

Progetto Data Mining

Andrea Carnevale
Federico Canepuzzi
Gloria Segurini

FMA: A Dataset For Music Analysis

Data Science & Business Informatics
Università di Pisa
Anno Accademico 2020/2021

Contents

1 Modulo 1	1
1.1 Introduzione	1
1.1.1 Tracks	1
1.1.2 Genres	1
1.1.3 Echonest	2
1.1.4 Features	2
1.1.5 Scelta del label e selezione degli attributi	3
1.2 Classification	4
1.2.1 Decision Tree	4
1.2.2 KNN	5
1.3 Anomaly Detection	6
1.3.1 Box Plot	6
1.3.2 DbScan	7
1.3.3 Isolation Forest	7
1.3.4 ABOD	7
1.3.5 Confronto tra i diversi metodi	8
1.4 Imbalanced Learning	9
1.5 Analisi dei risultati	11
2 Modulo 2	12
2.1 Naive Bayes	12
2.2 Logistic Regression	12
2.3 Rule Based	13
2.4 SVM	13
2.5 Neural Network	14
2.6 Random Forest	15
2.7 AdaBoost	16
2.8 Confronto Classificatori	16
2.9 Regression Problem	17
2.9.1 Simple Linear Regression	17
2.9.2 Ridge e Lasso con Simple Linear Regression	18
2.9.3 Multiple Linear Regression	18
3 Modulo 3	19
3.1 Time Series Selection	19
3.2 Time Series Clustering	19
3.2.1 Partitional Clustering	20
3.2.2 Hierarchical Clustering	21
3.2.3 Features-based Clustering	21
3.3 Motifs and Anomalies	22
3.4 Time Series Classification	24
4 Modulo 4	27
4.1 Sequential Pattern Mining	27
4.2 Advanced Clustering	28
4.2.1 K-Means vs X-Means	28
4.2.2 DbScan vs Optics	28
4.3 Transactional Clustering	29
4.3.1 K-Modes	29

Modulo 1

1.1 Introduzione

L'analisi del presente progetto è stata svolta con riferimento al Free Music Archive (FMA), un dataset, liberamente accessibile, che mette a disposizione degli utenti 106.574 tracce audio realizzate da 16.341 artisti, appartenenti a 14.854 diversi album ed associate ad una gerarchia tassonomica di 163 diversi generi musicali. I dati sono strutturati in 4 sub-datasets chiamati rispettivamente 'tracks', 'genres', 'features' ed 'echonest' (tabella 1.1). Il dataset 'genres' ha come oggetto principale i vari generi musicali che caratterizzano le tracce audio ed associa ad ogni record un diverso 'genres_id'. Gli altri tre datasets invece hanno tutti come oggetto di analisi le tracce audio ed identificano ogni record con un diverso 'track_id'; grazie a questo identificatore è possibile unire queste ultime 3 tabelle.

	Tracks	Genres	Features	Echonest
#Records	106574	163	106574	13129
#Attributes	52	4	518	249

Table 1.1: Tabella descrittiva dei 4 sub-dataset presenti in FMA

1.1.1 Tracks

In questo dataset sono riportati i metadati che caratterizzano ogni traccia audio. Sono presenti 52 attributi organizzati in un multi index gerarchico a due livelli; al livello superiore sono presenti 4 indici: 'track', 'artist', 'album' e 'set' (tabella 1.2). Sotto l'indice 'track' sono presenti due attributi molto importanti per la successiva analisi: 'genres' e 'genre_top'. Ad ogni traccia musicale sono infatti associati uno o più generi che vengono riportati nella colonna 'genres'. E' presente una gerarchia tassonomica dei generi che permette, a partire dai valori di 'genres', di stabilire il 'genre_top' e cioè il genere alla radice della gerarchia. Gli autori hanno scelto di lasciare un valore nullo nel caso di un numero di generi top maggiore di uno.

Livello 0	track	artist	album	set
Livello 1	20 colonne	13 colonne	17 colonne	2 colonne
Significato	Attributi riferiti alla traccia audio	Attributi riferiti all'artista della traccia	Attributi riferiti all'album di cui fa parte la traccia	Attributi utilizzati per le precedenti analisi sul dataset eseguite dagli autori

Table 1.2: Tabella descrittiva del dataset Tracks

1.1.2 Genres

In 'Genres' sono presenti 163 record, ognuno rappresentante un diverso genere musicale e 4 attributi: 'tracks', 'parent', 'title', 'topLevel'. Attraverso questo dataset è possibile ricostruire la gerarchia tassonomica dei vari generi, con l'attributo 'parent' che indica il genere superiore e l'attributo 'topLevel' che indica invece il genere top alla radice dell'albero. In totale sono presenti 16 differenti generi top. Di seguito è riportata come esempio la struttura gerarchica del genere top 'Jazz' (figura 1.1)

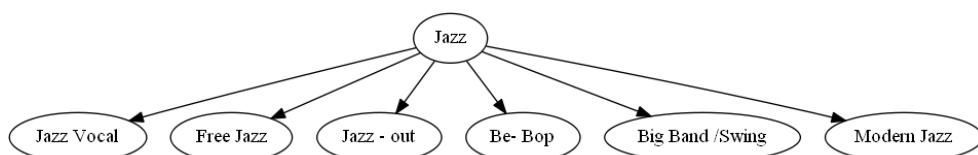


Figure 1.1: Albero del genere top Jazz

1.1.3 Echonest

Anche questo dataset presenta su ogni record un diverso track_id rappresentante una traccia audio. Diversamente da ‘Tracks’ e da ‘Features’, qui il numero di tracce è inferiore ed è pari a soli 13.129 record. Sono presenti 249 attributi organizzati in un multi index gerarchico a due livelli; al livello superiore sono presenti 5 indici: ‘audio_features’, ‘metadata’, ‘ranks’, ‘social_features’ e ‘temporal_features’ (tabella 1.3).

Livello 0	audio_features	metadata	ranks	social.features	temporal.features
Livello 1	8 colonne	7 colonne	5 colonne	5 colonne	224 colonne
Significato	Caratteristiche della traccia audio	Metadati della traccia audio	Classifiche varie della traccia audio e del suo artista	Successo sociale della traccia audio e del suo artista	Attributi temporali associati alla traccia audio

Table 1.3: Tabella descrittiva del dataset Echonest

1.1.4 Features

In questo dataset sono riportate alcune caratteristiche della traccia audio ottenute attraverso la libreria ”Librosa”. Sono presenti 518 attributi organizzati in un multi index gerarchico a tre livelli: dall’alto verso il basso della gerarchia sono presenti gli indici ‘feature’ -> ‘statistics’ -> ‘number’ (tabella 1.4). Ogni colonna è quindi identificata da una tupla (‘feature’,‘statistics’,‘number’). Le feature, calcolate mediante l’utilizzo della libreria librosa, sono le seguenti:

- Chroma features -> chroma_cens, chroma_cqt, chroma_stft.
Queste feature sono estratte da un Chromagram, che rappresenta la quantità di energia presente nel segnale per ogni semitonino distinto.
- Mel-Frequency Cepstral Coefficients -> mfcc.
Un coefficiente che rappresenta le caratteristiche della voce umana.
- Zero-crossing rate -> zcr.
Rappresenta il numero di volte in cui il segnale audio cambia da positivo a negativo in un dato lasso temporale.
- Spectral features -> rmse, spectral_bandwidth, spectral_centroid, spectral_contrast, spectral_rolloff.
Caratteristiche relative allo spettrogramma della traccia audio, con il quale è possibile rappresentare l’intensità del suono (in termini di decibel) rispetto alla frequenza ed il tempo.
- Tonal centroid features -> tonnetz.
Rappresenta l’armonia, quindi l’arrangiamento di note nella traccia audio.

Le funzioni di Librosa restituiscono per ogni feature una matrice i cui i ‘number’ identificano le righe. Su ogni riga vengono calcolate le statistiche (7) tra cui, oltre alle comuni max, min, median, mean, std, vi sono anche kurtosis, che data una distribuzione ne indica il grado di appiattimento e skew, il quale indica se la distribuzione è simmetrica rispetto ad un punto.

Ad esempio, per le chroma_cqt viene restituita una matrice con shape (12,t) dove ogni riga corrisponde ad un semitonino musicale di cui abbiamo t valori temporali.

Quindi chroma_cqt ha 12 number (da 01 a 12), e per ogni number 7 statistiche per un totale di 84 attributi. In (‘chroma_cqt’,‘max’,‘01’) ci sarà il massimo della prima riga della matrice, in (‘chroma_cqt’,‘min’,‘01’) il minimo e così via per ogni statistica e per ogni number.

Features	chroma cens	chroma cqt	chroma stft	mfcc	rmse	spectral bandwidth	spectral centroid	spectral contrast	spectral rolloff	tonnetz	zcr
Statistics	7	7	7	7	7	7	7	7	7	7	7
Number	12	12	12	20	1	1	1	7	1	6	1
Totale	84	84	84	140	7	7	7	49	7	42	7

Table 1.4: Tabella descrittiva del dataset Features. Ad ogni features sono associate 7 statistiche; ad ogni statistica un numero di colonne che dipende da features

1.1.5 Scelta del label e selezione degli attributi

Per questo Task è stato scelto come label l'attributo ‘genre_top’. Come già detto, questo attributo presenta valori nulli sulle tracce audio caratterizzate da generi appartenenti ad alberi diversi della gerarchia. In totale 56976 presentano un valore nullo su ‘genre_top’. Delle restanti 49598 tracce audio, ne sono state selezionate 25000; la selezione è stata eseguita in accordo alle scelte fatte dagli autori del dataset FMA, che nella loro analisi avevano scelto le tracce in base alla qualità dei dati a disposizione: più null value sono infatti sintomo di una peggior qualità generale delle informazioni a disposizione per la traccia audio.

Questa riduzione a 25000 record è stata necessaria per abbassare il tempo computazionale necessario per le analisi che saranno svolte. Il numero di tracce audio per ogni genere è riportato in tabella 1.5. Come si può osservare dalla tabella, lo studio che andremo a condurre sarà basato su un multiclass label (16 generi) e su un dataset sbilanciato.

