Data Mining I

Aldo Cerulli[1], Francesco Daquino[2], and Mattia Arancio Febbo[3]

[1]*Dipartimento di Filologia Letteratura Linguistica, Piazza Torricelli, 2, 56126, Pisa.*
Email: a.cerulli1@studenti.unipi.it
[2]*Dipartimento di Informatica, Largo Bruno Pontecorvo, 3, 56127 Pisa.*
Email: f.daquino@studenti.unipi.it
[3]*Dipartimento di Informatica, Largo Bruno Pontecorvo, 3, 56127 Pisa.*
Email: m.aranciofebbo@studenti.unipi.it

Abstract

Nel corso del nostro lavoro abbiamo avuto l'opportunità di esplorare e analizzare un dataset per comprendere meglio e mettere in pratica le tecniche di data mining apprese durante il corso. Le fasi principali dell'analisi includono la comprensione e preparazione dei dati, il clustering, la classificazione, la regressione e il pattern mining. In primo luogo, abbiamo esplorato il dataset per valutarne la qualità complessiva, comprenderne la struttura e le relazioni tra le variabili, nonché identificare le feature più rilevanti. Abbiamo poi applicato tecniche di clustering per individuare gruppi omogenei di osservazioni, rivelando così relazioni nascoste e ottenendo una visione della struttura interna del dataset. Successivamente, abbiamo utilizzato algoritmi di machine learning per classificare le istanze del dataset e stimare valori numerici di variabili di interesse tramite regressione. Ciò ci ha permesso di sviluppare modelli predittivi e valutarne le prestazioni utilizzando metriche adeguate. Infine, abbiamo applicato le tecniche di pattern mining per scoprire relazioni, tendenze o comportamenti ricorrenti all'interno del nostro dataset, sfruttando le regole di associazione ritenute più rilevanti per effettuare predizioni rispetto a una variabile target.

1 Data Understanding and Preparation

1.1 Data semantics

Il seguente progetto presenta un'analisi dettagliata condotta su un dataset che trae origine dal RAVDESS dataset (Ryerson Audio-Visual Database of Emotional Speech and Song): un insieme di registrazioni audio e video di attori professionisti che recitano una serie di frasi, parole e vocalizzi in diversi contesti emozionali e con un accento neutro nordamericano. In particolare, il dataset finale estratto si compone di 2452 record e 38 feature, dieci delle quali di tipo categorico e 28 di tipo numerico, riguardanti le misure di sintesi e le informazioni tecniche dei file audio contenuti nel dataset originale.

Proseguendo con una descrizione più dettagliata delle singole variabili, le categoriche forniscono informazioni riguardanti: il tipo di registrazione ('modality'); se la registrazione è un parlato o un cantato ('vocal_channel'); il tipo di emozione con cui vengono pronuciate o cantate le espressioni di ogni registrazione ('emotion') e i due livelli di intensità emotiva delle stesse ('emotional_intensity'); le due espressioni pronunciate dagli attori, oggetto di ogni registrazione ('statement') e la relativa indicazione sul numero di ripetizioni ('repetition'); il sesso dell'attore ('sex') e i valori relativi al numero identificativo dell'attore che parla o canta lo 'statement' ('actor'); il numero di canali audio relativi alle cateogorie mono e stereo ('channels'), ed infine, il numero di byte per campione ('sample_width'), i cui valore 1 e 2 indicano rispettivamente 8 bit e 16 bit. Le classi di ogni variabile descritta e la relativa frequenza sono riassunte in Tabella 1.
La maggior parte delle variabili del dataset sono invece numeriche continue. In particolare:

- 'frame_width': indica il numero di byte per ogni frame (2 o 4);
- 'frame_rate': descrive la frequenza, espressa in Hertz, dei campioni utilizzati;
- 'length_ms': indica la lunghezza del file audio in millisecondi;

Tabella 1. Variabili categoriche, classi associate e relative frequenze.

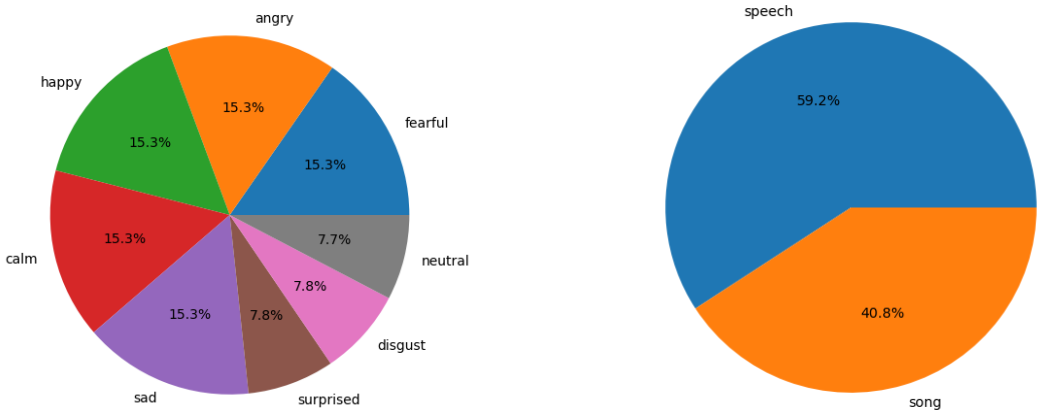
Feature	Classi	Frequenza
modality	audio-only	2452
vocal_channel	speech	1335
	song	921
emotion	fearful, angry, happy, calm, sad	376
	surprised, disgust	192
	neutral	188
emotional_intensity	normal	1320
	strong	1132
statement	Kids are talking by the door	1226
	Dogs are sitting by the door	1226
repetition	1st, 2nd	1226
sex	M	1248
	F	1204
channels	1	2446
	2	6
actor	da 1 a 24	1326
sample_width	2	2452

- 'frame_count': esprime il numero di fotogrammi del campione;
- 'intensity': è la misura del volume percepito della registrazione in dBFS ('Decibels relative to full scale'). I decibels come unità di misura dell'audio nel mondo digitale, eprimono il rapporto tra un segnale audio e il massimo livello consentito in digitale, ovvero 0 dBFS, e questo spiega perché nel dataset, per tale attributo, si assumono solo valori in un intervallo negativo;
- 'zero_crossings_sum': è una misura che indica la frequenza con cui un segnale attraversa l'asse zero in un certo periodo di tempo o in un intervallo specifico, calcolata determinando il rapporto tra il numero di attraversamenti dello zero nel segnale (ovvero il numero di volte in cui il segnale cambia di segno) e la sua durata totale. In tal modi si è in grado descrivere la variabilità o la dinamicità del segnale;
- 'mean', 'std', 'min', 'max', 'kur', 'skew': sono le statistiche del segnale audio originale, ed esprimono, in ordine, informazioni relative alla media, deviazione standard, minimo, massimo ed indici di asimmetria e curtosi per ciascuna osservazione.
- 'mfcc_mean', 'mfcc_std', 'mfcc_min', 'mfcc_max': sono le statistiche dei coefficienti cepstrali di frequenza Mel;
- 'sc_mean', 'sc_std', 'sc_min', 'sc_max', 'sc_kur', 'sc_skew': sono le statistiche del centroide spettrale (SC);
- 'stft_mean', 'stft_std', 'stft_min', 'stft_max', 'stft_kur', 'stft_skew': sono le statistiche del cromogramma STFT (Short-Time Fourier Transform).

1.2 Distribuzione delle variabili e statistiche

Tale sezione prevede una fase di esplorazione quantitativa delle variabili, utile per comprendere meglio le loro principali proprietà e statistiche descrittive, fondamentali per poter procedere alla successiva fase di pulizia e valutazione della qualità dei dati.

Delle variabili categoriche presentate in Tabella 1 si è scelto di rappresentare graficamente tramite pie chart (Figura 1) la distribuzione percentuale di 'emotion' e 'vocal_channel', in quanto selezionate come variabili target delle tecniche di classificazione descritte in sezione 3:
Si può facilmente notare come sia la variabile binaria (dopo la sostituzione dei missing values effettuata in sezione 1.3) che la multiclasse siano abbastanza bilanciate, ovvero le frequenze delle classi siano ben distribuite in termini di istanze nel dataset. In particolare, è da evidenziare che le emozioni sorpresa e disgusto abbiano la metà della frequenza delle altre emozioni, e ciò è da attribuire al fatto che non ci sono osservazioni relative



(a) (b)

Figura 1. Distribuzione percentuale delle variabili emotion (a) e vocal_channel (b).

ala classe 'song' di vocal channel con tali emozioni; mentre le emozioni di neutralità hanno una frequenza dimezzata, poiché nessuna osservazione con tale emozione è pronunciata con intensità emotiva 'strong'.

Da una successiva esplorazione del dataset, appare subito evidente come le variabili 'modality', 'sample_width', 'frame_rate' e 'stft_max' siano accomunate dal fatto che assumono un singolo valore per ogni osservazione (Tabella 2), e pertanto saranno soggette a successiva eliminazione dal dataset in quanto non apportano alcun contributo informativo.

Tabella 2. Feature con valore unico.

Feature	Valore	Tipo	Numero
modality	audio-only	object	2452
sample_width	2	int64	2452
frame_rate	48000	int64	2452
stft_max	1.0	float64	2452

In seguito sono state esaminate le distribuzioni delle variabili continue tramite density plot, che forniscono un importante supporto nell'identificazione di eventuali picchi, code, modalità multiple o regioni di alta densità. Dal momento che la maggior parte delle variabili segue una distribuzione normale è stata posta maggior attenzione alle variabili riportate in Figura 2.
Tra quelle mostrate, molte risultano essere asimmetriche (negativamente o positivamente), mentre altre presentano alcune caratteristiche evidenti, in particolare:

- i due picchi presenti per le distribuzioni delle variabili 'sc_min' (3b) e 'frame_count' (3c) hanno rivelato la presenza di valori anomali, approfondita in fase di valutazione della qualità dei dati in sezione 1.3;
- la varibile 'lenght_ms' ha una distribuzione bimodale, come mostrato nella Figura 3a. Questo perché le canzoni e le registrazioni parlate seguono due distinte distribuzioni normali e, in media, le canzoni sono più lunghe;
- le rimamenti variabili, valutata la loro asimmetria, saranno oggetto di trasformazione logaritmica (sezione 1.5) al fine di rendere la loro distribuzione più simmetrica o approssimativamente normale.

