione di tecniche di *text mining* finalizzate all'elaborazione del linguaggio naturale, lo studio si è focalizzato maggiormente sul secondo file.
Dopo aver analizzato il dataset nel complesso, ci si è concentrati sulla variabile *body*, contenente il testo della recensione, e sulla variabile *review_rating*, ossia la votazione data dall'acquirente al prodotto, espressa su una scala da 1 a 5 stelle.
In seguito alla preparazione del testo sono stati applicati prima algoritmi di apprendimento non supervisionato, ossia di clustering finalizzato a raggruppare recensioni simili, poi di supervisione debole, ossia di *weak labeling* di recensioni come positive o negative, e infine di apprendimento supervisionato finalizzato alla *Sentiment Analysis*.

Da subito sono state notate alcune criticità: in primo luogo, non è detto che una recensione sia coerente con la votazione espressa e soprattutto che tutti gli acquirenti adottino lo stesso metro di giudizio; in secondo luogo, non esiste una definizione di quale sia una votazione positiva in una scala da 1 a 5 stelle. Se il primo problema rientra nella natura dell'analisi, per il secondo si è dovuta prendere una decisione basata su una visione soggettiva, che necessariamente ha avuto un'influenza sull'analisi.

[1] https://www.kaggle.com/grikomsn/amazon-cell-phones-reviews?select=20191226-reviews.csv

1.2 Analisi del dataset e distribuzioni

Il file *items.csv* è composto da 720 osservazioni e 10 variabili. Le variabili descrivono i telefoni venduti sul sito *Amazon.com* e sono analizzate nella Tabella 1.1.

Nome	Descrizione	Tipologia	Valori
asin	Amazon Standard Identification Number	Categorico	{“B0000SX2UC”, …}
brand	Marca del telefono	Categorico	{“Motorola”, …}
title	Nome del prodotto	Categorico	{“Motorola I265 phone”, …}
url	Link al prodotto	Categorico	{“https://www.amazon.com/Dual-Band…”, …}
image	Link all’immagine del prodotto	Categorico	{“https://m.media-amazon.com/images, …”}
rating	Votazione media del prodotto	Continuo	{3, 2.7, 4, …}
reviewUrl	Link alla recensione	Categorico	{“https://www.amazon.com/product-reviews/B0000SX2UC”, …}
totalReviews	Numero di recensioni per prodotto	Discreto	{17, 7, 3, … }
price	Prezzo	Continuo	{49.95, 99.99, 149.99, …}
originalPrice	Prezzo originale	Continuo	{49.95, 99.99, 149.99, …}

Tabella 1.1: items.csv

Nella Figura 1.1 vengono rappresentate le distribuzioni di *review_rating* e *price*. È possibile notare che la maggior parte dei prodotti presenti una votazione media compresa tra il 3 e il 4,5 e che le fasce di prezzo più basse siano le più frequenti.

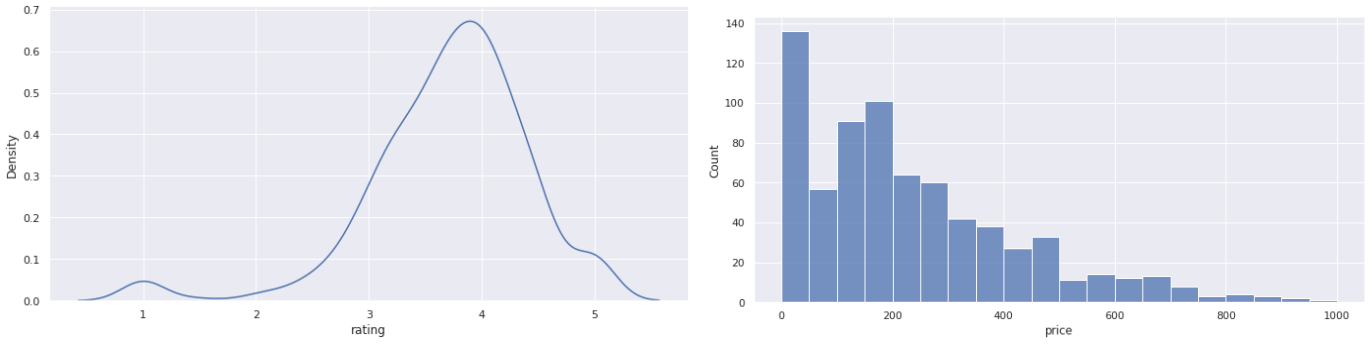

Figura 1.1: Distribuzioni di rating e price

Successivamente il dataset *items.csv* è stato unito al dataset contenuto in *reviews.csv* (il *merging* è stato eseguito sull'attributo *asin*). Alcuni attributi sono stati rinominati perché presentavano nomi simili ad altri. Il dataset ottenuto dall'unione dei due file, composto da 67986 osservazioni e 17 variabili, è descritto nella Tabella 1.2.