Per quanto riguarda gli attributi, lo studio si è concentrato sul dataset ‘features’ e sui suoi 518 attributi: l’obiettivo è stato quello di cercare di classificare ogni traccia audio con un genere musicale generico (‘genre_top’), utilizzando le caratteristiche della traccia stessa ottenute attraverso la libreria ‘Librosa’. Un classificatore di questo tipo permetterebbe di riconoscere il genere senza l’esigenza di alcun dato accessorio, ad esempio i metadata, bensì soltanto riproducendo la traccia.

I 518 attributi non presentavano null values ed inizialmente tutti potevano avere la stessa importanza relativa nella classificazione. Non potendo però lavorare con una tale dimensionalità, è stato deciso di selezionare i migliori per l’analisi in due fasi:

- è stata inizialmente calcolata una matrice di correlazione di Pearson tra i vari attributi e, per ogni coppia di essi che presentava una correlazione maggiore di 0.6 (in valore assoluto), è stato eliminato uno dei due. Questa prima fase ha ridotto il numero di attributi a 191;
- nella seconda è stato utilizzato un algoritmo di features selection, ‘SelectKBest’, che, utilizzando il class label, va ad eseguire un’analisi statistica su ogni attributo singolarmente (ANOVA, Analysis of Variance) e restituisce uno score, tanto maggiore quanto più diverseificate sono le distribuzioni delle classi in base all’attributo. Questo ha permesso di andare a selezionare i migliori 20 attributi con cui condurre lo studio. In tabella 1.6 sono stati riportati i 20 attributi selezionati con lo score ottenuto dall’algoritmo ‘SelectKBest’.

Genre_Top	#Record
Rock	7103
Electronic	6314
Experimental	2251
Hip-Hop	2201
Folk	1519
Instrumental	1350
Pop	1186
International	1018
Classical	619
Old-Time / Historic	510
Jazz	384
Country	178
Soul-RnB	154
Spoken	118
Blues	74
Easy Listening	21

Table 1.5: tabella dove è riportato per ogni genere il numero di record associato

Rank	Livello 0	Livello 1	Livello 2	Score
1	mfcc	max	04	858.7
2	mfcc	max	01	768.2
3	mfcc	max	03	416.8
4	mfcc	max	07	384.3
5	spectral bandwidth	kurtosis	01	357.2
6	mfcc	mean	02	340.1
7	chroma stft	mean	01	290.4
8	mfcc	std	04	281.8
9	mfcc	std	02	280.4
10	spectral contrast	max	04	267.9

Rank	Livello 0	Livello 1	Livello 2	Score
11	mfcc	min	04	265.6
12	mfcc	std	01	258.5
13	mfcc	skew	04	247.3
14	mfcc	min	05	247.2
15	spectral contrast	max	03	240.4
16	mfcc	min	02	234.1
17	mfcc	max	11	222.0
18	tonnetz	std	01	218.4
19	spectral contrast	min	02	216.4
20	mfcc	min	11	208.4

Table 1.6: classifica dei 20 attributi selezionati dall'algoritmo SelectKBest

1.2 Classification

In questa sezione vengono analizzati i risultati ottenuti utilizzando il Decision Tree e il KNN al fine di classificare il genre_top dei brani presenti nel dataset.

Dopo aver ristretto gli attributi ad un subset di 20, il dataset è stato spartito in training set e test set con proporzione 0.70/0.30. Nella fase di model selection è stata utilizzata la Grid Search in combinazione con la Stratified Cross-validation sul training set (in modo da mantenere le proporzioni dei 16 generi nei diversi fold) e ciò ha permesso di ottenere i migliori iperparametri per il modello. Il test set è stato utilizzato nella fase di assessment per valutare le performance e confrontare tra loro i diversi modelli ottenuti. Lo score utilizzato per il parameter tuning è l' F1-macro, più adatta ai dataset sbilanciati in quanto prende in considerazione sia la precision che la recall. Lo score globale viene calcolato come la media non pesata (macro average) tra le F1 score di ogni label/genere.

1.2.1 Decision Tree

Per il Decision Tree sono stati provati i parametri nei range specificati in tabella 1.7, dove si trova inoltre la miglior combinazione identificata in termini di F1-score.

Param	Range	Best
max_depth	(8,20)	18
min_sample_leaf	(30,80)	30
min_sample_split	(30,80)	40
criterion	[gini, entropy]	entropy

	Precision	Recall	F1	Accuracy
Train	0.35	0.31	0.31	0.55
Val	0.25	0.25	0.24	0.48
Test	0.27	0.26	0.25	0.48

Table 1.7: Griglia parametri

Table 1.8: Performance Decision Tree

In tabella 1.8 sono riportate le performances del classificatore sul training set, sul validation set con l'utilizzo della cross validation e sul test set.

Analizzando la feature importance del classificatore trovato si può notare in figura 1.2 che tutti gli attributi precedentemente selezionati tramite il SelectKBest vengono utilizzati per la costruzione dell'albero, nonostante alcuni abbiano meno importanza.

Di seguito vengono analizzati nel dettaglio i risultati ottenuti sul test set, considerando singolarmente le diverse labels. Come si può notare, lo score differisce di molto da una label all'altra. Infatti, generi come Old-Time/ Historic, Rock, Electronic e Classical presentano un score F1 alto, tenendo conto della numerosità delle

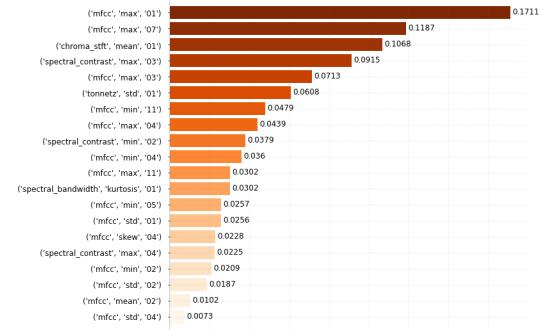


Figure 1.2: Feature Importance

possibili label da assegnare. La situazione è ben diversa per i generi evidenziati in rosso che sono caratterizzati da F1 prossimo o uguale a 0.00. Il risultato pessimo può essere in alcuni casi ricondotto ad una support bassa (in proporzione uguale a quella del training set). Non si può dire la stessa cosa per il genere 'Pop', che ha una support nella media. Tuttavia, trattandosi di un genere molto vario è probabile che la sua classificazione risulti difficoltosa.

	precision	recall	f1-score	auc	support
Blues	0.00	0.00	0.00	0.65	22
Classical	0.44	0.47	0.46	0.91	186
Country	0.00	0.00	0.00	0.63	54
Easy Listening	0.00	0.00	0.00	0.57	6
Electronic	0.49	0.65	0.56	0.78	1894
Experimental	0.26	0.10	0.15	0.64	675
Folk	0.37	0.36	0.36	0.80	456
Hip-Hop	0.43	0.35	0.39	0.79	660
Instrumental	0.27	0.22	0.25	0.79	405
International	0.24	0.15	0.19	0.72	305
Jazz	0.12	0.01	0.02	0.68	115
Old-Time / Historic	0.80	0.88	0.83	0.97	153
Pop	0.10	0.02	0.03	0.60	356
Rock	0.55	0.72	0.62	0.82	2131
Soul-RnB	0.00	0.00	0.00	0.60	46
Spoken	0.26	0.19	0.22	0.75	36
accuracy				0.48	7500
macro avg	0.27	0.26	0.25	0.73	7500

Table 1.9: Label performance DT

TRUE LABEL	PREDICTED LABEL															
	Blues	Classical	Country	Easy Listening	Electronic	Experimental	Folk	Hip-Hop	Instrumental	International	Jazz	Old-Time / Historic	Pop	Rock	SoulRnB	Spoken
Blues	0	2	0	0	3	2	3	1	0	3	0	0	0	8	0	0
Classical	0	88	0	0	21	15	6	0	38	6	0	1	2	9	0	0
Country	0	0	0	0	8	2	6	2	1	2	0	0	1	32	0	0
Easy Listening	0	0	0	0	4	1	0	0	0	0	0	0	0	0	1	0
Electronic	0	20	0	0	1222	65	44	120	73	13	1	4	7	324	0	1
Experimental	0	29	0	0	203	70	34	26	50	28	0	12	9	207	0	7
Folk	0	9	0	0	35	5	163	26	16	30	2	2	15	153	0	0
Hip-Hop	0	0	0	0	270	9	13	232	4	10	1	0	4	115	0	2
Instrumental	0	28	0	0	98	36	26	3	90	4	2	5	2	108	0	3
International	0	5	0	0	91	12	24	29	7	46	0	0	5	83	0	3
Jazz	0	7	0	0	37	2	9	2	18	5	1	0	2	32	0	0
Old-Time / Historic	0	2	0	0	2	6	0	0	0	2	0	0	134	0	7	0
Pop	0	3	0	0	98	14	31	28	4	10	0	3	7	155	0	3
Rock	0	5	0	0	335	30	80	72	26	31	1	6	16	1528	0	1
SoulRnB	0	0	0	0	30	0	0	1	1	1	0	0	1	12	0	0
Spoken	0	0	0	0	14	4	4	2	0	1	0	1	0	3	0	7

Figure 1.3: Confusion Matrix DT

1.2.2 KNN

Per l'algoritmo K-Nearest Neighbor è stata inizialmente eseguita una fase di normalizzazione con lo Standard Scaler, reputato adatto sulla base delle distribuzioni gaussiane delle variabili.

Anche in questo caso sono stati presi in considerazione i 20 attributi ottenuti dalla feature selection. Sebbene l'algoritmo lavori con le distanze e sia pertanto soggetto alla ‘Curse of dimensionality’, in termini di risultati esso non risentiva dell’alta dimensionalità; anzi, si è notato un peggioramento dello score diminuendo gli attributi.