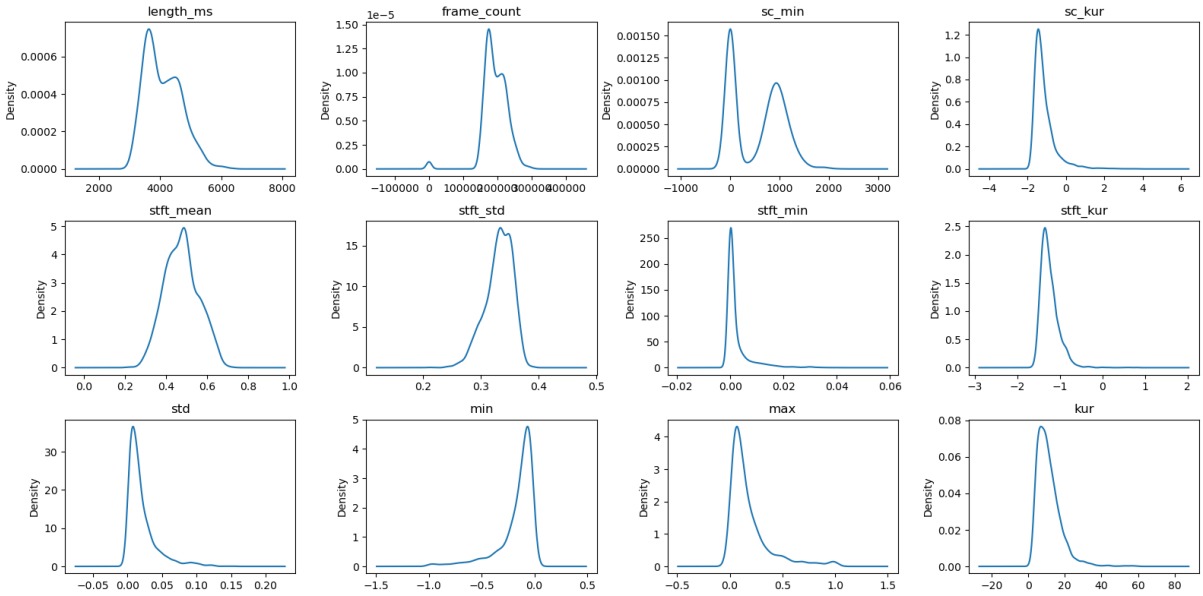
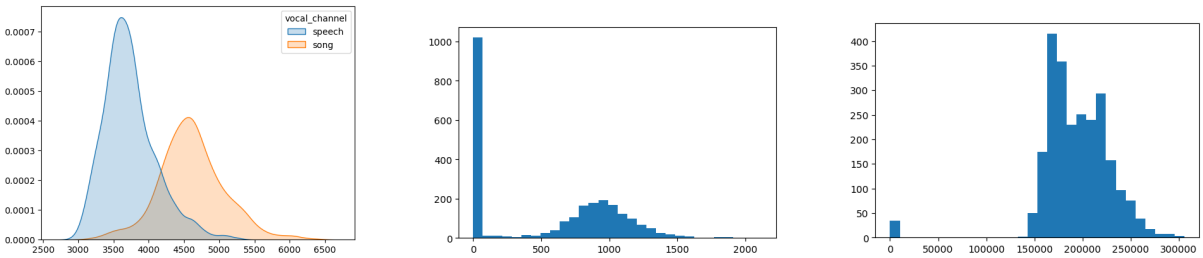


Figura 2. *Density plot* delle variabili continue rilevanti.



(a) (b) (c)

Figura 3. Visualizzazione delle distribuzioni di length_ms (a), sc_min(b) e frame_count(c).

1.3 Valutazione della qualità dei dati e riduzione dataset

Il dataset presenta in totale 2138 missing values, distribuiti nel seguente modo: 816 in 'intensity', 196 in 'vocal_channel' e 1126 in 'actor'. A questi si aggiungono, come già anticipato, i 35 valori uguali a -1 della variabile 'frame_count' e i 1021 valori uguali a 0 della variabile 'sc_min', in entrambi i casi considerati come valori anomali (3b e 3c). Infatti, per entrambe le variabili, tali valori sono stati considerati significativamente al di fuori dei range di valori assumibili, sulla base del dominio e delle caratteristiche tipiche dei segnali audio. La prima, infatti, dovrebbe rientrare in un intervallo positivo che riflette il numero di frame effettivamente presenti nel segnale, mentre per la seconda è anomalo il fatto che il minimo valore di "spectral centroid" assuma cosi tanti valori pari a zero essendo questa una caratteristica acustica che indica la posizione spettrale media di un segnale audio.

Tali valori sono stati gestiti tramite l'utilizzo di diverse tecniche. In particolare, i valori mancanti/anomali delle variabili numeriche 'intensity' ed 'sc_min' sono stati sostituiti tramite regressione, tecnica eseguita ed ampiamente descritta nella sezione 4 del report. Con riferimeno alla sostituzione dei valori mancanti relativi alla variabile categorica 'vocal_channel' si è deciso di procedere nel seguente modo: per prima cosa sono stati contrassegnati come 'speech' tutte le osservazione aventi disgusto o sorpreso come classe di 'emotion', dal momento che nel dataset di partenza non risultavano osservazioni con tali valori appartenenti alla classe 'song'. I 172 valori mancanti rimanenti sono stati imputati come 'song' e 'speech' utilizzando un metodo di ripartizione per genere (M o F), sulla base dei dati riportati nel paper originale del dataset, ed ottenendo i valori finali mostrati in Tabella 3:

Tabella 3. Ripartizione finale per genere dei valori mancanti di 'vocal_channel'.

Classi	F	M
song	434	487
speech	671	664

Le due variabili rimanenti da considerare sono 'actor' e 'frame_count'. Per la prima si è deciso di procedere con la sua eliminazione immediata dal dataset, sia per l'irrilevanza del tenere traccia dell'identificativo di ciascun attore che per il notevole numero di valori mancanti; riguardo alla variabile 'frame_count', sebbene si era inizialmente pensato di sostituire i pochi valori mancanti con quello ottenuto dalla sua media,è stata anch'essa eliminata a causa dell'alta correlazione con la variabile 'length_ms' (sezione 1.4). Infatti, dal momento che 'frame_count' rappresenta la divisione di un segnale audio in segmenti temporali discreti e 'length_ms' indica la durata totale del segnale audio, è ragionevole aspettarsi una correlazione positiva significativa tra le due variabili. Quando i valori di 'frame_count' sono uguali, si osservano corrispondenti valori uguali per 'length_ms'.

La successiva fase prevede l'osservazione e l'identificazione tramite boxplot dei valori che rappresentano outliers per il nostro dataset: utilizzando il criterio del range interquartile infatti è possibile individuare le osservazioni che si discostano significativamente dalla distribuzione dei dati. E' stato rilevato un numero consistente di outliers, con particolare concentrazione nella variabile 'sc_max' che conta ben 439 casi, e in Figura 4 sono stati riportati i soli boxplot delle sei variabili con il maggior numero di outliers. Alcune di queste sono state oggetto di ulteriori approfondimenti o eliminazioni nelle successive sezione del report, mentre altre sono state mantenute parte del dataset, valutando di volta in volta se escluderle o meno dai modelli di clustering o classificazione utilizzati, sulla base dei risultati finali ottenuti.

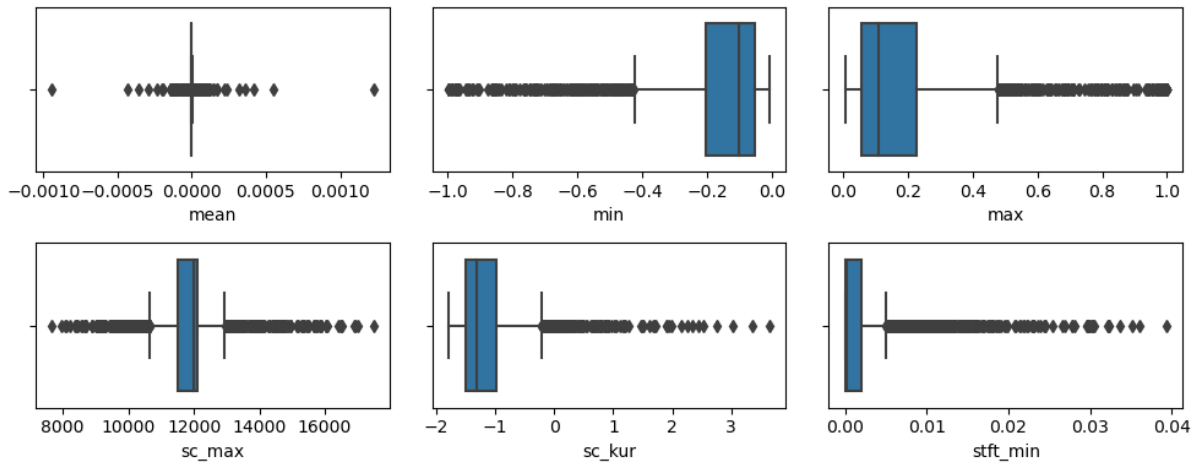


Figura 4. Variabili con maggior numero di outliers.

Per avere risultati più accurati e significativi possibili e ridurre la complessità dei dati, si è deciso di eliminare, come già accennato prima, le variabili 'modality', 'actor', 'sample_width', 'frame_rate' e 'stft_max' in quanto attributi che assumono lo stesso valore per ogni osservazione. E' stata eliminata anche la variabile 'frame_width' in quanto uguale alla variabile 'channels' in termini di informazioni e quindi ridondante (un file audio stereo ha infatti il doppio dei byte di un file audio mono in ogni frame), ed infine, sono stati resi numerici interi (come 1 e 2) i due valori assunti dell'attributo 'repetition', in origine '1st' e '2nd'.

1.4 Analisi di correlazione

In questa sezione è stata condotta un'analisi delle correlazioni tra le variabili quantitative del dataset utilizzando l' indice di correlazione di Pearson, e valutando una loro eventuale eliminazione. La Figura 5 mostra il grafico heatmap da cui emergono alcune correlazioni significative tra le variabili. Una forte correlazione si può notare tra le variabili 'length_ms' e 'frame_count', 'intensity' e 'mfcc_std', 'intensity' e 'mfcc_min', 'mfcc_std' e 'mfcc_min', 'stft_mean' e 'stft_skew', 'std' e 'min', 'std' e 'max', 'min' e 'max'.