Nome	Descrizione	Tipologia	Valori
asin	Amazon Standard Identification Number	Categorico	{"B0000SX2UC", …}
name	Nome dell'autore della recensione	Categorico	{"Janet", "Luke Wyatt", …}
review_rating	Voto espresso dall'autore della recensione	Discreto	{1, 2 ,3, 4, 5}
date	Data della recensione	Attributo temporale	{"October 11, 2005", …}
verified	Indica se l'utente è verificato	Binario	{False, True}
review_title	Titolo della recensione	Categorico	{"Def not best, but not worst", …}
body	Testo della recensione	Categorico	{"I had the Samsung A600 for a while …", …}
helpfulVotes	Numero di persone che hanno ritenuto utile la recensione	Discreto	{1, 17, 33, …}
brand	Marca del telefono	Categorico	{"Motorola", …}
product_title	Nome del prodotto	Categorico	{"Motorola I265 phone", …}
url	Link al prodotto	Categorico	{"https://www.amazon.com/Dual-Band…", …}
image	Link all'immagine del prodotto	Categorico	{"https://m.media-amazon.com/images, …"}
product_rating	Votazione media del prodotto	Continuo	{3, 2.7, 4, …}
reviewUrl	Link alla recensione	Categorico	{"https://www.amazon.com/…", …}
totalReviews	Numero di recensioni	Discreto	{14, 27, 3, …}
price	Prezzo	Continuo	{49.95, 99.99, 149.99, …}
originalPrice	Prezzo originale	Continuo	{49.95, 99.99, 149.99, …}

Tabella 1.2: Descrizione del dataset unito

Le osservazioni sono registrate dal 2009 al 2019, e come rappresentato in Figura 1.2 il numero di recensioni è aumentato con il passare del tempo.
Per quanto riguarda invece i voti assegnati ai prodotti, ossia il numero di stelle indicato nella recensione del telefono, è possibile notare che gli estremi 1 e 5 sono assegnati più frequentemente. I risultati sono presentati in Figura 1.3.

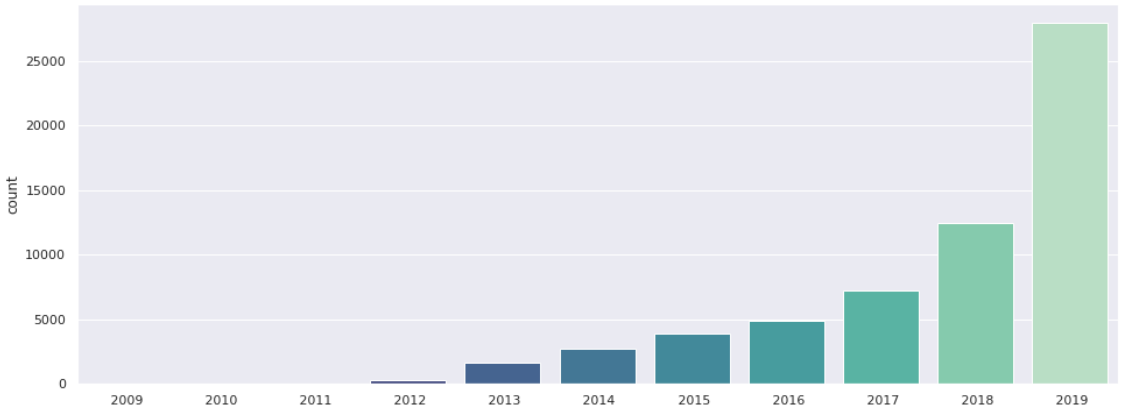

Figura 1.2: Numero di recensioni per anno

Figura 1.3: Distribuzione dei voti

Infine sono state analizzate le correlazioni tra gli attributi (Figura 1.4). Come prevedibile, la correlazione più forte (0.43) è presente tra le variabili *price* e *originalPrice*. Più interessante invece la seconda correlazione più forte (0.37), ossia quella presente tra *price* e *product_rating*, che indica l'esistenza di una correlazione positiva tra il prezzo di un telefono e il giudizio medio sullo stesso.