Param	Range	Best
weghts	[distance, uniform]	distance
p	[1, 2]	1
n_neighbors	(5,20)	6
criterion	[gini, entropy]	entropy

Table 1.10: Griglia parametri

	Precision	Recall	F1	Accuracy
Val	0.40	0.35	0.36	0.55
Test	0.41	0.36	0.36	0.55

Table 1.11: Performance KNN

Con il KNN si ottengono risultati sensibilmente migliori. Si hanno infatti una precision e recall più alte per tutte le labels, e, di conseguenza, maggior F1, mentre non si nota un miglioramento per i generi Easy Listening e Soul-RnB.

	precision	recall	f1-score	auc	support
Blues	0.40	0.09	0.15	0.67	22
Classical	0.50	0.70	0.58	0.93	186
Country	0.33	0.15	0.21	0.76	54
Easy Listening	0.00	0.00	0.00	0.58	6
Electronic	0.60	0.61	0.60	0.82	1894
Experimental	0.37	0.13	0.19	0.66	675
Folk	0.48	0.50	0.49	0.83	456
Hip-Hop	0.49	0.48	0.48	0.83	660
Instrumental	0.40	0.34	0.37	0.77	405
International	0.44	0.31	0.36	0.79	305
Jazz	0.35	0.17	0.22	0.70	115
Old-Time / Historic	0.85	0.97	0.91	1.00	153
Pop	0.21	0.08	0.11	0.60	356
Rock	0.58	0.81	0.68	0.87	2131
Soul-RnB	0.12	0.02	0.04	0.65	46
Spoken	0.48	0.33	0.39	0.84	36
accuracy				0.55	7500
macro avg	0.41	0.36	0.36	0.77	7500

	Blues	Classical	Country	Easy Listening	Electronic	Experimental	Folk	Hip-Hop	Instrumental	International	Jazz	Old-Time / Historic	Pop	Rock	Soul-RnB	Spoken
TRUE LABEL	2	0	0	0	2	0	2	2	2	1	0	0	0	11	0	0
PREDICTED LABEL	Blues	Classical	Country	Easy Listening	Electronic	Experimental	Folk	Hip-Hop	Instrumental	International	Jazz	Old-Time / Historic	Pop	Rock	Soul-RnB	Spoken
Blues	2	0	0	0	2	0	2	2	2	1	0	0	0	11	0	0
Classical	0	130	0	0	4	5	10	1	17	3	3	1	0	12	0	0
Country	0	0	8	0	0	1	7	4	0	1	0	0	2	31	0	0
Easy Listening	0	0	0	0	1	0	0	3	1	0	0	0	0	1	0	0
Electronic	0	23	1	0	1136	49	40	149	58	32	9	3	28	343	3	0
Experimental	0	39	1	0	178	87	26	30	52	14	2	10	9	221	2	4
Folk	1	7	1	0	20	12	229	11	20	9	3	3	14	125	0	1
Hip-Hop	1	0	1	0	183	2	11	315	5	9	0	1	8	121	1	2
Instrumental	0	38	0	0	51	24	26	7	138	12	11	4	10	84	0	0
International	0	3	4	0	55	4	16	34	2	94	1	0	5	83	1	3
Jazz	0	8	2	0	16	5	12	3	16	4	19	0	1	29	0	0
Old-Time / Historic	0	2	0	0	0	1	0	0	0	0	0	0	0	149	0	1
Pop	0	3	1	0	81	5	33	28	9	7	2	0	28	157	0	2
Rock	1	6	4	0	154	38	63	54	27	25	4	3	26	1725	0	1
Soul-RnB	0	0	1	0	25	0	4	5	0	3	1	0	0	6	1	0
Spoken	0	0	0	0	7	4	2	3	0	1	0	1	1	5	0	12

Table 1.12: Label performance KNN

Figure 1.4: Confusion Matrix KNN

1.3 Anomaly Detection

Il processo di Anomaly Detection è uno step di primaria importanza per migliorare la qualità dell’analisi. Le tecniche per individuare questi outliers sono molteplici e possono condurre a risultati diversi; per questi motivi, nel presente studio, sono state applicate al dataset 4 differenti tecniche di Anomaly Detection. Prima di queste analisi i dati sono stati normalizzati attraverso il metodo “StandardScaler”, al fine di dare pari importanza ad ogni attributo. I risultati sono stati poi confrontati tra di loro al fine di individuare la miglior tecnica da applicare a questo specifico dataset, così da eliminare l’1% dei valori con outlier score più alto.

1.3.1 Box Plot

La prima tecnica utilizzata è quella del Box Plot; si tratta di una strategia di Anomaly Detection che va ad analizzare ogni attributo singolarmente e non restituisce un outlier score, bensì una label del record: ”outlier” o ”inlier”. Sottponendo i 20 attributi all’algoritmo, ben 9072 records vengono restituiti come outliers, cioè il 36%. Questo valore così alto è spiegabile dall’elevato numero di features ed attraverso lo studio delle curve di distribuzione dei singoli attributi: esse presentano tutte una distribuzione simile alla normale con picchi molto alti in corrispondenza della media. Alti valori nel numero di outliers sono stati rilevati negli attributi che presentavano un picco centrale più elevato e snello, e code allungate. Un esempio è riportato in figura 1.5, dove si può notare come l’attributo (‘chroma_stft’, ‘mean’, ‘01’) presenti una distribuzione piuttosto simmetrica con un picco centrale relativamente basso e code poco allungate. I suoi outliers sono equamente distribuiti tra i due lati e sono in numero relativamente ridotto (436 outliers). L’attributo (‘tonnetz’, ‘std’, ‘01’) al contrario presenta una distribuzione più asimmetrica, con picco centrale particolarmente elevato e coda destra allungata; questo causa un grande numero di outliers tra i valori più alti (710 outliers).

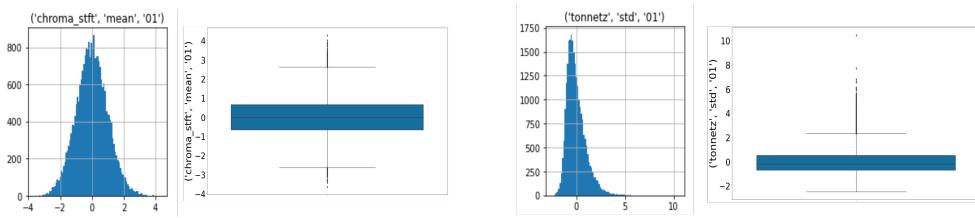


Figure 1.5: Grafici di distribuzione e box plot degli attributi (‘chroma_stft’, ‘mean’, ‘01’) e (‘tonnetz’, ‘std’, ‘01’)

Per rendere l'analisi più significativa, dato l'elevato numero di records anomali identificati, è stato deciso di assegnarvi un valore di outlierness calcolando l'outlier_score per ogni entry del record e prendendo il massimo.

$$\forall i \in records. max : (\forall j \in attributes. outlier_score(df[i, j]))$$

$$outlier_score(value) = \begin{cases} value - (Q3 + IQR \times 1.5) & \text{if } value > Q3 + IQR \times 1.5 \\ |value - (Q1 - IQR \times 1.5)| & \text{if } value < Q1 - IQR \times 1.5 \\ 0 & \text{else} \end{cases} \quad (1.1)$$

Dopodichè sono stati selezionati i 250 record con outlier_score più alto (figura 1.6), la cui distribuzione tra le varie classi è riportata in tabella 1.13.

1.3.2 DbScan

Il DbScan è una tecnica di clustering locale basata sulla densità, che permette di individuare gli outliers senza restituire un score per la classificazione. Utilizzando questa tecnica è necessario scegliere attentamente due parametri: MinPoints ed Epsilon. La ricerca di questi 2 parametri è stata inizialmente condotta attraverso l'individuazione di un range in cui selezionare l'Epsilon migliore. A tale scopo è stato utilizzato il metodo del K-Nearest neighbor, sia con la distanza Manhattan, che con la distanza Euclidea. Il metodo è stato testato con diversi valori di K, ossia calcolando la distanza del k-esimo "vicino" di ogni punto del dataset. Successivamente, si è provato a selezionare il miglior compromesso tra MinPoints e Epsilon, in base al valore della Silhouette. I valori di Epsilon testati vanno da 5 a 10, in base ai risultati precedentemente ottenuti. Modificando il valore di K, si è osservato che il valore della Silhouette più alto (0.87) si ottiene con Epsilon=8 e MinPoints uguale a 10. Applicando l'algoritmo DbScan con questi parametri, si ottiene un unico grande cluster formato da 24997 record e soli 3 record classificati come outliers (figura 1.7). Due di questi outliers hanno come genere top 'Experimental', mentre uno ha 'Instrumental'.

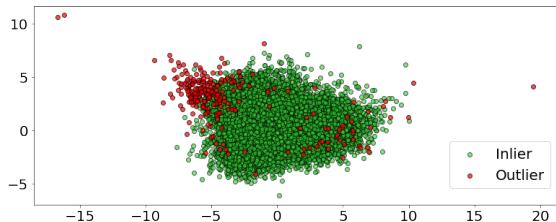


Figure 1.6: Rappresentazione PCA Outlier con BoxPlot

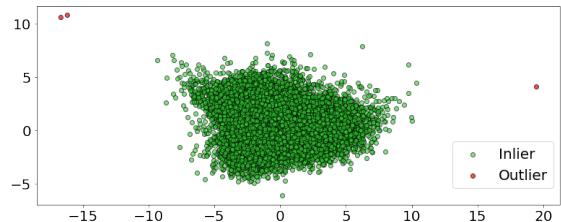


Figure 1.7: Rappresentazione PCA Outlier con DbScan

1.3.3 Isolation Forest

Isolation Forest crea un albero selezionando ogni volta una dimensione in modo random e successivamente selezionando, sempre in modo casuale, uno split value per dividere i dati. Questo procedimento continua sino a realizzare un albero con un solo record in ogni foglia. I records che si trovano nelle foglie più vicine alla radice saranno classificati come anomalie. Il metodo permette di realizzare più alberi, cioè una foresta, e poi prendere per ogni record il valore medio. In questo studio, per avere risultati più precisi, sono stati realizzati 100 alberi, che hanno quindi portato ad una classificazione dell'anomalia caratterizzante ogni record. Inserendo il parametro 'contamination' pari a 0,01 è stato possibile individuare i 250 records in testa a questa classifica (figura 1.8), la cui classe di appartenenza è riportata in tabella 1.14.