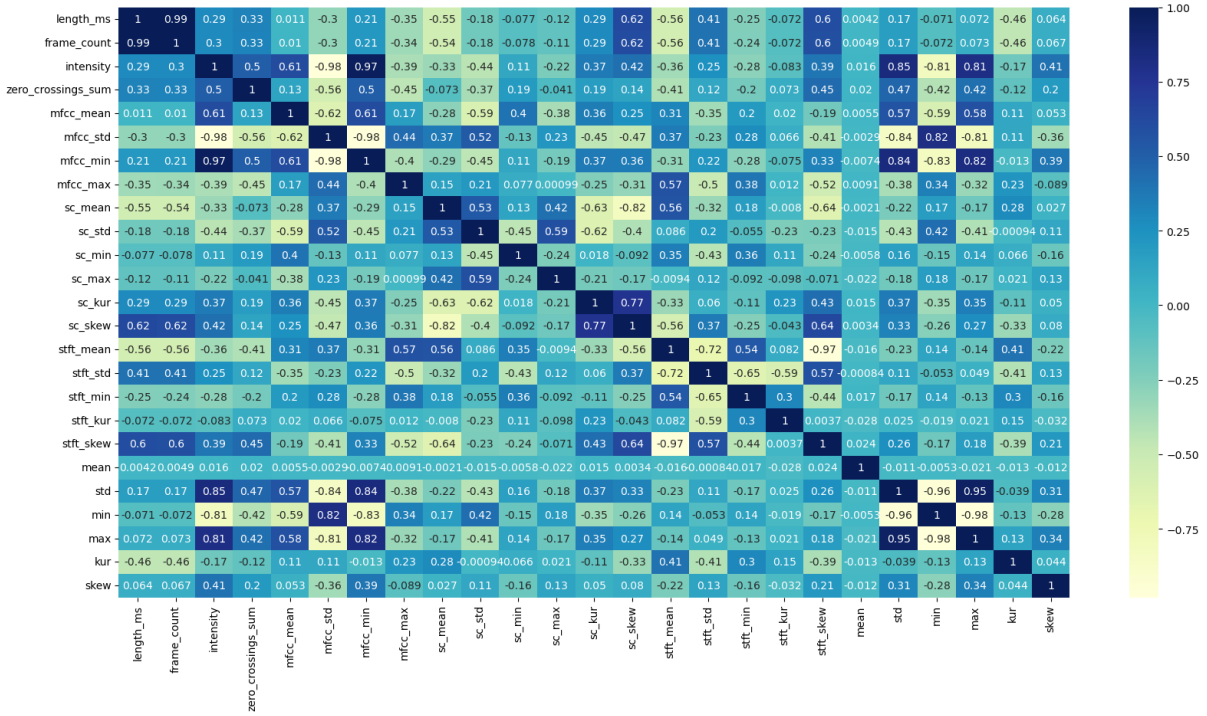
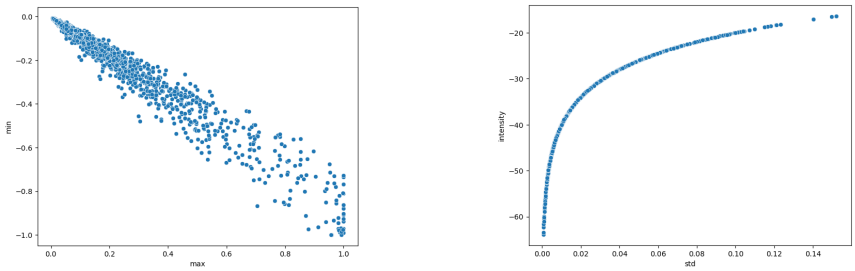


Figura 5. Matrice di correlazione tramite heatmap.

Tra quelle già citate, sono state esaminate e riportate tramite scatterplot (Figura 6) alcune correlazioni rilevanti, come ad esempio quelle tra le variabili 'min' e 'max' e tra 'intensity' e 'std'. Entrambe le relazioni possono essere spiegate tramite riferimento alla natura del segnale audio: nel primo caso la sua oscillazione naturale potrebbe portare a registrare picchi elevati del segnale ma di segno opposto, mentre il secondo caso suggerisce che quando l'intensità del segnale audio aumenta, è probabile che aumenti anche la sua variabilità o dispersione, discostandosi dal valore di base 0 (in particolar modo dopo la trasformazione logaritmica in sezione 1.5 la correlazione aumenta fino ad essere prossima ad 1, diventando perfettamente lineare).



(a) (b)

Figura 6. Scatterplot tra 'max' e 'min' (a) e tra 'intensity' e 'std' (b).

Al termine dell'analisi, si è deciso di eliminare una variabile da ciascuna coppia presa in considerazione utilizzando una soglia minima di correlazione pari a 0.85. Sulla base di quanto già detto prima, si è scelto di eliminare la variabile 'frame_count', e le variabili 'mfcc_min', 'mfcc_std', 'stft_skew', 'max' e 'min', così ridurre la ridondanza di informazioni nel nostro dataset e migliorare la precisione delle analisi successive, concentrandoci sulle variabili più informative, indipendenti, e con minor numero di outliers.

1.5 Trasformazione delle variabili

Valutando l' asimmetria delle variabili numeriche rimanenti, si è deciso di applicare una trasformazione logaritmica ai valori delle variabili con indici di asimmetria > 1 o < -1 (Figura 7), soglie oltre le quali possono essere considerate variabili altamente asimmetriche postivamente e negativamente.

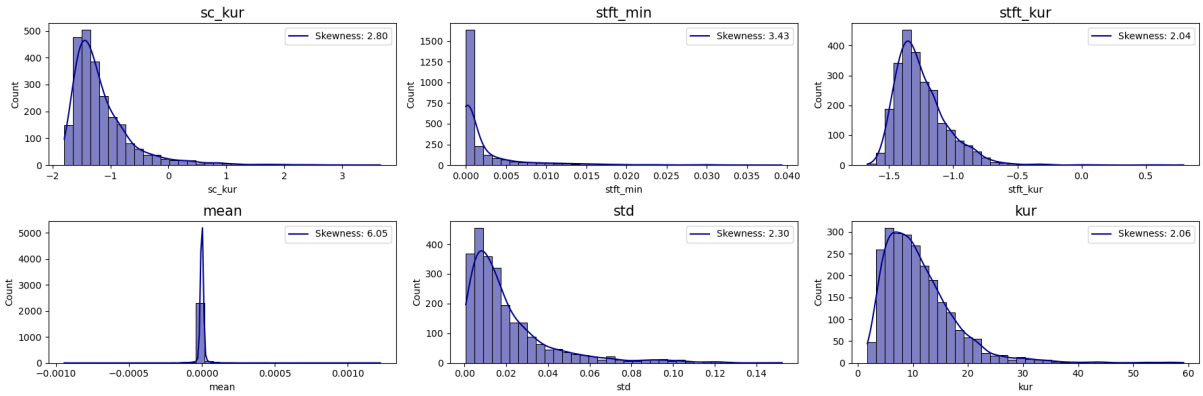


Figura 7. Variabili asimmetriche con relativo indice di skewness.

La scelta di eseguire tali trasformazioni è stata presa in funzione di un miglioramento della qualità e dell'utilizzo dei dati: ridurre l'asimmetria dei dati infatti riduce la scala dei valori che si distribuiscono in modo più uniforme, facilitando l'interpretazione dei dati e migliorare l'adattamento ad alcuni modelli statistici. In alcune situazioni, la trasformazione logaritmica può contribuire a stabilizzare la varianza dei dati, può rendere le relazioni tra le variabili più lineari e può ridurre l'impatto degli outliers. Sulla base di tali considerazioni è stato possibile affermare che correggendo l'asimmetria positiva delle variabili 'sc_kur', 'stft_min', 'stft_kur', 'mean', 'std' e 'kur', la loro distribuzione si è avvicinata a una distribuzione normale, e che per alcune è diminuito il numero di outliers rilevati. Ciò può contribuire a rendere l'analisi più robusta e meno influenzata da valori estremi che potrebbero distorcere le stime o i risultati. Un' ulteriore eliminazione è avvenuta per la variabile 'mean' in quanto fortemente asimmetrica, incorrelata alle altre variabili e contenente un' elevato numero di outliers, quindi si è ritenuto potesse influire negativamente sulle successive fasi di clustering e classificazione.

Al termine di questa lunga fase di comprensione e preparazione dei dati, e avendo effettuato tutte le operazioni necessarie di eliminazione, sostituzione e traformazione di varabili, il dataset risulta essere composto da 24 feature e 2452 osservazioni, pronto alle successive analisi tramite tecniche di clustering, classificazione e pattern mining.

2 Clustering

Nel corso di questa sezione, sono state esaminate in dettaglio alcune delle tecniche di clustering più importanti, adottate per valutare eventuali raggruppamenti omogenei e similarità tra gli elementi all'interno del nostro dataset. Sono state analizzate anche le metriche di valutazione della qualità dei cluster, le strategie di determinazione del numero ottimale di cluster e le considerazioni sulla dimensionalità dei dati. K-means, clustering gerarchico e DBSCAN sono le tre tecniche di clustering implementate per la nostra analisi, supportate da tecniche di visualizzazione dei dati che rendono l'interpretazione dei risultati stessi più intuitiva. A tal fine sono state

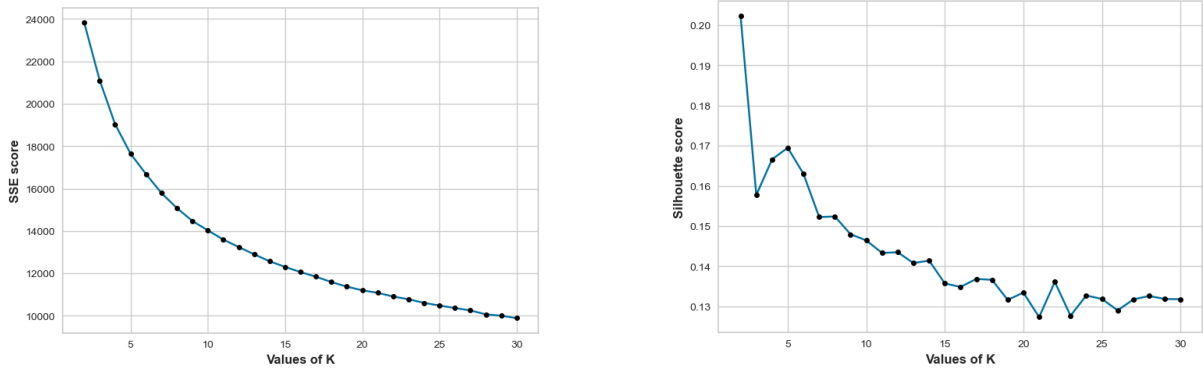
considerate solo le variabili numeriche del dataset, sulle quali è stata eseguita una trasformazione dei valori tramite standardizzazione (classe StandadScaler di Scikit-learn), in modo da rendere le variabili comparabili tra loro, eliminando eventuali differenze nella scala e nell'unità di misura. Per l'implementazione degli algoritmi k-means e gerarchico si è deciso di escludere dal dataset gli outlier rilevati precedentemente e le variabili 'sc_max', 'stft_min', essendo quelle in assoluto con maggior numero di outlier, mentre per l'algoritmo DBSCAN si è deciso di non modificare i dati essendo progettato per gestire efficacemente punti di rumore nel dataset. Come misura di distanza, si è scelto di utilizzare la distanza Euclidea.