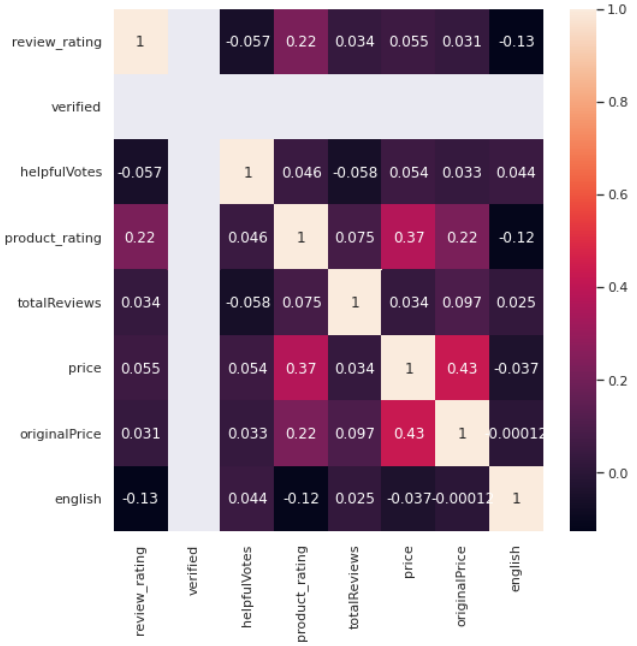

Figura 1.4: Correlazioni

1.3 Analisi dei missing values

Il dataframe ottenuto dall'unione di *items.csv* e *reviews.csv* contiene i missing values riassunti nella Tabella 1.3. Tutte le variabili che non presentavano valori mancanti sono state omesse. È stato notato che, nonostante non presentassero valori mancanti, le variabili *price* e *original_price* presentavano molti valori uguali a zero, rispettivamente 11755 e 53704. Nella fase di Data Preparation sono stati eliminati i valori mancanti.

name	review_title	body	helpfulVotes	brand
2	14	21	40771	200

Tabella 1.3: Missing values

2 Data Preparation

Il primo passo nella preparazione del dataset per gli algoritmi di classification e clustering è stato quello di eliminare gli attributi che non sarebbero stati usati. Questi sono: *asin, name, verified, brand, url, image, reviewUrl, totalReviews, price, originalPrice,* e *helpfulVotes.* Successivamente, sono state eliminate tutte le osservazioni che presentavano almeno un missing value. Il dataset è stato così ridotto a 9 attributi e 61198 osservazioni. Anche le recensioni in una lingua diversa dall'inglese sono state eliminate, riducendo ulteriormente le osservazioni a 53337.

La variabile target da determinare in fase di classificazione è quella che descrive la qualità della recensione scritta. Per gli algoritmi di classificazione multiclasse, è stata usata come attributo target *review_rating* che presenta 5 classi, come le stelle che si possono assegnare in una recensione Amazon. La distribuzione delle recensioni è illustrata nella Figura 1.3. Per gli algoritmi di classificazione binaria, abbiamo creato un nuovo attributo binario basato sui valori di *review_rating*. Le osservazioni di questo attributo, chiamato *pos_review* (*positive review*), assumono valore 1 quando il valore di *review_rating* è maggiore o uguale di 4, 0 altrimenti. In questo modo, la variabile binaria ha la distribuzione illustrata in Figura 2.1.