1.3.4 ABOD

ABOD è una tecnica che classifica gli outliers in base alla varianza dello spettro degli angoli che ogni record forma con ogni altra coppia di records del dataset. Qui l'assunzione è che un record

è un outlier se si trova al bordo della distribuzione. Una varianza più bassa identifica un valore maggiormente anomalo. Il metodo originale ha un costo computazionale molto elevato, quindi in questa analisi è stato scelto di considerare per ogni punto solo gli angoli con i 30 punti più vicini. Anche qui è stato inserito il parametro ‘contamination’ pari a 0,01 in modo da individuare i 250 record che presentavano una varianza dello spettro più bassa. Il metodo ha però restituito solo 248 record (figura 1.9); un risultato simile può essere spiegato dal fatto che il parametro ‘contamination’ indica qual è la porzione attesa di outliers nel dataset e quindi non deve necessariamente restituire la percentuale precisa. La classe di appartenenza di questi records è riportata in tabella 1.15.

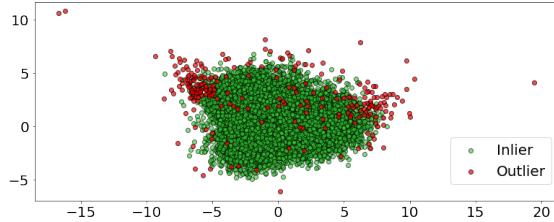


Figure 1.8: Rappresentazione PCA Outlier con Isolation Forest

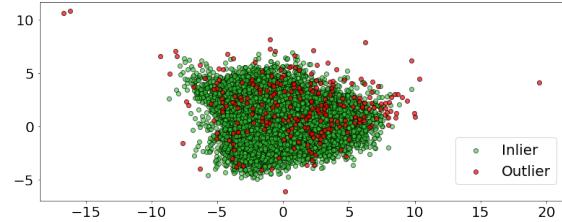


Figure 1.9: Rappresentazione PCA Outlier con ABOD

CLASS LABEL	#REC
Old-Time / Historic	119
Experimental	38
Electronic	37
Instrumental	20
Rock	14
Classical	10
International	6
Hip-Hop	3
Pop	2
Jazz	1

Table 1.13: Class label dei 250 outliers ottenuti con la tecnica del Box Plot

CLASS LABEL	#REC
Old-Time / Historic	87
Electronic	62
Experimental	59
International	17
Instrumental	11
Rock	6
Hip-Hop	3
Classical	2
Pop	1
Folk	1
Spoken	1

Table 1.14: Class label dei 250 outliers ottenuti con la tecnica dell’IsolationForest

CLASS LABEL	#REC
Experimental	98
Electronic	76
Instrumental	27
Rock	18
Old-Time / Historic	8
International	5
Hip-Hop	4
Pop	4
Folk	3
Jazz	2
Spoken	2
Classical	1

Table 1.15: Class label dei 248 outliers ottenuti con la tecnica dell’ABOD

1.3.5 Confronto tra i diversi metodi

Il confronto dei risultati di queste 4 tecniche, come si può notare dalle tabelle riportanti i class label di appartenenza, ha portato a risultati abbastanza discordanti. I 3 record individuati dal DbScan sono presenti anche nelle altre 3 tecniche e permettono di giungere alla conclusione che questi 3 punti sono estremamente distanti dagli altri ed isolati. Dalle tabelle con i class label degli outliers si può notare che i 2 metodi Box Plot ed Isolation Forest classificano come outliers un gran numero di record della classe “Old-Time / Historic”: guardando il totale dei record di questa classe su tutto il dataset (502), è possibile calcolare che dall’analisi con i BoxPlot si classificano come outliers il 24% dei record di questa classe, mentre il 17% con la tecnica dell’Isolation Forest. Questi risultati sono particolarmente alti se comparati con le percentuali degli outliers delle altre classi che non superano mai il 2% con entrambe le tecniche. Inoltre si osserva un valore molto più basso nel numero di outliers per la classe “Old-Time / Historic” utilizzando il metodo ABOD: esso restituisce solo 8 outliers di tale classe, pari all’ 1% dei record. Con quest’ultimo metodo il numero degli outliers su tutte le classi è ben bilanciato. Da tali riflessioni si può ipotizzare che le tecniche Box Plot ed Isolation Forest vadano a identificare come outliers dei record che appartengono invece ad una classe (“Old-Time / Historic”) con una struttura o densità diversa delle altre.

Per confermare questa tesi, i dati in 20 dimensioni sono stati trasformati in dati in 3 dimensioni con il metodo PCA al fine di poterli visualizzare con il grafico 3D riportato in figura 1.10. Essa conferma tutte le ipotesi precedentemente descritte: in primo luogo si notano in modo chiaro i 3 record piuttosto distanti dagli altri, i quali sono i record identificati attraverso il DbScan; in

secondo luogo è possibile vedere come la classe “Old-Time / Historic” (i punti in rosso in figura) abbia una forma allungata che tende ad allontanarsi dal centro dell’ ammasso globulare dei record.

La forma di questa classe, causata da valori su alcuni attributi distanti dalla media generale, rende i 2 metodi Box Plot ed Isolation Forest non adatti all’ analisi questo dataset. Allo stesso modo è stato ritenuto inadeguato il metodo del DbScan: la motivazione ricade sul significato di un’analisi di densità in 20 dimensioni. Il risultato con un solo grande cluster e l’individuazione di 3 outliers (molto distanti dagli altri punti) conferma la difficoltà di questo metodo con l’aumentare della dimensionalità. Per tutte le ragioni sopraelencate è stato deciso di utilizzare il metodo ABOD, il quale lavora bene anche con elevate dimensioni e va ad individuare 248 record bilanciati tra le varie classi. Questi 248 record sono stati quindi eliminati dal dataset e non saranno utilizzati nelle prossime analisi.

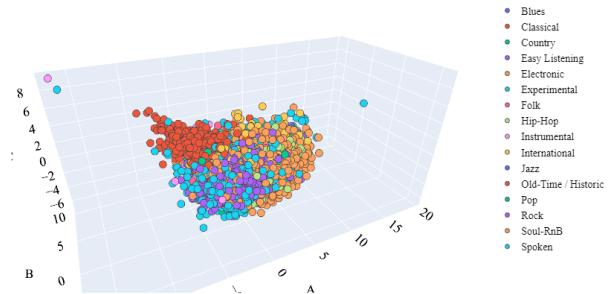


Figure 1.10: visuale 3D dei dati trasformati con la tecnica PCA
Per tutte le ragioni sopraelencate è stato deciso di utilizzare il metodo ABOD, il quale lavora bene anche con elevate dimensioni e va ad individuare 248 record bilanciati tra le varie classi. Questi 248 record sono stati quindi eliminati dal dataset e non saranno utilizzati nelle prossime analisi.

1.4 Imbalanced Learning

In seguito alla fase di Anomaly Detection che ha portato all’eliminazione degli outlier che avrebbero potuto influenzare in maniera negativa il risultato della classificazione, sono state adottate delle tecniche di Imbalanced Learning al fine di ridurre l’impatto dello sbilanciamento presente sui dati. Nel grafico è estremamente evidente la predominanza di record delle classi Rock ed Electronic, oltre 6000 ciascuna, al contrario delle classi il cui numero di sample è minore di 200, che non risultano sufficienti per addestrare in maniera affidabile il modello.

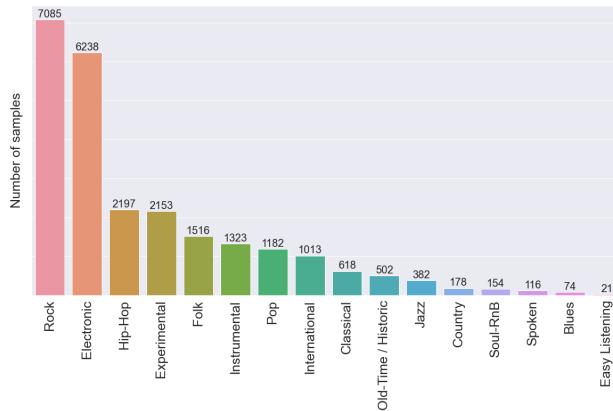


Figure 1.11: Distribuzione record per genere

E’ stato scelto come oggetto di analisi il Decision Tree, in quanto tramite il KNN non si osservano miglioramenti significativi con l’utilizzo delle tecniche di Imbalanced Learning.

Le strategie utilizzate sono le seguenti:

- Bilanciamento delle classi tramite Over/Under-sampling
- Utilizzo del parametro class_weight del DecisionTreeClassifier

Nel primo caso il dataset è stato spartito in Training, Validation e Test set. Il bilanciamento è stato effettuato solo sul training set, in modo da valutare poi le performance solo su un subset di dati originali.

Per bilanciare il training set è stato utilizzato l’algoritmo SMOTE per l’Oversampling, in combinazione con il NearMiss1 per l’Undersampling, in modo da avere lo stesso numero di sample per ogni label (1000). Il NearMiss (versione 1) è una tecnica che mantiene i punti della majority

class la cui distanza media dai punti della minority class è minima. La scelta del NearMiss1 deriva dal fatto che si tratta di una ‘Controlled under-sampling technique’: permette cioè di indicare, attraverso il parametro sampling_strategy, il numero di record desiderati per ogni classe. Mentre con l’utilizzo di altre tecniche come il CNN e il Tomek’s link ciò non è permesso e l’Undersampling porta ad avere un numero troppo ridotto di record sul dataset utilizzato.