2.1 K-means

Durante l'analisi, sono state utilizzate due misure per determinare il numero ottimale di cluster K e implementare correttamente l'algoritmo K-means: SSE e Silhouette. In particolare, sono stati esaminati valori di K compresi tra 2 e 30, 'K-means++' è stato utilizzato come parametro per trovare i valori iniziali dei centroidi, mentre i valori di 'n_init' e 'max_iter' sono rimasti quelli predefiniti, cioè 10 e 100 rispettivamente. Per ogni K, sono stati calcolati i relativi valori di SSE e Silhouette, ed i risultati sono mostrati nella figura 8. Non è stato possibile identificare un'area significativa nel grafico SSE relativo al 'gomito' della curva per determinare il numero ottimale di cluster. Tuttavia, tramite analisi della Silhouette, è stato scelto un valore di K=5. Infatti, nel valutare il numero ottimale di cluster , abbiamo ritenuto ragionevole scegliere il valore di k relativo al secondo valore più alto di silhouette (0.1687) ma che garantisse comunque un SSE relativamente basso, in modo da trovare un compromesso tra l'omogeneità dei cluster (misurata dalla Silhouette) e la compattazione dei cluster (misurata dall'SSE). La tabella 4 riporta i cluster trovati e per ciascun cluster il numero di osservazioni.

Tabella 4. Cluster trovati e numero di osservazioni.

Cluster	#Osservazioni
0	533
1	396
2	447
3	318
4	353



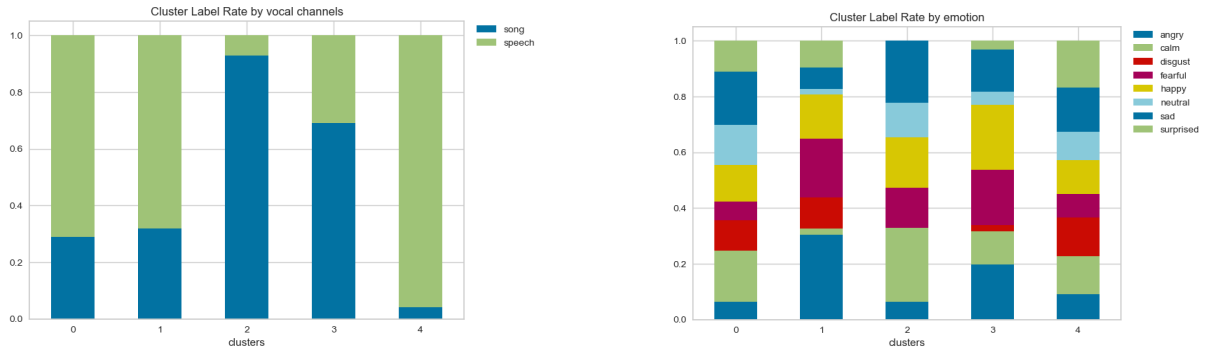
(a) (b)

Figura 8. Valori di SSE (a) e Silhouette (b) al variare di K.

Il clustering ha determinato una distribuzione equilibrata delle osservazioni in tutti i cluster, che, pertanto, risultano avere dimensioni simili tra loro. I grafici in figura 10 mostrano invece la ripartizione tra i vari cluster delle classi relative alle variabili 'vocal_channel' e 'emotion'. Se per la variabile binaria è possibile notare la presenza di cluster contententi quasi esclusivamente l'una o l'altra classe, per la multiclasse è logico aspettarsi una distribuzione più uniforme essendo emotion formata da ben 8 categorie. A supporto dei seguenti grafici, sono state valutate le statistiche descrittive dei 5 dataset ottenuti filtrando le righe per osservazioni appartenenti allo stesso cluster nel tentativo di individuarne le caratteristiche più rilevanti, come si può notare in figura 9: il cluster 2, composto per la maggior parte dalla classe 'song', ha in media registrazioni più lunghe e con maggior intensità.

	length_ms	intensity
count	324.000000	324.000000
mean	4792.101852	-36.548757
std	402.171619	5.738787
min	3604.000000	-54.745974
25%	4504.000000	-40.046576
50%	4738.000000	-36.650292
75%	5072.000000	-32.649987
max	5806.000000	-21.521703

Figura 9. Stat. descrittive cluster 2.



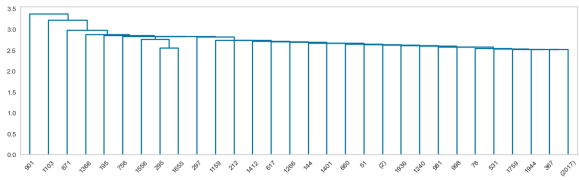
(a) vocal_channel

(b) emotion

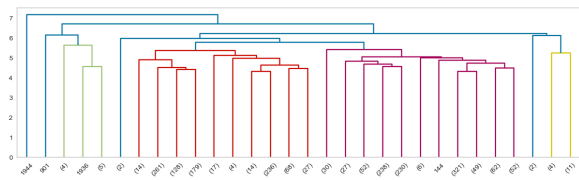
Figura 10. Distribuzione delle osservazioni di *vocal_channel* e *emotion* rispetto ai cinque cluster.

2.2 Clustering gerarchico

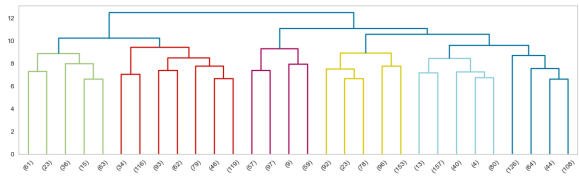
Similmente alla precedente analisi, è stato applicato l'algoritmo di clustering gerarchico al nostro dataset privo di outliers, poichè, come il k-means, è sensibile alla presenza di dati anomali, ma diversamente dal k-means non richiede di specificare un numero iniziale di cluster. Il tipo più comune di clustering gerarchico è quello definito 'agglomerativo' che raggruppa i cluster in base alle loro somiglianze secondo un approccio 'bottom up' in cui si parte dall'inserimento di ciascun elemento in un cluster differente e si procede alla loro fusione graduale. Le metriche utilizzate per misurare la 'distanza' tra i cluster sono 4: single linkage, complete linkage, group average e Ward's linkage. I dendogrammi ottenuti sono mostrati in Figura 11.
Come si può facilmente intuire, il primo metodo è quello che si rivela meno appropriato per il dataset analizzato, in quanto crea un unico cluster che considera quasi tutte le istanze (2017 su 2452). Tuttavia, considerando una distribuzione molto densa di punti nel dataspace (come confermato successivamente dai risultati ottenuti tramite clustering DBSCAN), e che il metodo single linkage unisce i punti sulla base della loro distanza minima, è logico affermare che sia stato creato un unico cluster che include quasi tutte le istanze. Il metodo group average sembrerebbe fornire risultati leggermente migliori del precedente, ma non tanto da poter affermare che le osservazioni all'interno di ciascun cluster sono più simili tra loro rispetto a quelle di altri cluster. Gli ultimi due metodi evidenziano chiaramente la presenza di cluster ben definiti e separati. Ogni cluster infatti mostra una distanza significativa dagli altri, suggerendo una forte coesione interna dei punti che li compongono. Riguardo al metodo Ward's linkage (tabella 5), che ha mostrato i migliori risultati in termini di visualizzazione, è importante sottolineare che ha suggerito lo stesso numero di cluster del modello K-means, e ciò indica che la struttura dei dati a disposizione si presta meglio ad una suddivisione in 5 cluster.



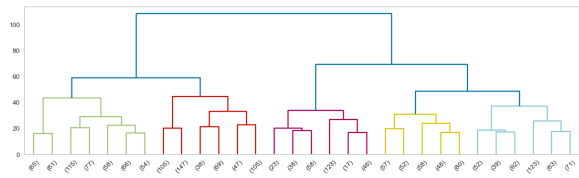
(a) Single linkage



(b) Group average



(c) Complete linkage



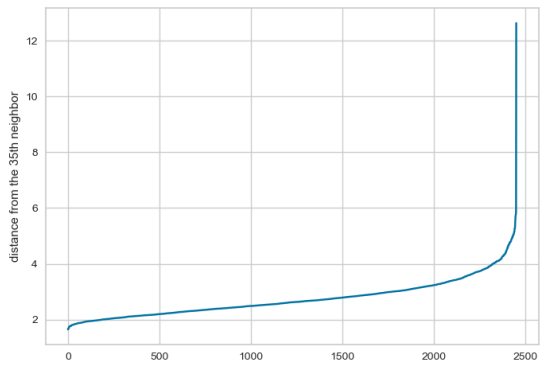
(d) Ward's linkage

Figura 11. Risultati dei dendogrammi dei differenti metodi di fusione.

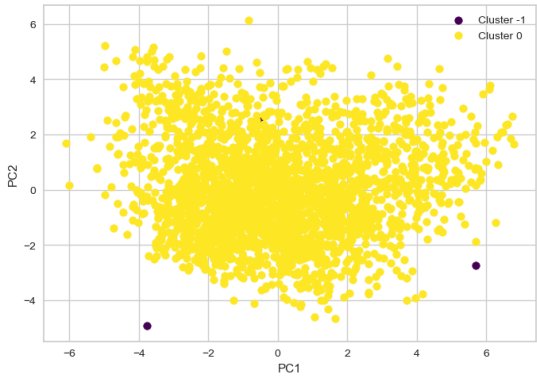
Tabella 5. Risultati metodo Ward's linkage.

Ward's linkage	**Risultato**
Silhouette	0.1138
Array	[511, 496, 305, 440, 295]

2.3 DBSCAN



(a)



(b)

Figura 12. Visualizzazione dei risultati del DBSCAN tramite K-nn distance plot (a) e PCA a due dimensioni (b).

Come ultima tecnica di clustering si è scelto di utilizzare l'algoritmo DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) la cui forza risiede nel suo approccio basato sulla densità, che gli consente di individuare cluster di forma arbitraria e identificare regioni dense di punti nello spazio degli attributi e le raggruppa in cluster. Gli outlier, che si trovano in regioni a bassa densità, non vengono assegnati a nessun cluster specifico. Per la sua implementazione sono richiesti due parametri chiave: il raggio di ricerca (Eps) e il numero minimo di punti necessari per formare un cluster (MinPts).