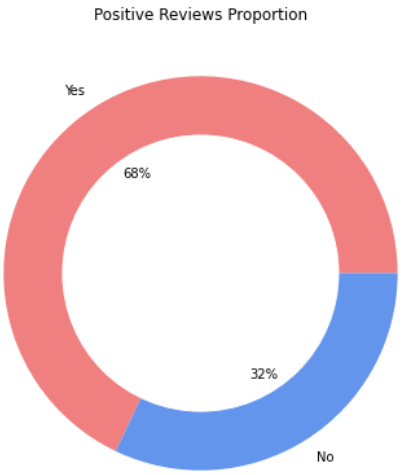

Figura 2.1: Distribuzione recensioni positive e negative

L'ultima cosa fatta prima di creare i tokens e n-grams per la classificazione è stata risolvere alcune criticità legate all'attributo *body,* il contenuto della recensione di un utente. Nella Tabella 2.1 sono mostrati i trigrammi più frequenti nell'attributo *body* (ossia prima della pulizia del testo) e nell'attributo *body_tok* (ottenuto con tokenizzazione e pulizia del testo). Alcune recensioni non sono state verificate da Amazon, e quindi nell'attributo *verified* presentano il valore 0. Il secondo problema è che è stato notato che alcune recensioni non erano in inglese. È stata fatta una ricerca della lingua della recensione utilizzando il metodo *LanguageDetector* del modulo *spacy_langdetect*. Questo metodo, data una stringa di testo, riesce a fare un'inferenza sulla lingua di appartenenza di quel testo. Abbiamo quindi creato due nuovi attributi: *language* e *english* che indicano rispettivamente la lingua individuata (variabile categorica) e se la lingua individuata è l'inglese o meno (variabile binaria). Una volta terminata quest'analisi, sono state eliminate le righe appartenenti a recensioni non verificate o in una lingua diversa dall'inglese.
Una volta risolte le criticità sopra menzionate, il dataset è stato processato svolgendo le seguenti operazioni:

- trasformazione del testo in minuscolo;
- rimozione di tutti i caratteri di punteggiatura;
- creazione dei token dalle stringhe testuali;
- creazione di bigrammi e trigrammi.

Un'altra trasformazione che è stata effettuata è la lemmatizzazione del testo, che è stata usata per le Support Vector Machines. Si rimanda al paragrafo 5.2 per maggiori informazioni.

L'ultima trasformazione è stata la creazione del test set e dei due training set per lo studio del *content drift*. Per fare ciò, il dataset è stato ordinato in base ai valori dell'attributo *year*. Le 20000 osservazioni più recenti hanno formato il test set, le 10000 più prossime al test set hanno formato un primo training set *(training set 2*) mentre le 10000 osservazioni antecedenti hanno formato il secondo training set (*training set 1*). Le osservazioni rimanenti, ossia quelle antecedenti al *training set 1*, hanno formato un altro set chiamato *remaining set,* il quale inizialmente non è stato utilizzato per lo studio del *content drift*.
Nella Tabella 2.2 è descritta la distribuzione degli anni nei quattro dataset.

Trigrammi più frequenti in *body*		Trigrammi più frequenti in *body_tok*	
'the', 'phone', 'is'	1367	'phone', 'works', 'great'	331
'this', 'phone', 'is'	1245	'great', 'battery', 'life'	315
'this', 'phone', 'for'	1109	'brand, 'new', 'phone'	270
'I', 'had', 'to'	1032	'like', 'brand', 'new'	264
'a', 'lot', 'of'	1030	'looks', 'brand', 'new'	262
'battery', 'life', 'is'	864	'battery', 'life', 'great'	246

Tabella 2.1: Trigrammi più frequenti pre e post pulizia del testo

	Training Set 1	Training Set 2	Test Set	Remaining Set
2019	-	3238	20000	-
2018	4425	6762	-	-
2017	5575	-	-	1075
2016	-	-	-	4336
2015	-	-	-	3490
2014	-	-	-	2485
2013	-	-	-	1594
2012	-	-	-	289
2011	-	-	-	50
2010	-	-	-	12
2009	-	-	-	6

Tabella 2.2: Distribuzione degli anni nei dataset

3 Clustering

3.1 K-Means

Nella fase di clustering si è cercato di scoprire se esistono dei raggruppamenti di recensioni per le quali è possibile notare delle differenze in termini di parole utilizzate, e in generale differenze nei valori di altri attributi differenti da quelli testuali. L'algoritmo utilizzato per questa ricerca è KMeans, il quale è stato applicato al dataset contenente tutte le recensioni, ottenuto dall'unione del *test set*, dei due *training set* e del *remaining set*.
L'algoritmo è stato implementato con valori del parametro k da 2 a 15. Le metriche di valutazione invece sono SSE e silhouette score. Il valore ottimale del parametro k è stato trovato tramite un'analisi di queste metriche, i cui valori sono illustrati nella Tabella 3.1. Per semplicità, abbiamo rimosso dalla tabella e dai grafici i valori di k maggiori di 7, dato la loro scarsa rilevanza. Dalla Figura 3.1 è possibile notare che 4 sia il valore ottimale per il clustering che è stato effettuato.