La Grid Search, in questo caso utilizzata con validation set fisso, ha prodotto gli iperparametri mostrati in tabella 1.16.

max_depth	12
min_sample_leaf	30
min_sample_split	35
criterion	entropy

Table 1.16: Iperparametri

	Precision	Recall	F1	Accuracy
Train	0.26	0.51	0.27	0.28
Val	0.21	0.31	0.20	0.23
Test	0.21	0.29	0.20	0.23

Table 1.18: Performance Decision Tree Balanced

	precision	recall	f1-score	auc	support
Blues	0.01	0.13	0.02	0.55	15
Classical	0.35	0.60	0.44	0.93	124
Country	0.05	0.39	0.08	0.74	36
Easy Listening	0.00	0.00	0.00	0.59	4
Electronic	0.46	0.18	0.26	0.61	1248
Experimental	0.17	0.11	0.13	0.61	431
Folk	0.35	0.34	0.34	0.77	303
Hip-Hop	0.34	0.31	0.32	0.74	439
Instrumental	0.22	0.26	0.23	0.77	265
International	0.12	0.20	0.15	0.66	203
Jazz	0.05	0.24	0.08	0.69	76
Old-Time / Historic	0.61	0.90	0.73	0.98	100
Pop	0.07	0.07	0.07	0.58	236
Rock	0.52	0.19	0.28	0.69	1417
Soul-RnB	0.01	0.19	0.03	0.65	31
Spoken	0.04	0.48	0.07	0.71	23
accuracy				0.23	4951
macro avg	0.21	0.29	0.20	0.70	4951

Table 1.17: Label Perfomance DT Balanced

Lo score ottenuto dal modello sul test set presenta una migliore recall in confronto al Decision Tree allenato sul training set sbilanciato. Rimane solamente un genere completamente misclassificato (Easy Listening). Calano invece la precision e la f1-macro a causa dell’introduzione di numerosi record fintizi in 8 delle 16 label totali.

Migliori sono invece i risultati del Decision Tree con class_weight settato a ‘balanced’ come mostrato in tabella 1.21. In questo caso si ottiene un valore pari a 0.31 per average recall con performance migliori anche in termini di precision e f1.

max_depth	14
min_sample_leaf	20
min_sample_split	30
criterion	gini

Table 1.19: Iperparametri

	Precision	Recall	F1	Accuracy
Train	0.31	0.62	0.33	0.36
Val	0.23	0.29	0.21	0.26
Test	0.23	0.31	0.23	0.28

Table 1.21: Performance Decision Tree Class Weight

	precision	recall	f1-score	auc	support
Blues	0.01	0.14	0.02	0.55	22
Classical	0.33	0.64	0.44	0.86	185
Country	0.03	0.25	0.06	0.66	53
Easy Listening	0.01	0.17	0.03	0.58	6
Electronic	0.55	0.21	0.31	0.72	1871
Experimental	0.21	0.11	0.14	0.64	646
Folk	0.28	0.33	0.31	0.77	455
Hip-Hop	0.30	0.39	0.34	0.76	659
Instrumental	0.28	0.29	0.29	0.74	397
International	0.15	0.21	0.18	0.67	304
Jazz	0.04	0.20	0.07	0.65	115
Old-Time / Historic	0.71	0.86	0.78	0.96	151
Pop	0.08	0.11	0.10	0.60	355
Rock	0.64	0.32	0.43	0.77	2126
Soul-RnB	0.02	0.22	0.03	0.59	46
Spoken	0.10	0.54	0.17	0.76	35
accuracy				0.28	7426
macro avg	0.23	0.31	0.23	0.71	7426

Table 1.20: Label Perfomance DT Class Weight

Inoltre, osservando singolarmente le label si può notare che nessuna di essa viene completamente misclassificata sul test set. In contropartita si ottiene una recall più bassa per i generi più popolari

quali Electronic e Rock.

In tabella vengono riportate le recall del modello allenato sul training set sbilanciato e quelli con le tecniche di imbalanced learning.

	Blues	Classical	Country	Easy List	Electr	Experim	Folk	Hip-Hop	Instrum	Internat	Jazz	Old/Hist	Pop	Rock	Soul-RnB	Spoken
Unbalanced	0.00	0.47	0.00	0.00	0.65	0.10	0.36	0.35	0.22	0.15	0.01	0.88	0.02	0.72	0.00	0.19
Balanced	0.13	0.60	0.39	0.00	0.18	0.11	0.34	0.31	0.26	0.20	0.24	0.90	0.07	0.19	0.19	0.48
Class_weight	0.14	0.64	0.25	0.17	0.21	0.11	0.33	0.39	0.29	0.21	0.20	0.86	0.11	0.32	0.22	0.54

1.5 Analisi dei risultati

L'analisi scelta nel progetto è risultata, per certi aspetti, difficoltosa ed i risultati lo confermano: sia la classificazione con il Decision Tree (con o senza tecniche di imbalanced learning), sia quella con il KNN, non portano a risultati brillanti come si può vedere dalle tabelle nei paragrafi precedenti. I motivi di tali esiti sono senz'altro da ricondursi al grande sbilanciamento dei dati presenti nel dataset originale. A questo vanno uniti l'obiettivo ed il significato dell'analisi: in questo lavoro è stato assunto che nella realtà non ci sia uno sbilanciamento delle tracce musicali su alcuni generi e si è cercato quindi di classificare i record in modo che anche le classi minori venissero identificate. Per questo è stata utilizzata come misura di riferimento la media su F1 tra le varie classi. I risultati migliori sono stati individuati con il metodo del KNN, che ha mostrato un risultato medio di F1 pari a 0,36. Quest'ultimo dato non ha un valore particolarmente alto, ma occorre tener presente che le classi sono ben 16 e questo aumenta la difficoltà della classificazione.

In figura 1.12 è riportato il grafico della Precision-Recall Curve ottenuto con il metodo del KNN. Questo grafico conferma ancora la difficoltà del metodo nel classificare le classi con meno record; le aree minori sono rappresentate infatti da 'Blues', 'Country', 'Easy Listening', 'Soul-RnB'. Ad esse va aggiunta 'Pop' che, nonostante presenti un numero molto maggiore di record, non viene classificata correttamente. L'area più grande si ha invece con la classe 'Old-Time / Historic', che come si era visto dagli outliers presenta record un po' distaccati nello spazio dagli altri.

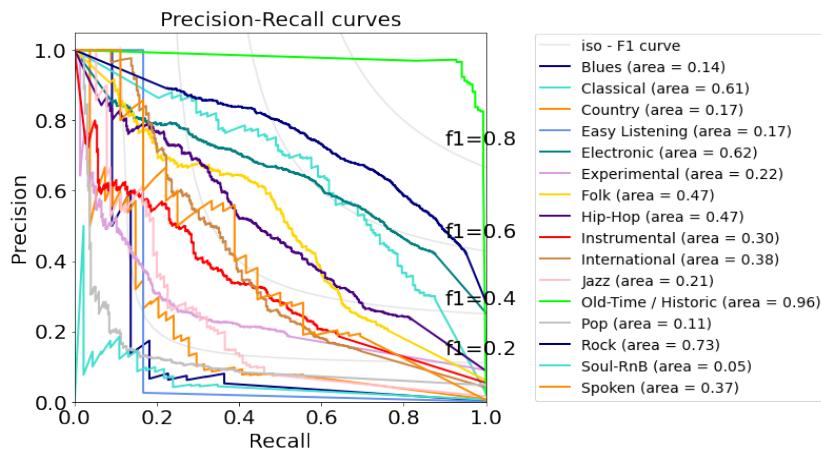


Figure 1.12: Precision-Recall Curve KNN

Oltre a quanto detto, è stato anche osservato che, nonostante il metodo del KNN abbia problemi con l'aumentare della dimensionalità, i risultati subivano miglioramenti continui aumentando il numero di attributi. Utilizzare quindi un numero maggiore, dei 20 attributi utilizzati per questa analisi, sebbene computazionalmente più complesso potrebbe migliorare i risultati. Questo unito ad un maggior numero di record, in particolare delle classi meno rappresentate, potrebbe portare a buoni risultati per questo studio.

Modulo 2

2.1 Naive Bayes

L'assunzione principale del classificatore Naive Bayes, è che gli attributi siano indipendenti l'uno dall'altro. Tale supposizione è solo in parte rispettata dai dati di questo studio: la feature selection iniziale ha portato a scegliere solo gli attributi con una correlazione minore di 0.60 (figura 2.1). Questo non esclude totalmente una dipendenza, ma ne limita almeno l'importanza. Di conseguenza è stata valutata come ragionevole (con i limiti del caso) l'approssimazione per cui questi attributi vengono considerati indipendenti tra di loro in modo da poter utilizzare questo metodo.

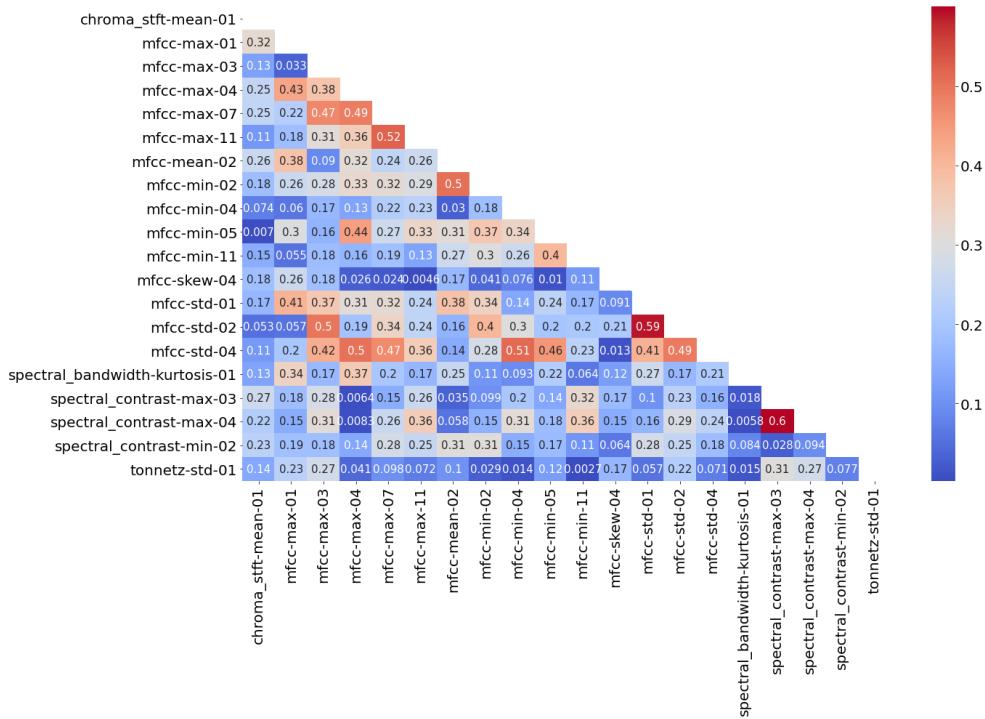


Figure 2.1: Correlazione di Pearson

Per il classificatore Naive Bayes, utilizzato solo su dati numerici continui, non è stato necessario settare parametri. I risultati sul training e sul test set sono riportati in tabella 2.1

I risultati sono abbastanza in linea con quelli degli altri classificatori; questo conferma che l'assunzione che i dati utilizzati siano indipendenti è ragionevole e sicuramente non porta ad un importante decadimento delle prestazioni.