Per identificare i valori ottimali dei due parametri, date le dimensioni del dataset e la complessità dei dati, sono stati valutati i grafici ottenuti ordinando i punti in base alla loro distanza dal 35-esimo vicino. Tale scelta è stata dettata dai numerosi tentativi effettuati, al termine dei quali è stato possibile osservare che la curva relativa al k-nn distance plot non subiva modifiche significative utilizzando valori diversi di k. Sulla base di tali

considerazioni, e seguendo la pratica comune per cui il valore di k viene scelto come $\sqrt{\frac{n}{2}}$ (dove n è il numero totale di osservazioni nel dataset), è stato imposto k approssimativamente uguale a 35. La figura 12a mostra il valore migliore del raggio epsilon in corrispondenza del gomito della curva, pari a circa 4.2.
Il risultato ottenuto dall'algoritmo DBSCAN mostra un unico cluster molto denso (2449 osservazioni) e solo 3 punti di rumore rilevati, come si può notare dalla visualizzazione tramite tecnica di riduzione dimensionale PCA a due dimensioni in figura 12b. DBSCAN è infatti progettato per rilevare cluster basati sulla densità locale dei punti, e mostra tutti suoi limiti quando i punti dei cluster regolari sono molto vicini tra loro e hanno una densità uniforme, non riuscendo a distinguere tra core point e border point.

2.4 Considerazioni finali

Volendo fare un confronto tra le tre tecniche di clustering utilizzate, il DBSCAN sembra essere il meno adatto al nostro dataset, dal momento che, come descritto in precedenza, la distribuzione dei punti nello spazio appare abbastanza densa e uniforme. Confrontando, inoltre, i valori di Silhouette in tabella 6 dei tre algoritmi, si è deciso di scegliere l'algoritmo K-means come il migliore, poiché il valore più alto di Silhouette rispetto al clustering gerarchico indica maggiore qualità della struttura dei gruppi formati e maggiore coesione interna dei dati, sebbene non si possa ritenere un valore 'ottimale'.

Tabella 6. Valori di silhouette per tecnica di clustering.

Tecnica	Silhouette
K-Means	0.1687
Hierarchical (ward's method)	0.1138
Dbscan	0.4744

3 Classificazione

In questa sezione verranno brevemente descritti i processi e i risultati più interessanti di numerosi esperimenti di classificazione binaria e multiclasse, condotti scegliendo rispettivamente 'vocal_channel' ed 'emotion' come variabili target. Il dataset di partenza, costituito da 24 feature (cinque categoriche e diciassette numeriche), è il risultato delle operazioni di riduzione, trasformazione e sostituzione delle variabili descritte nella Sez. 1. Le variabili categoriche 'vocal_channel', 'emotion', 'emotional_intensity', 'statement', 'sex', 'repetition' e 'channels' sono state discretizzate. Per entrambe le tipologie di classificazione, dopo aver separato la variabile dipendente *y* dal resto del dataset, il 70% dei dati è stato destinato all'addestramento dei modelli ed il restante 30% ai test. Tutti i risultati presentati sono frutto di dieci cicli addestramento-test dove ogni addestramento a sua volta è una 5-cross-validation ripetuta dieci volte, realizzata tramite GridSearchCV e altre funzioni di Scikit-learn.

3.1 Classificazione binaria

Per rendere il dataset idoneo all'esecuzione degli esperimenti di classificazione binaria, la variabile 'emotion' è stata trattata con la tecnica *One-hot encoding*, che ha comportato l'aggiunta di otto nuove variabili binarie (una per ciascuna emozione) con conseguente passaggio del numero di variabili totali da 24 a 31, e la rimozione di 'emotion' dal dataset. In riferimento a quanto già descritto in Sez. 1.2, si può notare un leggero sbilanciamento della variabile target in favore della classe 'speech', che potrebbe avere un effetto sui risultati dell'analisi.

3.1.1 K-NN

Il primo algoritmo che abbiamo addestrato è K-NN (*k-Nearest Neighbors*) nell'implementazione *KNeighborsClassifier* fornita dalla libreria Scikit-learn. Dal momento che K-NN è basato sulla nozione di vicinanza tra dati, una

sua corretta applicazione richiede che le variabili numeriche vengano scalate. A tale scopo, si è fatto ricorso alla classe *StandardScaler* della suddetta libreria.

La presenza di variabili sia qualitative che quantitative nel dataset pone il problema di individuare la metrica più adeguata per il calcolo delle distanze tra i record. Ci siamo quindi riferiti alle due metodologie indicate nell'articolo *Distance functions in machine learning: A primer in simple language with few action points*[1], che consistono rispettivamente nel:

1. convertire tutte le features in numeriche, utilizzando l'*one-hot encoding* in presenza di variabili categoriali non binarie, e applicare una metrica di distanza adatta a dati quantitativi;
2. dividere le variabili numeriche da quelle categoriche, calcolare le distanze separatamente avvalendosi di misure appropriate, e combinare i risultati.

Si è deciso di seguire il primo metodo, in quanto, come specificato all'inizio di questa sezione, la fase di preparazione del dataset per gli esperimenti di classificazione aveva previsto la conversione di tutte le variabili categoriali in numeriche binarie. Si è dunque proceduto ad addestrare KNN sulla totalità delle features, utilizzando la funzione GridSearchCV di Scikit-learn per individuare la configurazione ottimale di iperparametri: sono stati testati i valori di *k* compresi tra 1 e 20, le funzioni per il peso delle distanze *uniform* e *distance* e le metriche *euclidean* e *manhattan* per calcolarle. In tabella 7 sono riportate le performance del miglior classificatore ottenuto seguendo questo approccio, a cui nel seguito ci si riferirà con 'TOT_KNN'. Si è poi provato ad addestrare K-NN con la stessa *param_grid* ma isolando le variabili numeriche ('NUM_KNN'). I risultati (tabella 7) sono sorprendenti: le prestazioni del miglior modello ottenuto sono risultate pressoché identiche a quelle del precedente. Ciò potrebbe suggerire che il contributo delle feature qualitative nel *task* classificatorio sia tanto marginale da poterle escludere, ottimizzando il modello, o indicare che l'uso di metriche adatte a dati quantitativi su dati qualitativi, seppur discretizzati, possa ridurre il loro peso ai fini della predizione della classe.

Per questo motivo, in linea con la seconda delle metodologie esposte, abbiamo implementando due approcci che tengono conto della differenza nella natura dei dati: (1) adottare la distanza di Gower[2], che ben si adatta a set di dati misti (nominali, ordinali o numerici), come metrica per il calcolo delle distanze. Non essendo integrata in Scikit-learn, è stato necessario usare il pacchetto gower[3] ('GOW_KNN'); (2) calcolare separatamente le distanze tra variabili in categoriali e numeriche – rispettivamente con le metriche *hamming* e *manhattan* – e combinarle in un'unica matrice ('COM_KNN'). In entrambi i casi, il parametro *metric* è settato a *precomputed* e tanto i dati di training quanto quelli di test sono passati al classificatore sotto forma di matrici di distanze. In tabella i risultati dei migliori classificatori dei due approcci.

Tabella 7. Performance dei migliori modelli ottenuti rispetto all'algoritmo K-NN.

Modello	Classe	Precision	Recall	F1-Score	Accuratezza	metric, n_neighbors, weights
TOT_KNN	Song	0.93	0.93	0.93	0.943	manhattan, 8, distance
	Speech	0.95	0.95	0.95		
NUM_KNN	Song	0.93	0.93	0.93	0.942	manhattan, 12, distance
	Speech	0.95	0.95	0.95		
GOW_KNN	Song	0.87	0.92	0.90	0.913	precomputed, 19, distance
	Speech	0.94	0.91	0.92		
COM_KNN	Song	0.93	0.93	0.93	0.944	precomputed, 14, distance
	Speech	0.95	0.95	0.95		

Si può notare che, fatta eccezione per 'GOW_KNN', le prestazioni dei migliori modelli sono pressoché identiche. L'accuratezza complessiva maggiore è raggiunta da 'COM_KNN', sebbene il numero di *nearest neighbors* a cui ricorre è più alto attestato.

1. https://tinyurl.com/MixedData
2. https://tinyurl.com/gowerD
3. https://pypi.org/project/gower/

3.1.2 Naive Bayes

A differenza dell'algoritmo precedente, i classificatori della famiglia *Naive Bayes* si basano sulla stima delle probabilità, risultando pertanto insensibili alla scala delle features. Se, da una un lato, viene meno la necessità di scalare i dati, dall'altro si deve porre l'attenzione sui due seguenti aspetti:

1. Scikit-learn mette a disposizione più implementazioni di Naive Bayes. Tra le altre, GaussianNB e CategorialNB operano rispettivamente su dati quantitativi e qualitativi. Poiché, come già visto in precedenza, il nostro dataset contiene variabili di ambo i tipi, è stato necessario utilizzarle entrambe;
2. GaussianNB assume che le variabili numeriche siano distribuite normalmente, cosa che in realtà non è vera per tutte le feature del dataset.

Il primo aspetto è stato affrontato adottando un approccio misto ispirato al punto n.6 dell'articolo *Naive Bayes Classifier — How to Successfully Use It in Python?*[4] che si articola in quattro passaggi: (1) addestramento di un CategoricalNB sulle variabili categoriali e di un GaussianNB sulle numeriche; (2) utilizzo dei due modelli per il calcolo delle *probability predictions* separatamente sui dati di addestramento e di valutazione; (3) creazione di un nuovo training set e test set combinando le probabilità relative alla classe 1 fornite dai due modelli per ciascuna partizione; (4) addestramento e valutazione di un GaussianNB sui nuovi dati. Implementando questa strategia, abbiamo quindi addestrato un classificatore generale (a cui nel seguito ci si riferirà con 'TOT_NB'), le cui prestazioni sono riassunte in tabella 8.

Si è poi proceduto allo studio dell'asimmetria e della forma delle variabili numeriche stimando per ciascuna gli indici statistici di skewness e curtosi. Sebbene le trasformazioni descritte in Sezione 1 abbiano in gran parte normalizzato le distribuzioni, alcune anomalie sono rimaste: due feature sono estremamente asimmetriche e allungate (*stft_min* con skew = 2,33 e kurt = 5,42 e *stft_kur* con skew = -1,20 e kurt = 15,29), altre due sono leggermente asimmetriche (*zero_crossings_sum* con skew = 0,74 e *stft_std* con skew = -0,68) e una è simmetrica ma allungata (*sc_max* con kurt = 4,61). A partire da tali evidenze, sono stati condotti alcuni esperimenti di rimozione di queste variabili a gruppi per valutare il loro impatto sulle prestazioni del modello. La tabella 8 mostra i risultati dei migliori classificatori addestrati secondo la strategia descritta, escludendo (1) solo le colonne *stft_min* e *stft_kur* ('PAR_NB_1'), (2) le due precedenti insieme a *zero_crossings_sum* e *stft_std* ('PAR_NB_2'), e (3) le prime due menzionate e *sc_max* ('PAR_NB_3').