K	2	3	**4**	5	6	7
SSE	3246454.55	3246454.55	3150469.74	3132867.02	3120111.44	3110286.11
Silhouette score	0.511036	0.326184	0.182902	0.168777	0.175420	0.156867

Tabella 3.1: Valori di SSE e silhouette per k da 2 a 7

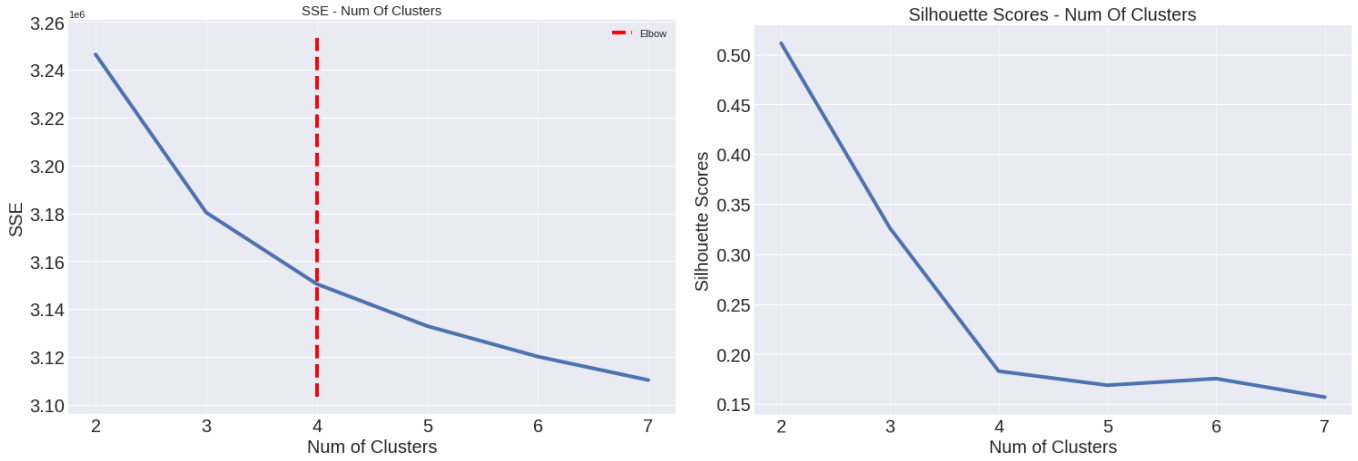

Figura 3.1: SSE e silhouette score di K-Means

I cluster ottenuti sono molto sbilanciati in termini di dimensione: infatti, il primo cluster (Cluster 0) in ordine di grandezza conta 37008 elementi, mentre il secondo (Cluster 1) non supera le 14000 osservazioni. La distribuzione della variabile target *pos_review,* al contrario, è equiparabile alla distribuzione presente nell'intero dataset (68%, 32%). Le Tabelle 3.1, 3.2 e 3.3 presentano delle statistiche relative ai cluster: mostrano rispettivamente la distribuzione della variabile target, la distribuzione della variabile *year* e i token più frequenti.

	Cluster 0	Cluster 1	Cluster 2	Cluster 3
Dimensione	37008	13610	2462	257
pos_review = 0	10562 (29%)	5504 (40%)	978 (40%)	72 (28%)
pos_review = 1	26446 (71%)	8106 (60%)	1484 (60%)	185 (72%)

Tabella 3.1: Distribuzione della variabile target nei cluster

	Cluster 0	Cluster 1	Cluster 2	Cluster 3
2019	16331	5811	1014	82
2018	7647	2958	529	53
2017	4650	1648	310	0
2016	3196	967	151	0
2015	2637	705	132	16
2014	1588	742	132	0
2013	811	622	144	17
2012	119	133	36	1
2011	24	15	10	1
2010	2	7	2	0
2009	3	2	1	0

Tabella 3.2: Distribuzione della variabile year nei cluster

Cluster 0		Cluster 1		Cluster 2		Cluster 3	
phone	15200	**phone**	32490	**phone**	16074	**phone**	4270
great	5888	**great**	4791	**screen**	2410	**screen**	770
good	4710	**battery**	4052	**battery**	2248	**one**	717
battery	4125	**one**	3701	**one**	2204	**like**	716
new	3790	**screen**	3596	**like**	2125	**use**	695
screen	3406	**new**	3518	**use**	2020	**camera**	639
phone	3339	**good**	3477	**get**	1958	**battery**	632
works	3268	**would**	339	**would**	1931	**get**	600
one	3171	**like**	3219	**good**	1809	**would**	490
great	3147	**use**	3093	**great**	1727	**phones**	487

Tabella 3.3: Token più frequenti nei cluster

4 Weak Labeling

Il weak labeling è una tecnica di supervisione debole finalizzata all'etichettatura automatica di grandi quantità di dati. Nella nostra analisi il weak labeling è stato utilizzato con una duplice funzione: ottenere una prima classificazione delle recensioni ed ottenere un metro di paragone per la variabile *pos_review*.
È stato utilizzato l'algoritmo *rule-based* VADER.