	Precision	Recall	F1	Accuracy
Train	0.30	0.33	0.29	0.47
Test	0.30	0.31	0.29	0.47

Table 2.1: Performance Naive Bayes

2.2 Logistic Regression

Per la Logistic Regression sono stati testati diversi algoritmi per l'ottimizzazione, presenti nella libreria scikit learn; in tabella 2.2 sono mostrate le combinazioni di penalty e schema di multi_class classification, dato che non tutte sono supportate. Oltre allo standard One-vs-rest, che risolve il problema attraverso più classificatori binari, è possibile utilizzare il metodo multinomial che, tramite la funzione 'softmax', restituisce la probabilità per ogni label. Per le penalty si segnala la presenza di elasticnet che combina l1 ed l2 e può essere utilizzato solo con il solver 'saga'.

		solver				
		liblinear	lbfgs	newton-cg	sag	saga
penalty	l1	✓				✓
	l2	✓	✓	✓	✓	✓
	elasticnet					✓
	ovr	✓	✓	✓	✓	✓
	multinomial		✓	✓	✓	✓

Table 2.2: Combinazioni supportate

Dalla GridSearch emerge che i migliori iperparametri in termini di F1 score sul validation set si ottengono con l'utilizzo del solver 'liblinear', con il parametro C settato a 100; quest'ultimo indica l'intensità della regolarizzazione e valori bassi implicano una regolarizzazione più forte.

Param	Range	Best
solver	[liblinear, lbfgs, newton-cg, sag, saga]	liblinear
penalty	[l1, l2, elasticnet]	l1
multi_class	[ovr, multinomial]	ovr
C	(10^{-3} , 10^4)	10^2
class_weight	[balanced]	balanced

	Precision	Recall	F1	Accuracy
Train	0.31	0.40	0.31	0.48
Test	0.31	0.37	0.31	0.48

Table 2.4: Performance Logistic Regression

Table 2.3: Griglia parametri

2.3 Rule Based

Per l'analisi con il classificatore Rule based è stata utilizzata la libreria Python 'wittgenstein' che offre la possibilità di utilizzare l'algoritmo Ripper, ma solo con un target label binario. Perciò per utilizzarlo nel task multiclass di genre recognition, è stato realizzato un ciclo (for loop) in cui ad ogni iterazione vengono estratte le regole per uno specifico class label (attraverso il parametro pos_class), assicurandosi successivamente di togliere i record 'coperti' da tali regole in modo da non essere presi in considerazione nelle successive iterazioni. Lo stesso principio è stato utilizzato in fase di predizione, identificando ad ogni iterazione i record coperti dalle regole ed assegnando infine i record non coperti da nessuna regola al genere predominante. Siccome le regole potrebbero non essere mutualmente esclusive, una volta assegnata una label al record da predire, quest'ultima non può essere più modificata. Il ciclo è stato realizzato andando dalla classe con meno record sino a quella predominante. Per migliorare i risultati del classificatore, è stato anche necessario agire sui parametri dell'algoritmo Ripper, riportati in tabella 2.5, ed ottimizzati con l'uso del validation set.

Param	Range	Best
n_discretize_bins	[3,5,7,10]	3
k	[0,1,2]	0
prune_size	[0.1, 0.15, 0.2, 0.25, 0.3]	0.2

Table 2.5: Parametri Rule Based

	Precision	Recall	F1	Accuracy
Train	0.70	0.34	0.39	0.51
Test	0.31	0.22	0.22	0.44

Table 2.6: Performance Rule Based

Dai risultati sul validation set emerge che aumentando il parametro 'prune_size', i risultati su F1 migliorano sino al valore 0.2, per poi calare nuovamente. Questo parametro permette quindi di evitare l'overfitting, andando a rivedere ogni regola creata. Valori maggiori di 0.2 portano ad un risultato di underfitting. Guardando ai risultati non sembra invece che il parametro k (il numero di run di ottimizzazione) abbia effetto. Applicando i migliori parametri ottenuti, si giunge ai risultati riportati in tabella 2.6.

2.4 SVM

Per il Support Vector Machine è stato necessario eseguire la normalizzazione dei dati attraverso l'uso di StandardScaler. Dopodiché si è proceduto con la fase di hyperparameter tuning per

selezionare il kernel da utilizzare con i relativi parametri (tabella 2.7). Anche in questo caso, come per la logistic regression, vi è la presenza del parametro C per la regolarizzazione.

Param	Range	Best
kernel	[linear, sigmoid, poly, rbf]	rbf
C	(10^{-3} , 10^3)	10
gamma	[auto, scale]	auto
class_weight	[balanced]	balanced

	Precision	Recall	F1	Accuracy
Train	0.71	0.88	0.77	0.75
Test	0.38	0.43	0.40	0.54

Table 2.8: Performance SVM

Table 2.7: Griglia parametri

Il kernel Radial basis function (o 'rbf') ottiene la miglior performance sul validation set con uno score F1 di 0.39. Gli score sul training set in tabella 2.8 potrebbero suggerire overfitting sul training set con valori intorno a 0.8. Tuttavia, provando a diminuire il fattore C, che determina una regolarizzazione più aggressiva, si nota un peggioramento sul validation set oltre che sul training set. Ad esempio utilizzando C=1 si osserva un calo di F1 a 0.52 sul training set, valore meno anomalo rispetto al 0.77 del modello scelto, ma le performance calano anche sul validation set con uno score di 0.36 (rispetto a 0.39 con C=10). Inoltre il modello scelto presenta delle performance in linea anche sul test set. Ciò porta a concludere che la differenza tra lo score su training e test set non sia dovuta all'overfitting.

2.5 Neural Network

Per la classificazione con Neural Network è stata utilizzata la libreria Scikit-learn. In questo caso viene utilizzato l'intero dataset, comprensivo dei 518 attributi, essendo la metodologia adatta in caso di alta dimensionalità. Dopo un'apposita normalizzazione dei dati si è passati alla scelta dell'architettura da utilizzare. Sono state perciò selezionate 4 combinazioni con differenti hidden layer e hidden node per layer:

1. (512, 256, 128,)
2. (512, 256, 128, 64,)
3. (1024, 512, 256, 128,)
4. (128, 256, 512, 256,)

Le funzioni di attivazione utilizzate negli hidden layer sono 'ReLU' e 'tanh'. Non è invece possibile impostare la funzione di attivazione dell'output layer sebbene sia possibile estrarla dopo il training del modello: in tal caso viene utilizzata la softmax, ideale per i task multiclass. Su queste architetture sono state provate diverse combinazioni di algoritmi e parametri, riportati in tabella 2.9. Inoltre per limitare l'overfitting, oltre all'uso del parametro alpha, che permette di gestire la regolarizzazione, viene utilizzata la strategia di early stopping fermando il training della rete neurale quando per 10 successive epoche non si verifica un miglioramento dello score sul validation set.

Param	Range	Best
solver	[adam, sgd]	adam
activation	[relu, tanh]	tanh
learning_rate	[adaptive]	adaptive
alpha	(0.0001, 10)	0.01
momentum	(0.3, 0.9)	/

	Precision	Recall	F1	Accuracy
Train	0.97	0.95	0.96	0.97
Test	0.58	0.49	0.52	0.68

Table 2.10: Performance NN

Table 2.9: Griglia parametri

L'architettura che ha ottenuto migliori risultati sul validation set è stata la numero 3, con uno F1 score di 0.52 sul test set. Da notare l'alta Precision che si ottiene, pari a 0.58: solo Easy Listening continua ad essere completamente misclassificato.

In figura è mostrato l'andamento della loss durante la fase di training della rete neurale: il valore ottimale di 0.083 viene raggiunto prima del picco, dopodichè si verificano 10 iterazioni con uno score peggiore sul validation set che causano l'arresto della discesa di gradiente. Picco che persiste anche modificando il parametro `n_iter_no_change` e la dimensione dei mini-batch. L'alto punteggio sul training set fa pensare che sia possibile ottimizzare ancora il modello, abbassando lo score sul training set al fine di permettere una generalizzazione migliore. Tuttavia, aumentando il valore del parametro `alpha` per la regolarizzazione si ottiene una riduzione delle performance sul training set ma ciò non si traduce in un miglior score sul validation set. A tal proposito sarebbe risultato utile l'utilizzo di dropout, i quali non sono però supportati dalla libreria di Scikit-learn.

2.6 Random Forest

Per risolvere il task sono state testate 2 tipologie di Ensamble method: RandomForest e AdaBoost.

Il RandomForest è una tipologia particolare di Bagging: fa quindi uso della tecnica di bootstrap sui record per allenare i base estimator che, in questo caso, sono Decision Tree. Diversamente del bagging classico, però, il Random Forest utilizza ad ogni iterazione un sottinsieme degli attributi del dataset originale. Per quanto riguarda il numero di feature prese in considerazione, sono state testate attraverso la GridSearch la radice e il logaritmo in base 2 delle 20 feature precedentemente selezionate. Inoltre, trattandosi di un dataset sbilanciato, è stata adottata la strategia di imbalanced learning con `class_weight` settato a 'balanced' e a 'balanced_subsample'. Quest'ultimo agisce come 'balanced' calcolando però i pesi in base al bootstrap sample, anziché all'intero dataset. I risultati sul validation set della fase di hyperparameter tuning hanno portato alla scelta dei parametri evidenziati in tabella 2.11.