Tabella 8. Performance dei migliori modelli ottenuti rispetto all'algoritmo Naive Bayes.

Modello	Classe	Precision	Recall	F1-Score	Accuratezza
TOT_NB	Song	0.85	0.89	0.87	0.893
	Speech	0.92	0.89	0.91	
PAR_NB_1	Song	0.87	0.91	0.89	0.909
	Speech	0.94	0.91	0.92	
PAR_NB_2	Song	0.87	0.91	0.89	0.905
	Speech	0.93	0.90	0.92	
PAR_NB_3	Song	0.88	0.91	0.89	0.912
	Speech	0.94	0.91	0.92	

Si può notare che l'esclusione delle due variabili estremamente asimmetriche ('PAR_NB_1') ha portato a un sensibile miglioramento delle capacità predittive del modello (+0.02 per la classe 0, +0.01 per la classe 1 e +0.016 sull'accuratezza generale). Pressoché identici i risultati della valutazione di 'PAR_NB_2', sebbene il lievissimo peggioramento di -0,004 potrebbe suggerire che la leggera asimmetria di *zero_crossings_sum* e *stft_std* non impatti negativamente sul modello. I migliori risultati in termini di accuratezza sono stati forniti da 'PAR_NB_3'.

4. https://tinyurl.com/dm1MixedNB

3.1.3 Decision tree

In ultima istanza, riportiamo i risultati più significativi di una serie di esperimenti di addestramento e valutazione dell'algoritmo *Decision tree* nell'implementazione *DecisionTreeClassifier* di Scikit-learn. Poiché esso è intrinsecamente robusto alla scala delle variabili e non fa assunzioni sulla loro distribuzione, i dati di training e di test sono stati passati all'algoritmo senza essere sottoposti ad alcun preprocessing.

Rispetto agli altri due algoritmi, *Decision tree* ha un numero maggiore di iperparametri che ne influenzano comportamento e prestazioni. Perciò, per trovarne una configurazione che portasse a buoni risultati, si è eseguita una ricerca sistematica su un'ampia gamma di valori ricorrendo a due approcci distinti: la già menzionata GridSearchCV e la randomizedSearch. Per ciascun approccio è stata definita una specifica 'param_grid', costituita dagli stessi parametri ('min_samples_split', 'min_samples_leaf' e 'max_depth'), a cui però sono stati assegnati intervalli diversi di valori, più ampi nel caso della randomizedSearch, in quanto il tipo di ricerca che essa effettua si basa su alcune combinazioni di parametri casualmente selezionate. Così come fatto in relazione ai classificatori precedenti, per ciascuna strategia si sono realizzati dieci cicli di addestramento e test. La valutazione dei migliori classificatori ottenuti è riassunta in Tabella 9.

Tabella 9. Performance dei migliori modelli ottenuti rispetto all'algoritmo Decision tree ('criterion' = 'gini').

Modello	Classe	Precision	Recall	F1-Score	Accuratezza	max_depth, min_samples_leaf, min_samples_split
GRI_DT	Song	0.89	0.88	0.89	0.908	6, 0.001, 0.01
	Speech	0.92	0.92	0.92		
RAN_DT	Song	0.90	0.87	0.89	0.909	86, 0.0045, 0.0029
	Speech	91	94	92		

Anche se il 'RAN_DT' ha restituito risultati (minimamente) migliori, la configurazione di parametri su cui è stato addestrato lo rende significativamente più complesso rispetto 'GRI_DT', che pertanto possiamo considerare il migliore in termini di compromesso tra prestazioni e complessità.

La figura 13 mostra le dieci feature più importanti dei due modelli. Emerge la preponderanza di 'length_ms', che si è rivelata di cruciale importanza nella separazione delle due classi. In entrambi i casi, infatti, 'length_ms' è la variabile scelta dall'algoritmo per effettuare il primo split (i.e., la radice dell'albero decisionale).

La seconda variabile più importante è 'disgust', che costituisce uno dei nodi di primo livello dell'albero. Si è osservato che, in fase di predizione, laddove la feature 'disgust' abbia valore 1, l'algoritmo assegna la classe 1 ('speech'). Ciò trova riscontro con quanto detto in Sez. 1.2: nel dataset nessuna registrazione cantata esprime disgusto. È una regolarità che l'algoritmo ha correttamente appreso e che è riuscito a generalizzare. Non si può dire lo stesso di 'surprised', che invece ha importanza pari a 0 per i due modelli.

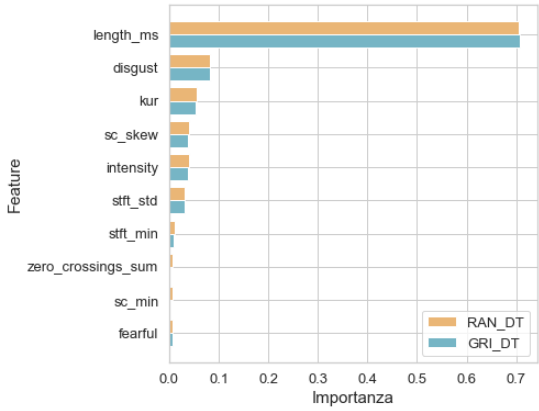


Figura 13. Le dieci features più importanti per i modelli.

3.2 Classificazione multiclasse

Per rendere il dataset idoneo all'esecuzione degli esperimenti di classificazione multiclasse, ci si è limitati a discretizzare la variabile 'emotion' nel range numerico 0-7.

3.2.1 K-NN

I medesimi approcci descritti in Sez. 3.1.1 sono stati applicati anche alla classificazione multiclasse. *KNeighborsClassifier*, infatti, è progettato per gestire in modo efficiente non solo casi di classificazione binaria, adattandosi automaticamente al numero delle classi. Per gli stessi motivi esposti in 3.1.1, prima di procedere all'addestramento dei modelli, i dati numerici sono stati scalati sia nel training set sia nel test set. Per la ricerca della migliore combinazione di iperparametri, sono state usate le stesse griglie definite per il *task* binario. Anche in contesto multiclasse, la strategia 'COM_KNN' ha raggiunto le prestazioni migliori, che riportiamo in tabella.

Tabella 10. F1-Score per classe e accuratezza generale del miglior K-NN.

Modello	angry	calm	disgust	fearful	happy	neutral	sad	surprised	Accuratezza
MUL_COM_KNN	0.57	0.60	0.43	0.46	0.38	0.28	0.39	0.49	0.465

3.2.2 Naive Bayes

Anche per quanto riguarda Naive Bayes, gli esperimenti già riferiti alla classificazione binaria sono stati riseguiti in un contesto multiclasse. Questa volta, però, non si è potuto seguire l'approccio misto descritto in Sez. 3.1.2 poiché intrinsecamente legato a problemi di tipo binario. Pertanto, sempre in ottica di mantenere la distinzione tra variabili qualitative e quantitative, si è deciso di ricorrere non a uno dei modelli di Scikit-learn, bensì all'implementazione di Naive Bayes fatta dalla libreria *mixed-naive-bayes*[5], sviluppata proprio per effettuare compiti di classificazione su dataset misti basandosi sulla logica di Naive Bayes. Di nuovo, il modello più performante (tabella 10) è quello addestrato secondo l'approccio del 'PAR_NB_3', vale a dire togliendo dai dati i valori relativi alle variabili 'stft_min', 'stft_kur' e 'sc_max'.

Tabella 11. F1-Score per classe e accuratezza generale del miglior Naive Bayes.

Modello	angry	calm	disgust	fearful	happy	neutral	sad	surprised	Accuratezza
MUL_PAR_NB_3	0.54	0.50	0.42	0.29	0.32	0.26	0.15	0.34	0.371

3.2.3 Decision tree

Anche rispetto all'algoritmo Decision tree sono state applicate al caso multiclasse le stesse tecniche viste per il caso binario, indirizzate all'individuazione della migliore combinazione di iperparametri tramite ricerca esasustiva e randomizzata. Per implementarle, si è fatto nuovamente ricorso alle griglie parametriche definite per il problema binario. Il miglior modello ('MUL_GRI_DT') è stato ottenuto tramite il primo approccio con 'entropy' come criterio di splittig, 'max_depth' di 10, 'min_samples_leaf' di 0.001 e 'min_samples_split' di 0.002. I risultati della sua valutazione sono riassunti in tabella 12.

Abbiamo anche provato a implementare due tra le strategie per l'estensione a problemi di natura multiclasse degli algoritmi di classificazione esclusivamente binaria: la 'One-vs-Rest' e la 'One-vs-One'. La prima addestra un modello per ciascuna classe, confrontandola con tutte le altre; la seconda ne costruisce uno per ogni coppia di classi. Pur consapevoli che Decision tree (come K-NN e Naive Bayes), supportando nativamente la classificazione multiclasse, non richiederebbe la loro applicazione, che comporta comunque un costo computazionale maggiore, si è condotto questo esperimento con lo scopo di testare queste strategie e i loro risultati. Pertanto, per ciascuna strategia, si sono realizzati dieci cicli addestramento-test con GridSearchCV e RandomizedSearchCV. La strategia 'OvO' è quella che in generale ha restituito le più accurate predizioni rispetto al 'ground truth'. Dal confronto dei suoi risultati (tabella 12) con quelli di 'MUL_GRI_DT', si può notare che gli F1-Score ottenuti per

5. https://pypi.org/project/mixed-naive-bayes/

le diverse classi con la strategia 'OvO' hanno una distribuzione meno disomogenea, con una differenza tra il punteggio massimo e minimo di 0.18 (contro lo 0.27 di 'MUL_GRI_DT'). Ciò potrebbe suggerire una maggiore capacità di generalizzare e gestire le classi più uniformemente.

Tabella 12. F1-Score per classe e accuratezza generale dei migliori Decision tree.