4.1 VADER

L'algoritmo di *weak labeling* VADER prende come input un testo e utilizza un *lexicon* per etichettare i testi come positivi, negativi o neutri. Più precisamente restituisce valori compresi tra -1 e 1 per ognuna delle seguenti metriche: *positive*, *neutral*, *negative* e *compound*. Per le prime tre il valore ottenuto indica quanto un testo appartenga ad ognuna delle tre categorie, mentre *compound* è la sintesi dei tre valori ottenuti.

L'attributo *pos_review*, descritto nel capitolo 2, è stato utilizzato come variabile target. La variabile utilizzata per la classificazione, invece, è stata creata partendo dal *compound*: quest'ultimo è stato prima normalizzato su un intervallo da 0 a 1, ed in seguito è stato reso un attributo binario, chiamato *vader_compound_binary*. Per discretizzarlo è stata seguita la distribuzione di *pos_review*: i valori da 0% a 60% sono stati sostituiti con lo 0, il restante 40% con 1. Nelle Tabelle 4.1 e 4.2 vengono presentati i risultati ottenuti.

Accuracy	Classe	Precision	Recall	F1-score	Support
0.8032	0	0.69	0.70	0.70	17116
	1	0.86	0.85	0.85	36221

Tabella 4.1: Evaluation di VADER binario

TP: 30823	**FN:** 5398
FP: 5095	**TN:** 12021

Tabella 4.2: Confusion matrix di VADER binario

Alla luce dei buoni risultati ottenuti si è deciso di testare l'algoritmo per una classificazione multiclasse, utilizzando come variabile target *review_rating*, ossia le stelle assegnate al telefono. In questo caso il *compound*, dopo essere stato normalizzato da 0 a 1, è stato discretizzato nella seguente maniera:

- da 0% a 19%: 1 stella
- da 20% a 39%: 2 stelle
- da 40% a 59%: 3 stelle
- da 60% a 79%: 4 stelle
- da 80% a 100%: 5 stelle

Nella Tabella 4.3 sono presentati i risultati ottenuti.

Accuracy	Classe	Precision	Recall	F1-score	Support
0.4844	1	0.61	0.24	0.34	10136
	2	0.13	0.23	0.17	3117
	3	0.11	0.24	0.15	3863
	4	0.18	0.24	0.20	7090
	5	0.76	0.69	0.72	29131

Tabella 4.3: Evaluation di VADER multiclasse

Prevedibilmente i risultati migliori sono stati ottenuti per la classe 1 e 5. Queste due classi, oltre ad essere le più numerose nel dataset (Figura 1.3), sono le classi più estreme: è evidente come sia più facile classificare una recensione estremamente positiva o estremamente negativa, a prescindere dal classificatore utilizzato.

Infine si è deciso di modificare la discretizzazione del *compound* per la classificazione binaria, in modo da ottenere un metro di paragone per i risultati ottenuti in precedenza. In particolare è stata semplificata la discretizzazione del *compound*: in questa istanza, tutti i *compound* positivi sono stati identificati con 1, ossia come recensioni positive, mentre i *compound* negativi sono stati approssimati a 0, ossia come recensioni negative. I risultati, come dimostrano le Tabelle 4.4 e 4.5, sono in linea con quanto ottenuto in precedenza.

Accuracy	Classe	Precision	Recall	F1-score	Support
0.8046	0	0.72	0.65	0.68	17116
	1	0.84	0.88	0.86	36221

Tabella 4.4: Evaluation di VADER

TP: 31847	**FN:** 4374
FP: 6049	**TN:** 11067

Tabella 4.5: Confusion matrix di VADER

5 Classificazione

Nello svolgimento della Sentiment Analysis si è deciso di sperimentare il *content drift.* Come descritto nel paragrafo 2, il dataset è stato diviso in test set, contenente le 20000 osservazioni più recenti, training set 2, contenente le 10000 osservazioni antecedenti quelle del test set, ed infine training set 1, contenente le 10000 osservazioni antecedenti quelle del training set 2. L'ipotesi da verificare è che le osservazioni più prossime a quelle del test set siano le più efficaci nella classificazione di queste ultime. Questa metodologia è stata applicata ai classificatori BERT e SVM.