Param	Range	Best
<code>max_depth</code>	(8,20)	18
<code>min_samples_split</code>	(30,80)	40
<code>min_samples_leaf</code>	(30,80)	30
<code>class_weight</code>	[balanced, balanced_subsample]	balanced_subsample
<code>max_features</code>	[sqrt, log2]	log2
<code>criterion</code>	[gini, entropy]	gini

Table 2.11: Griglia parametri

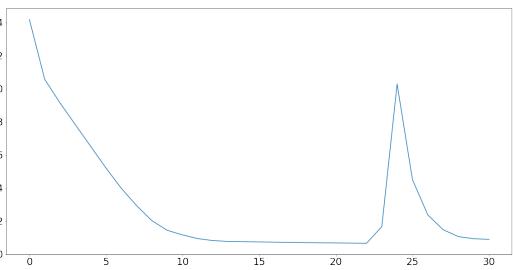


Figure 2.2: Loss curve

	Precision	Recall	F1	Accuracy
Train	0.49	0.74	0.55	0.58
Test	0.32	0.40	0.33	0.48

Table 2.12: Performance Random Forest

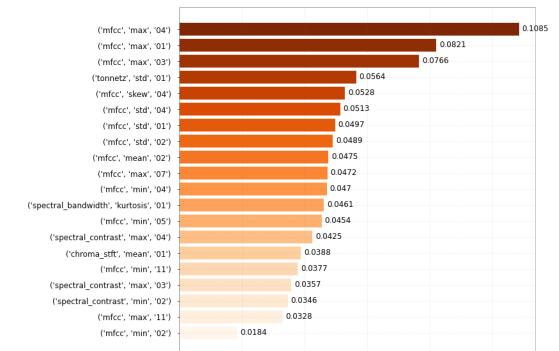


Figure 2.3: Feature Importance

molteplici feature (dato che ne seleziona solo un sottinsieme), è stato utilizzato anche con l'intero dataset (con tutte le 518 features) ed con il dataset con 191 features, escludendo cioè quelle con correlazione sopra lo 0.60. Le performance più alte vengono ottenute con la RandomForest su 518 attributi, con una F1 sul test che sale a 0.40, rispetto allo 0.38 ottenuto su 191 e 0.33 su 20 attributi.

Param	Range	Best
max_depth	(8,20)	14
min_samples_split	(30,80)	45
min_samples_leaf	(30,80)	30
class_weight	[balanced, balanced_subsample]	balanced
max_features	[sqrt, log2]	sqrt
criterion	[gini, entropy]	entropy

	Precision	Recall	F1	Accuracy
Train	0.66	0.81	0.70	0.66
Test	0.40	0.46	0.40	0.54

Table 2.14: Performance Random Forest con 518 attributi

Table 2.13: Griglia parametri

2.7 AdaBoost

L'altro metodo di Ensamble è AdaBoost. Si tratta di un algoritmo di Boosting che si distingue dal Bagging per l'utilizzo della tecnica di bootstrap su record a cui viene assegnato un peso: ai record misclassificati nel bootstrap precedente verranno infatti assegnati dei pesi maggiori in modo che essi compaiano con più frequenza nel bootstrap successivo. In aggiunta a ciò, anche ad ogni modello sarà assegnato un peso specifico; in questo modo, modelli con un errore più basso avranno maggior influenza sulla classificazione finale. Come classificatore per l'algoritmo AdaBoost è stato scelto il Decision Tree. Per cercare di ottenere risultati migliori (rispetto all'utilizzo di stump come weak classifier), sono stati testati alcuni valori di 2 parametri in particolare: la profondità dell'albero e il numero di modelli diversi realizzati da AdaBoost. Per fare ciò, è stato quindi splitato il dataset tra training, validation e test set in modo da non influenzare i valori finali dell'analisi. Nella tabelle 2.15 e 2.16 sono riportati i migliori parametri scelti ed i risultati finali su training e test set con questi parametri.

Param	Range	Best
criterion	[gini]	gini
max_depth	[5,7,10,14]	10
min_samples_split	[50]	50
min_samples_leaf	[50]	50
class_weight	[None,balanced]	balanced
n_estimators	[50,100,500,1000]	500

Table 2.15: Parametri Boosting

	Precision	Recall	F1	Accuracy
Train	0.90	0.82	0.85	0.80
Test	0.38	0.27	0.28	0.53

Table 2.16: Performance Boosting

Dai vari risultati ottenuti sul validation set, si è potuto osservare come, aumentando il numero di modelli (estimators), si ottengano risultati sempre più precisi. Dopo il valore di 500, però, i miglioramenti sono talmente bassi che il costo computazionale suggerisce di fermarsi. Per quanto riguarda la profondità dell'albero, si riscontrano miglioramenti fino alla profondità di 10, dopodichè si ha un incremento della performance solo sul training set, mentre si ha peggioramento sul validation set, chiaro indice di overfitting. Infine, si nota come l'utilizzare il parametro class_weight per tentare di bilanciare le varie classi conduca ad un miglioramento dei risultati veramente contenuto; questo è in accordo con la struttura dell'algoritmo AdaBoost che cerca già di bilanciare le classi meno presenti e quindi più difficili da classificare, aumentandone i pesi ad ogni iterazione.

2.8 Confronto Classificatori

Arrivati a questo punto dello studio, è possibile fare delle valutazioni su quali classificatori abbiano risposto in maniera più efficace all'analisi di riconoscimento di genere. In tabella 2.17 vengo riportati i diversi classificatori utilizzati, con i valori ottenuti sul test set. Come si può vedere, al

fine di avere migliori performance, dove è stato possibile sono stati aumentati gli attributi oggetto di analisi ed è stato anche utilizzato il parametro 'class weight' per contrastare lo sbilanciamento del dataset. Il modello che performa le migliori prestazioni è il Neural Network, con un valore di F1 medio pari a 0.52, seguito dal Random Forest; entrambi questi modelli sono stati costruiti con 518 attributi, testimonianza di un'analisi più precisa aumentando i dati a disposizione. Tra i modelli che, invece, utilizzano soltanto 20 attributi, le migliori prestazioni sono state ottenute con SVM e con KNN.

Classificatore	N* Attribute	Class Weight	Precision	Recall	F1	Accuracy
Decision Tree	20	SI	0.23	0.31	0.23	0.28
KNN	20	NO	0.41	0.36	0.36	0.55
Naive Bayes	20	NO	0.30	0.31	0.29	0.47
Logistic Regression	20	SI	0.31	0.37	0.31	0.48
Rule Based	20	NO	0.31	0.22	0.22	0.44
SVM	20	SI	0.38	0.43	0.40	0.54
Neural Network	518	NO	0.58	0.49	0.52	0.68
Random Forest	518	SI	0.40	0.46	0.40	0.54
AdaBoost	20	SI	0.38	0.27	0.28	0.53

Table 2.17: Performance dei classificatori utilizzati

2.9 Regression Problem

Ai fini della determinazione di un problema di regressione sono stati scelti i seguenti attributi del dataset Tracks: 'comments', 'favorites', 'interest', 'listens'. Tali attributi sono stati scelti anche sulla base della maggiore correlazione, in modo da poter eseguire il presente task nella maniera più efficace possibile (figura 2.4). L'analisi è quindi stata basata su di un dataframe di 25000 records coerentemente con le precedenti analisi. Non è stato necessario normalizzare i dati prima dell'analisi, grazie alla presenza del parametro 'Normalize' nelle librerie di SkLearn utilizzate per la regressione.

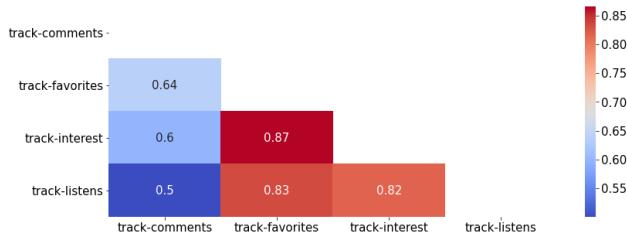


Figure 2.4: Correlazione di Pearson

2.9.1 Simple Linear Regression

Sono stati scelti gli attributi 'favorites' e 'listens', così da poter individuare quanti soggetti hanno espresso preferenza per una canzone (variabile dipendente) dato il numero di ascolti (variabile indipendente).

Le performances su training set e test set del modello generato sono visualizzabili rispettivamente nelle Tabelle 2.18 e 2.19. Attraverso l'intercetta ed i coefficienti individuati è possibile stabilire la seguente funzione-obiettivo (visualizzabile anche in figura 2.5):

$$y = 0.0016 \cdot x - 0.3340$$

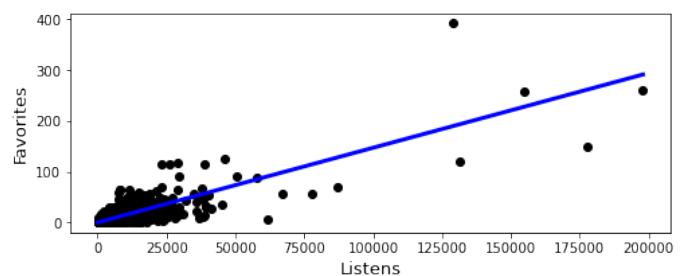


Figure 2.5: Plot Simple Linear Regression

2.9.2 Ridge e Lasso con Simple Linear Regression

Ai fini dell'individuazione dei migliori parametri, è stata eseguita una GridSearch per i seguenti valori di alpha: [0.1, 0.001, 0.01, 1, 10]; questa ha permesso di individuare, attraverso la cross validation, l'alpha migliore pari a 0.1 per Ridge e 0.001 per Lasso. Dall'analisi sono state ottenute le seguenti funzioni-objettivo:

$$\text{Ridge} : y = 0.0014 \cdot x + 0.0427$$

$$\text{Lasso} : y = 0.0016 \cdot x - 0.2948$$

E' possibile inoltre osservare (tabelle 2.18 e 2.19) che, nonostante aumenti leggermente il mean square error (MSE) sul training set con i due metodi, l'errore sul test set diminuisce (sia sul MSE che sul MAE). Questo è coerente con il loro obiettivo di contrastare l'overfitting.

Linear Regression Model			
	Linear	Ridge	Lasso
MSE	81.928	83.548	81.945
MAE	2.572	2.524	2.566

Table 2.18: MSE, MAE sul Training set

Linear Regression Model			
	Linear	Ridge	Lasso
MSE	32.985	30.814	32.694
MAE	2.367	2.318	2.361

Table 2.19: MSE, MAE sul Test set

2.9.3 Multiple Linear Regression

Oltre agli attributi utilizzati nella precedente sezione, sono stati considerati anche 'comments' e 'interest' ed è stato nuovamente utilizzato 'favorites' come variabile dipendente.