Modello	angry	calm	disgust	fearful	happy	neutral	sad	surprised	Accuratezza
MUL_GRI_DT	0.54	0.54	0.37	0.37	0.27	0.35	0.31	0.30	0.398
OVO_DT	0.50	0.48	0.39	0.40	0.32	0.37	0.34	0.46	0.404

3.3 Considerazioni finali

Tirando le fila della trattazione, si può dire che gli esperimenti di classificazione binaria hanno restituito risultati molto soddisfacenti trasversalmente rispetto agli algoritmi considerati. Va notato, però, che K-NN in tutti gli approcci seguiti (tranne 'GOW_KNN') ha superato gli altri modelli con riferimento a tutte le metriche di valutazione. In particolare, considerando l'F1-Score, si è raggiunto il 93% per la classe 0 e il 95% per la classe 1, ossia rispettivamente +4% e +3% rispetto agli altri due modelli. In termini di accuratezza, il 'COM_KNN' ha superato di 0.035 il 'RAN_DT' e di 0.027 il 'PAR_NB_3', rispettivamente i migliori Decision tree e Naive Bayes trovati. Anche nel contesto multiclasse, K-NN si è confermato il miglior modello di predizione. Questa volta, però, il distacco tra algoritmi è stato più evidente: con un'accuratezza complessiva del 46%, 'MUL_COM_KNN' ha superato di ben sei punti percentuali 'MUL_GRI_DT', che a sua volta si è distaccato da 'MUL_PAR_NB_3' di +3%.

Va detto che, a prescindere dal tipo di estimatore che si implementi, risolvere un *task* di classificazione che coinvolga otto classi è un problema estremamente complesso. Per questo motivo, abbiamo provato a ridimensionare il *task* riducendo il numero delle classi raggruppando le otto emozioni in tre classi, 'negative' ('angry', 'disgust', 'fearful', 'sad'), 'neutral' ('neutral', 'surprised'[6] e 'positive' ('happy', 'calm'). Apportate le dovute modifiche ai dati di training e test, si sono addestrati e testati i tre algoritmi, implementando per ognuno la strategia che nel problema a otto classi aveva dato i risultati migliori. Dei tre, 'COM_KNN' si è confermato il miglior estimatore, raggiungendo l'accuratezza di 0.61 e l'F1-Score di 0.71 per la classe 'negative', 0.49 per 'neutral' e 0.48 per 'positive'. L'analisi ha risentito del notevole sbilanciamento delle classi (1320 'negative', 380 'neutral' e 752 'positive'). Il successivo tentativo di livellare i supporti dei tre gruppi applicando la tecnica di undersampling ha, sì, bilanciato i dati, ma ne ha ridotto significativamente il numero (380 per classe, per un totale di 1140 record), rendendo qualsiasi analisi poco significativa.

4 Regressione

Si è deciso di utilizzare la tecnica di regressione per la creazione di un modello in grado di sostituire i missing values trovati per le variabili numeriche continue 'intensity' ed 'sc_min', scegliendo come modello quello con le performance migliori tra Linear, Ridge e Lasso. A tale scopo abbiamo pre-processato i dati con gli stessi metodi usati per i task di clustering e classificatione, ovvero eliminando le feature per alta correlazione, correggendo quelle asimmetriche, standardizzando il dataset ed escludendo ovviamente le variabili target da predire. Per ciascuna variabile target, abbiamo considerato il dataset privo delle righe relative ai valori mancanti, così da poter suddividere lo stesso in train e test e calcolare le principali misure di performance (R^2, MSE e MAE) di ciascun regressore utilizzato. In particolare, è stato ritenuto utile l'utilizzo di una grid-search per trovare il valore ottimale di alfa per i regressori Ridge e Lasso, come parametro di regolarizzazione di ciascun modello.

La prima sostituzione effettuata riguarda la variabile 'sc_min': sono state escluse 1021 righe del dataset

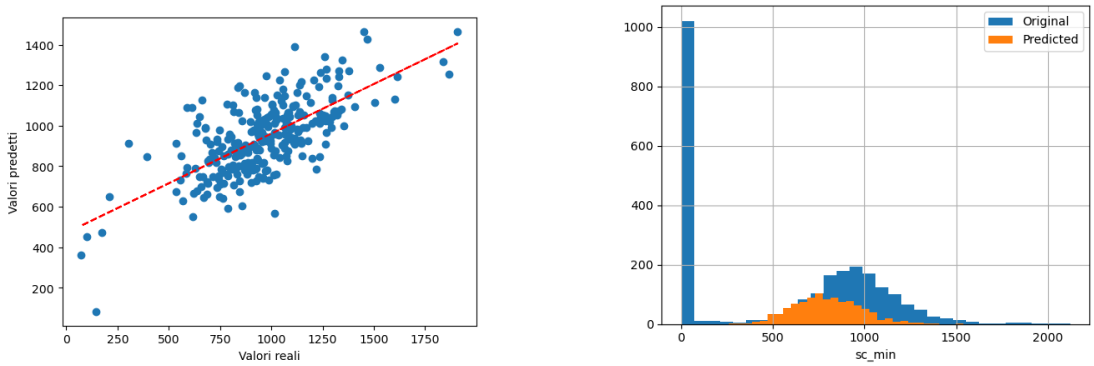
6. La scelta di inserire la sorpresa tra le emozioni neutre scaturisce dalla lettura di alcuni articoli di carattere psicologico che la descrivono come *un'emozione di per sé neutra: semmai è quello che avviene successivamente a connotare l'esperienza in senso positivo o negativo* (fonte: https://tinyurl.com/surpriseNeutral2)

contenenti valori uguali a 0, e le 1431 osservazioni rimanente sono state divise in dataset di train e test. I valori riportati in Tabella 13 mostrano come i tre modelli raggiungano risultati molto simili tra loro; si può comunque notare che il modello Ridge raggiunga un più alto valore di R^2 e valori più bassi di MSE e MAE, e per tal motivo è stato scelto come modello per la sostituzione dei missing values per la variabile 'sc_min'.

Tabella 13. Mean squared error, Mean absolute error e R^2 di ciascun modello (variabile 'sc_min').

Modello	R-squared	MSE	MAE
Linear	0.469	35073.222	142.704
Ridge	0.469	35057.212	142.481
Lasso	0.467	35181.876	142.141

I seguenti grafici mostrano il risultato derivante dall'utilizzo del modello Ridge: dalla figura 14a è possibile confrontare tramite scatterplot i valori reali con quelli predetti sul test set e vedere come questi si allineano. A conferma dei valori riportati in tabella, essendo i punti non perfettamente disposti intorno alla linea di riferimento, è possibile affermare che il modello di regressione non raggiunga ottimi risultati. Il secondo grafico in figura 14b mostra, invece, la nuova distribuzione (in arancione) dei soli 1021 valori predetti tramite regressione, in sostituzione a quelli aventi valore 0 (colonna in blu).



(a) Scatterplot modello Ridge.

(b) sc_min.

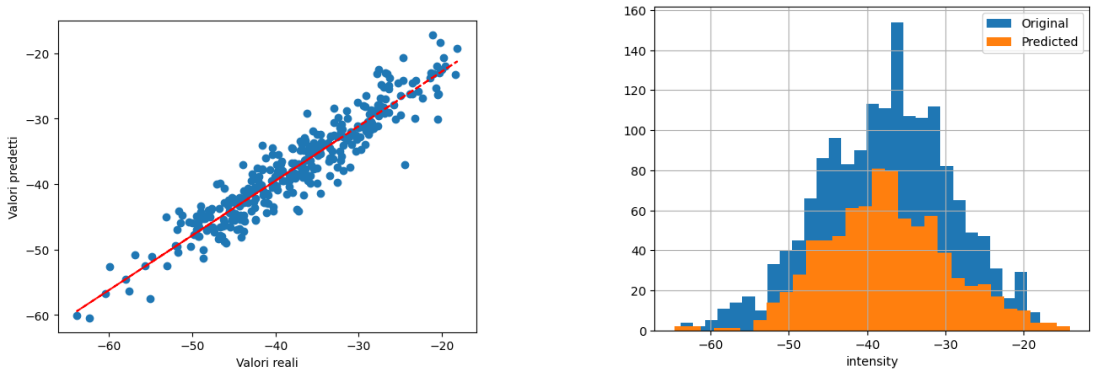
Figura 14. Performance del modello Ridge per la variabile 'sc_min'.

Lo stesso approccio è stato utilizzato per la variabile numerica 'intensity': sono state confrontate le performance dei tre modelli di regressione suddividendo in train e test il dataset privato delle righe relative agli 816 valori mancanti. Dalla Tabella 14 emerge una notevole somiglianza tra i risultati dei regressori Lineare e Ridge, entrambi capaci di ottenere risultati significativamente superiori rispetto al modello Lasso. Tra i due si è scelto di impiegare il regressore Lineare per la sostituzione finale dei valori, poiché riporta valori di MSE e MAE inferiori, ed offre un'elevata precisione, come mostrato chiaramente dallo scatterplot riportato in figura 15. Infatti, i punti si dispongono in prossimità della linea di previsione, confermando l'accuratezza del modello scelto.

Tabella 14. Mean squared error, Mean absolute error e R^2 di ciascun modello (variabile *intensity*).

Modello	R-squared	MSE	MAE
Linear	0.882	9.403	2.423
Ridge	0.882	9.411	2.426
Lasso	0.771	18.196	3.413

Sulla base delle analisi effettuate è possibile dedurre che l'approccio di classificazione tramite regressione non produce risultati ottimali per la variabile numerica 'sc_min', mentre si rivela estremamente efficace per la

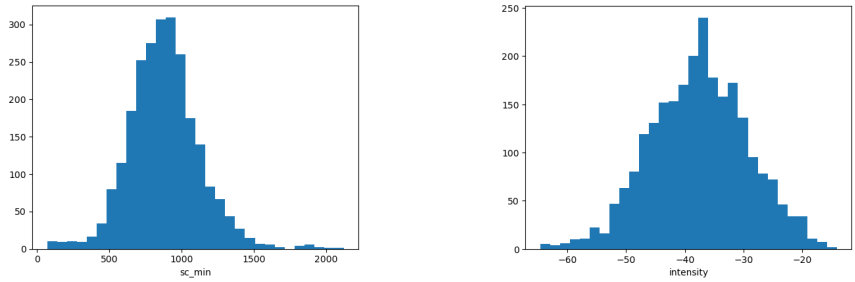


(a) Scatterplot modello Lineare.

(b) intensity.

Figura 15. Performance del modello Linear per la variabile 'intensity'.

variabile 'intensity'. Si riportano le distribuzioni finali delle due variabili in seguito alle sostituzioni definitive dei valori mancanti (Figura 16).



(a)

(b)

Figura 16. Distribuzione delle variabili sc_min(a) e intensity(b) dopo la sostituzione.