5.1 BERT

BERT, o *Bidirectional Encoder Representations from Transformers*, è un algoritmo di deep learning pre-addestrato. L'algoritmo, sviluppato e pubblicato da Google nel 2018, presenta delle caratteristiche che lo rendono particolarmente all'avanguardia nell'ambito del Natural Language Processing. Il classificatore BERT è:

- bidirezionale: ogni parola viene considerata nel suo contesto, poiché in ogni momento vengono prese in considerazione le parole antecedenti e successive al token analizzato.
- *attention-based*: l'*attention* cattura il contributo di ogni *input token* nella generazione degli *output tokens*. In altre parole, è un sistema per collegare gli elementi appartenenti a due sequenze attraverso la correlazione tra gli stessi. Un'altra particolarità risiede nell'utilizzo delle *attention mask*, per cui alcune parole vengono mascherate casualmente durante ogni iterazione.
- *transformer-based*: l'architettura *transformer* è utilizzata per problemi di tipo *sequence-to-sequence*, come ad esempio le traduzioni o il riassunto di testi. Sfrutta il concetto dell'*attention* abbinato all'architettura *encoder-decoder*.

Dopo aver utilizzato il *BertTokenizer* è stato necessario aggiungere i token speciali previsti da BERT, ossia *'CLS'* all'inizio di ogni testo e *'SEP'* alla fine. È stato necessario definire *MAX_LENGHT*, ossia la lunghezza massima delle frasi del *training set* e del *test set*. Dal momento che la maggior parte delle recensioni non contano molte parole, è stato scelto il limite di 25 token. Successivamente i token sono stati convertiti in un *integer index*, in cui ogni token è sostituito da un valore numerico.
Dopo aver impostato le *attention mask*, con valore 1 per i token e 0 per i *padding*, è stato importato il modello pre-addestrato *bert-base-uncased*, poi applicato ad entrambi i *training set* per il *content drift*. I risultati sono descritti nelle Tabelle 5.1, 5.2 e 5.3.

Modello	Accuracy	Classe	Precision	Recall	F1-score
BERT Training Set 1	0.7929	0	0.63	0.77	0.69
		1	0.89	0.80	0.84
BERT Training Set 2	0.8129	0	0.69	0.70	0.69
		1	0.87	0.86	0.87

Tabella 5.1: Evaluation di BERT binario

TP: 11194	**FN:** 2731
FP: 1411	**TN:** 4664

Tabella 5.2: Confusion matrix per Training Set 1

TP: 12001	**FN:** 1924
FP: 1818	**TN:** 4257

Tabella 5.3: Confusion matrix per Training Set 2

5.2 SVM

Anche per il classificatore SVM è stata utilizzata la tecnica del *content drift*. In questo caso, però, si è deciso di testare la classificazione multiclasse, con la variabile *review_rating* come target. Inoltre, per ottenere i risultati migliori possibili sono state sperimentate diverse configurazioni. Il classificatore è stato applicato sia all'attributo *body*, ossia al testo delle recensioni, sia all'attributo *body_lem*, ottenuto con la lemmatizzazione di *body*. La lemmatizzazione del testo consiste nella riduzione di ogni parola al suo lemma, ossia alla forma canonica (ad esempio da *vediamo* a *vedere*). In entrambi i casi sono stati testati due *vectorizer*, ossia il *CountVectorizer* e il *TfidfVectorizer*: il primo trasforma il testo in un vettore basandosi sulla frequenza di ogni parola, mentre il secondo è basato sull'algoritmo TF-IDF. Il secondo approccio, dati i risultati migliori, è stato utilizzato in un'ulteriore classificazione, per la quale è stato effettuato il *tuning* dei parametri con la *grid search*. Sono stati testati i seguenti parametri:

- C: 0.1, 1, 10, 100, 1000
- gamma: 1, 0.1, 0.01, 0.001, 0.0001
- kernel: rbf

I risultati sono descritti nella Tabella 5.4.

Alla luce dei buoni risultati ottenuti si è deciso di svolgere un'ulteriore classificazione con SVM, finalizzata all'ottenimento dei risultati migliori possibili. Per questa ragione sono stati uniti i quattro dataset (poi divisi con *test size* del 33%) ed è stata scelta come variabile target l'attributo binario *pos_review*, già utilizzato con VADER e BERT. Come in precedenza, sono stati testati due *vectorizer* ed infine è stata applicata la *grid search*. Come prevedibile, il modello ha portato a risultati eccellenti, come è possibile osservare dalla Tabella 5.5.

Modello	Accuracy		Classe	*body*		*body_lem*	
	body	*body_lem*		Precision	Recall	Precision	Recall
CountVectorizer Training Set 1	0.6907	0.6935	1	0.58	0.76	0.59	0.78
			2	0.50	0.00	0.00	0.00
			3	0.27	0.02	0.33	0.01
			4	0.47	0.01	0.71	0.01
			5	0.73	0.96	0.73	0.96
CountVectorizer Training Set 2	0.6873	0.6868	1	0.62	0.71	0.62	0.72
			2	1.00	0.00	0.00	0.00
			3	0.56	0.01	0.33	0.00
			4	0.36	0.00	0.56	0.01
			5	0.71	0.97	0.71	0.97
TfidfVectorizer Training Set 1	0.7179	0.7067	1	0.61	0.89	0.58	0.87
			2	0.00	0.00	0.33	0.01
			3	0.31	0.02	0.55	0.03
			4	0.45	0.04	0.42	0.04
			5	0.77	0.96	0.76	0.95
TfidfVectorizer Training Set 2	0.7161	0.7062	1	0.64	0.85	0.62	0.83
			2	0.29	0.00	0.14	0.00
			3	0.48	0.01	0.56	0.01
			4	0.50	0.02	0.43	0.02
			5	0.74	0.97	0.74	0.97
TfidfVectorizer Training Set 1 con *GridSearch*	0.7150	0.7067	1	0.63	0.87	0.58	0.87
			2	0.22	0.03	0.33	0.01
			3	0.30	0.09	0.55	0.03
			4	0.37	0.16	0.42	0.04
			5	0.79	0.92	0.76	0.95
TfidfVectorizer Training Set 2 con *GridSearch*	0.7168	0.6985	1	0.65	0.84	0.62	0.81
			2	0.26	0.03	0.17	0.03
			3	0.29	0.05	0.29	0.05
			4	0.36	0.11	0.31	0.10
			5	0.77	0.95	0.76	0.93

Tabella 5.4: Evaluation di SVM multiclasse con content drift

Modello	Accuracy	Classe	Precision	Recall	F1-score
CountVectorizer	0.8895	0	0.88	0.76	0.82
		1	0.89	0.95	0.92
TfidfVectorizer	0.9142	0	0.89	0.84	0.86
		1	0.93	0.95	0.94
TfidfVectorizer con *GridSearch*	0.9165	0	0.89	0.85	0.87
		1	0.93	0.95	0.94

Tabella 5.5: Evaluation di SVM binario senza content drift

5.3 Risultati

Come prevedibile, la classificazione binaria ha portato a risultati migliori della classificazione multiclasse. L'accuracy del 91% testimonia che il modello di classificazione binaria con SVM sia uno strumento efficace per la *Sentiment Analysis*. Per quanto riguarda la classificazione multiclasse, le classi più facili da classificare sono le classi 1 e 5, ossia le più estreme. Tuttavia, questo tipo di classificazione, già di per sé più complessa, è necessariamente viziata dalla possibile incoerenza delle recensioni con il numero di stelle assegnate al prodotto: come specificato nell'introduzione, non è detto che tutti gli acquirenti adottino lo stesso metro di giudizio.

Per quanto riguarda il *content drift*, l'ipotesi che le recensioni più recenti siano le migliori per classificare il *test set* è verificata per BERT, ma non per SVM. Per quest'ultimo classificatore si è deciso di sperimentare un approccio più estremo: il *Training Set 2* è stato messo a confronto con le prime 10000 recensioni del *Remaining Set*, ossia le più distanti temporalmente dal *test set*. Dal momento che la *grid search* non ha portato a miglioramenti significativi, per il confronto si è utilizzato il modello con *TfidfVectorizer*. I risultati, presentati nella Tabella 5.6, dimostrano che in questo caso l'ipotesi è verificata anche per SVM.

Modello	Accuracy	Classe	Precision	Recall	F1-score
TfidfVectorizer Remaining Set [0:10000]	0.6985	1	0.68	0.74	0.71
		2	0.00	0.00	0.00
		3	0.31	0.02	0.04
		4	0.39	0.06	0.11
		5	0.71	0.97	0.82
TfidfVectorizer Training Set 2	0.7161	1	0.64	0.85	0.73
		2	0.29	0.00	0.00
		3	0.48	0.01	0.02
		4	0.50	0.02	0.04
		5	0.74	0.97	0.84

Tabella 5.6: SVM, confronto tra Training Set 2 e Remaining Set