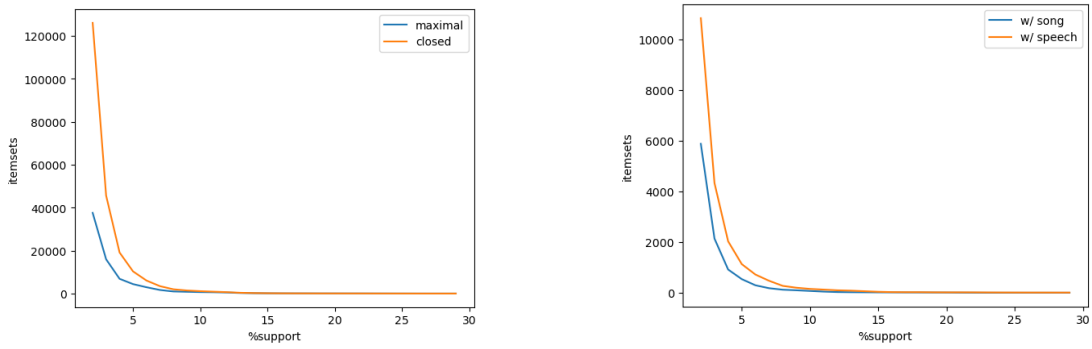
5 Pattern Mining

In quest'ultima sezione si discutono i risultati più interessanti dell'applicazione al dataset di tecniche di Pattern Mining e di analisi delle associazioni. A tale scopo è stato utilizzato il dataset ottenuto in seguito alle operazioni di data preparation descritte in Sez. 1 per garantire l'accuratezza e l'efficacia dell'estrazione dei pattern dai dati. Per prima cosa sono state discretizzate tutte le feature continue, dividendole in 4 bins, ciascuno contenente un numero uguale di osservazioni, dal momento che gli algoritmi richiedono dati discretizzati per poter essere correttamente implementati. Inoltre, si è deciso di escludere dall'analisi la variabile 'stft_min' che, avendo 1016 osservazioni con valore 0, se discretizzata, avrebbe prodotto dei bins identici, distorcendo i risultati dell'analisi. Allo stesso modo si è deciso di escludere anche la variabile categorica 'channels', in quanto la sua distribuzione altamente sbilanciata tra le classi e la sua conseguente presenza in quasi tutte le transazioni può influire negativamente sui risultati e non essere molto informativa per l'analisi delle associazioni. Al termine della fase di preparazione del dataset, e dopo aver apportato le opportune modifiche anche alle variabili categoriche, le osservazione appaiono nel seguente modo (esempio riga 100):

```
['speech', 'sad', 'normal', 'Dogs...the door', 'rep2', 'F', '(2935.999, 3604.0]_length',
'(4720.999, 10362.5]_zcs',..., '(1.875, 2.285]_kur', '(-2.358, -0.337]_skew']
```

5.1 Pattern frequenti

Per trovare i pattern frequenti nei nostri dati, è stato applicato il metodo Apriori dalla libreria *pyfim*, che ha rivelato che '(normal, speech)' è l'itemset più frequente con un valore di support pari al 31.28%. Si tratta di un risultato ragionevole in quanto 'normal' e 'speech' sono tra i valori più comuni nel dataset. Impostando invece il numero minimo di elementi per ogni itemset a 3, i più frequenti risultano essere composti da '(0.209, 0.318]_stft_std, M, speech)' e '(0.531, 0.724]_stft_mean, M, speech)', entrambi con valori di support pari a circa 20%. Si può notare la presenza comune delle classi M e speech rispettivamente degli attributi 'sex' e 'vocal_channel'. La Figura 17a mostra la variazione del numero di closed e maximal itemsets identificati per un aumento di valore di support in un range che va da 2 a 30. In particolare, si nota come la loro convergenza avvenga ad un valore di support pari a circa il 15%. È possibile applicare un ragionamento simile alla Figura 17b, ma questa volta in relazione agli itemset che contengono una classe dell'attributo 'vocal_channel', che abbiamo scelto per eseguire esperimenti di classificazione tramite generazione di regole di associazione (Sez. 5.3).



(a) (b)

Figura 17. Variazione del numero di closed e maximal itemsets (a), e del numero di itemset contenenti song e speech (b) per diversi livelli di support.

5.2 Regole di associazione

Con riferimento all'estrazione delle regole di associazione più rilevanti si è deciso di utilizzare 15 come soglia di supporto minimo e 70 come valore minimo di confidenza, con un numero minimo di 3 elementi per itemset cosi da evitare l'identificazione di regole banali. Impostando tali parametri otteniamo 114 regole. Il grafico in Figura 18 riporta il numero di itemset per ogni combinazione di confindence e support di un determinato range. Si evince che, utilizzando una soglia di confidenza più alta, il numero di regole trovate diminuirebbe e ciò potrebbe limitare la scoperta di regole interessanti. Dalla visualizzazione grafica dei valori di confidenza e di lift per tutte le regole menzionate prima (Figura 19a), notiamo che poche altre regole hanno un'alta confidenza dopo il picco raggiunto intorno al valore di 0.75. Per quanto riguarda la figura 19b, si possono fare alcune considerazioni: in primo luogo, non ci sono regole corrispondenti a valori inferiori a uno, il che indica assenza di associazioni negative o deboli tra l'antecedente e il conseguente; al contrario, sono presenti picchi significativi in corrispondenza dei valori di lift pari a 1.5 e 3, evidenza che potrebbe indicare l'esistenza di regole di associazione in cui antecedente e conseguente sono leggermente correlati nel primo caso, e hanno una forte correlazione nel secondo. Pertanto, essendo tali picchi riferiti solo ad alcune aree specifiche del grafico, è facile

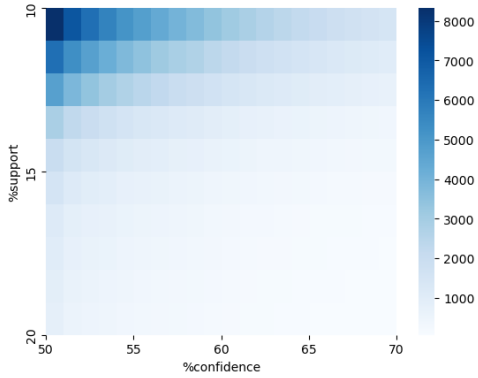
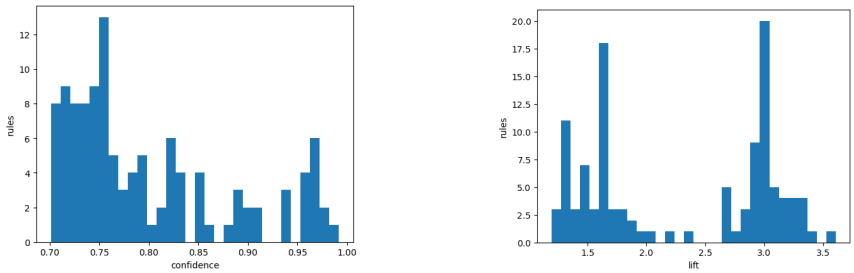


Figura 18. Variazione del numero di AR rispetto ai parametri di confidenza e supporto.

intuire che le regole di associazione non sono distribuite in modo uniforme in tutto il range di valori di lift.



(a) (b)

Figura 19. Regole di associazione al variare dei livelli di confidenza (a) e lift (b).

In Tabella 15 sono riportate le cinque regole che otteniamo con più alto valore di lift, scelto come misura di identificazione per le regole di associazione più significative.

Tabella 15. Regole di associazione con più alto lift (z-min = 3, conf = 70, supp = 15).

Conseguente	Antecedente	%_Support	Confidence	Lift
(0.209, 0.318]_stft_std	(-0.659, 0.902]_stft_kur, M	13.866	0.902	3.607
(0.557, 1.825]_sc_skew	(-0.202, 1.697]_sc_kur, (2360.8797, 4563.685]_sc_mean	13.662	0.855	3.418
(0.209, 0.318]_stft_std	((0.531, 0.724]_stft_mean, M, speech)	16.639	0.829	3.317
(0.209, 0.318]_stft_std	(-0.659, 0.902]_stft_kur, speech	13.254	0.829	3.316
(0.531, 0.724]_stft_mean	(2935.999, 3604.0]_lenght, M, speech	12.602	0.828	3.314

La maggior parte delle regole trovate ha come elemento conseguente l'item '(0.209, 0.318]_stft_std'; ciò consente di affermare, come visto precedetentemente in Sez. 5.1 che le osservazioni riferibili a registrazioni parlate da un soggetto maschile sono fortemente correlate con più bassi di valori 'stft_std'.

5.3 Classificazione tramite regole di associazione

Come obiettivo di classificazione tramite regole di associazione, sono state utilizzate le regole aventi le classi della variabile 'vocal_channel' come elemento conseguente. La scelta è stata determinata dal fatto che si tratta della stessa variabile utilizzata come target negli esperimenti di classificazione binaria descritti in Sez. 3.1, in modo da confrontare i risultati ottenuti tramite i due approcci. Per fare ciò, il dataset è stato partizionato in in dati di training e di test, estraendo dai primi tutte le regole che avessero 3, 70 e 15 come valori associati rispettivamente ai parametri 'z_min'[7], 'confidence' e 'support'. In seguito, sono state filtrate le sole regole di associazione che coinvolgevano la variabile target 'vocal channel', considerando maggiormente rilevanti quelle aventi maggior valore di lift. Tali regole sono state utilizzate per fare predizioni sul test set, e, confrontando queste ultime con il ground truth, è stato possibile valutarne l'accuratezza (Tabella 16).

Tabella 16. Regole di associazione relative a 'song' e 'speech' con più alto valore di accuracy.

Conseguente	Antecedente	%_Support	Confidence	Lift	Accuracy
song	(4538.0, 6373.0]_lenght,(0.563, 1.875]_kur	14.533	0.966	2.368	0.71
song	(0.563, 1.875]_kur, normal	14.329	0.915	2.244	0.69
speech	(0.531, 0.724]_stft_mean, (-0.511, 0.0985]_sc_skew	15.859	0.969	1.636	0.58
speech	(-0.511, 0.0985]_sc_skew, (2935.999, 3604.0]_lenght	15.043	0.967	1.634	0.58

7. Il parametro 'z_min' si riferisce al numero minimo di elementi di un itemset.

Basandosi sulla sola accuratezza, le regole che hanno come conseguente la classe 'song' hanno capacità predittive migliori di quelle che hanno 'speech'. Il più alto valore di accuratezza (71%) è raggiunto tramite la regola di associazione: '(4538.0, 6373.0]_lenght', '(0.563, 1.875]_kur') → song'.

Questi risultati sono in linea con le analisi in Sez. 3.1 specialmente in riferimento all'importanza dalla variabile 'length_ms' nel *task* predittivo. Si è già sottolineato, nel contesto degli alberi decisionali, il suo ruolo fondamentale nella separazione delle classi. Adesso si evince la sua rilevanza anche nella definizione delle regole associative più significative. Tuttavia, nessuna regola di associazione è riuscita a eguagliare le prestazioni di nessuno dei migliori classificatori addestrati precedentemente.