Anche in questo contesto, dopo aver creato il modello base, è stata eseguita una GridSearch per trovare il miglior valore di alpha per i metodi di regolarizzazione Ridge e Lasso; in questo caso il miglior valore di alpha è risultato essere 0.001 per entrambi.

Le funzioni ottenute dal modello normale e dai due regolarizzati, sono molto simili con i coefficienti e l'intercetta che variano solo se vengono considerate molte cifre decimali. Per questo motivo è riportata di seguito una sola funzione valida per i 3 modelli:

$$y = (5.80) \cdot x_1 + (2.58 \cdot 10^{-4}) \cdot x_2 + (6.51 \cdot 10^{-4}) \cdot x_3 + 0.86$$

In questa funzione le variabili x sono riferite, in ordine, agli attributi 'comments', 'interest' e 'listens'. Le diverse performance dei 3 modelli su training set e test set sono riportati nelle tabelle 2.20 e 2.21. Alla luce dei risultati ottenuti è possibile affermare che il modello senza regolarizzazione è privo di overfitting: questo perchè i risultati sono i soliti per tutti e tre i modelli, rendendo Ridge e Lasso inutili nel ridurre l'overfitting. Un'ulteriore conferma di ciò risiede anche nell'aver identificato un valore molto basso di alpha sia per Ridge che per Lasso; questo indica che il secondo addendo delle due funzioni di regolarizzazione assume poca importanza.

Linear Regression Model			
	Linear	Ridge	Lasso
MSE	45.882	45.882	45.882
MAE	2.459	2.459	2.459

Table 2.20: MSE, MAE sul Training set

Linear Regression Model			
	Linear	Ridge	Lasso
MSE	26.688	26.688	26.688
MAE	2.314	2.314	2.314

Table 2.21: MSE, MAE sul Test set

Modulo 3

3.1 Time Series Selection

Il Free Music Archive (FMA) da cui sono stati presi i dati per la parte finora realizzata del progetto, mette anche a disposizione le tracce mp3 delle canzoni; precisamente rende disponibili 30 secondi di ogni traccia audio. Attraverso la libreria librosa, è stato possibile trasformare questi mp3 in valori temporali riuscendo così ad avere delle time series. Per non avere problemi computazionali sia in questa fase, sia nelle successive analisi, si è scelto di estrarre le time series soltanto di 4000 tracce audio appartenenti a 4 generi top differenti equamente suddivisi: 'Rock', 'Hip-Hop', 'Electronic' e 'Experimental'. Per ognuna di queste tracce sono stati trasformati in valori numerici i primi 28 secondi di registrazione dei 30 a disposizione: questo è stato deciso al fine di evitare problemi con la terminazione delle tracce che non sempre coincideva. E' stato inoltre scelto come 'sampling rate', cioè il numero di osservazioni al secondo, il valore di 12000, al fine di poter lavorare con una buona qualità di dati.

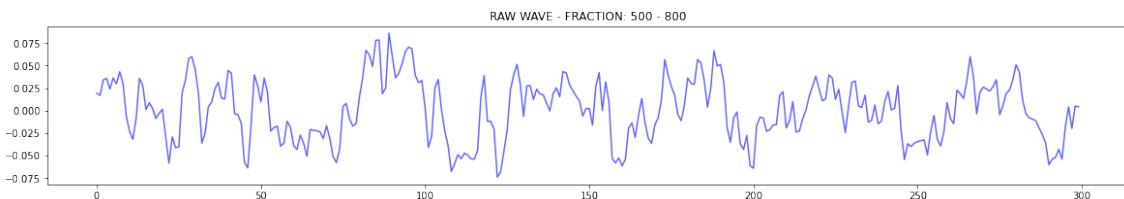


Figure 3.1: Onda ottenuta trasformando il file mp3 in valori numerici. Figura raffigurante la traccia audio con indice 2, tra i time stamp 500 e 800

Durante l'elaborazione 6 tracce sono risultate danneggiate e quindi il numero totale di tracce audio trasformate in time series è stato 3994.

Il numero di colonne ottenute è stato invece di 336000 ($12000 * 28$), decisamente troppe per poter condurre un'analisi. E' stato quindi deciso di calcolare da questi valori gli Spectral Centroid per ogni traccia (sempre attraverso la libreria Librosa). Questo ha permesso di ridurre notevolmente le colonne delle time series, passando da un valore 336000 a 657. Ogni 512 valori, è stato infatti calcolato il valore dello Spectral Centroid all'interno del frame, come la media pesata delle frequenze presenti nel segnale (Librosa utilizza la trasformata di Fourier per ottenere questo valore attraverso lo spettrogramma, utilizzando sia le frequenze sia l'intensità del segnale audio).

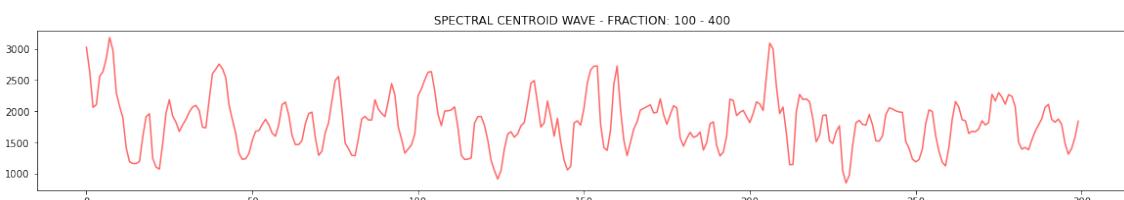


Figure 3.2: Onda ottenuta calcolando gli Spectral Centroid. Figura raffigurante la traccia audio con indice 2, tra i time stamp 100 e 400

Il risultato finale di questa trasformazione è stato quindi quello di ottenere 3994 time series con 657 time stamp ciascuna. Queste time series saranno utilizzate nelle successive analisi.

3.2 Time Series Clustering

Il dataset di time series descritto nella sezione precedente è stato utilizzato per effettuare il clustering al fine di delineare delle strutture sottostanti ai dati.

Su quest'ultimo è stato necessario innanzitutto applicare l'amplitude scaling in quanto le time series hanno un'ampiezza variabile e ciò potrebbe comportare dei problemi con la distanza Euclidea. Successivamente sono state utilizzate le 2 tipologie di clustering, partizionale e gerarchico, nel seguente modo:

- Partizionale con il KMeans sul dataset originale ed approssimato.
- Gerarchico agglomerativo sul dataset approssimato.
- Partizionale con il KMeans sul dataset di features calcolate a partire dalle time series.

3.2.1 Partitional Clustering

Inizialmente è stato eseguito il KMeans sul dataset originale, composto quindi da 3994 time series ciascuna di 657 timestamp, con la distanza Euclidea. In particolare utilizzando l'elbow method, al fine di selezionare il giusto K sul quale eseguire il clustering, si ottiene, con $k=5$, un silhouette score molto basso (0.009).

Con la metrica Dynamic Time Warping è stato invece necessario approssimare le time series con il metodo SAX utilizzando 50 segmenti e un alfabeto di lunghezza 8. Ciò permette di ottenere una complessità computazionale accettabile per le analisi successive.

Inoltre sono stati sfruttati i vincoli sul dtw path che permettono di non calcolare l'intera dtw matrix tra coppie di time series. Come vincolo è stato utilizzato Sakoe Chiba con raggio 10 che impiega circa un terzo del tempo necessario con Itakura con slope 2; questo nonostante entrambi i constraint impongano un dtw path di circa 900 celle sulle 2500 della matrice completa (50x50). In figura 3.3 è mostrata la dtw matrix con la Sakoe Chiba band tra le tracce di ID 2 e 5. Le time series sono poi allineate tramite il dtw path trovato (figura 3.4).

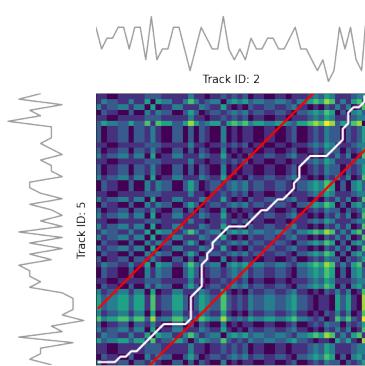


Figure 3.3: DTW path con Sakoe Chiba band

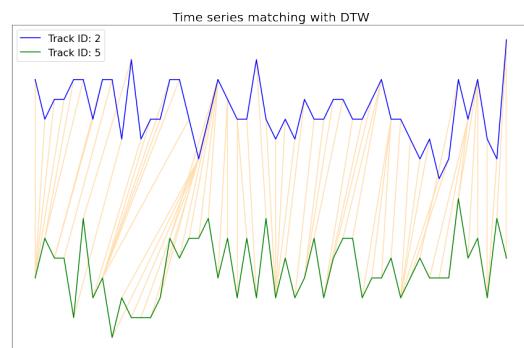


Figure 3.4: Allineamento con DTW

Con il dataset approssimato sono stati provati valori di K da 2 a 30, in modo da selezionare il numero di cluster migliore per il KMeans. Nel grafico in figura 3.6 si può notare l'andamento della silhouette che tende a decrescere raggiungendo, anche in questo caso, score piuttosto bassi. Infatti selezionando $K=8$ che coincide col gomito (tenendo conto anche della silhouette) si ottiene uno score di 0.038. Di fatto dalla rappresentazione in 2 dimensioni con PCA (figura 3.5) del dataset approssimato con SAX si può notare come i dati abbiano una forma globulare perciò dividendo l'agglomerato si ottiene una riduzione in termini di coesione e separazione.

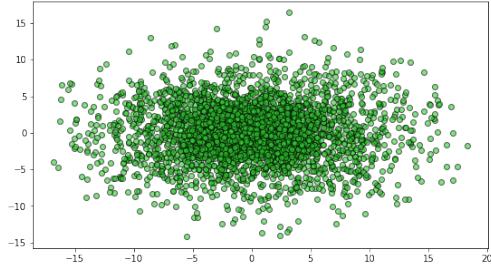


Figure 3.5: PCA del dataset approssimato con SAX

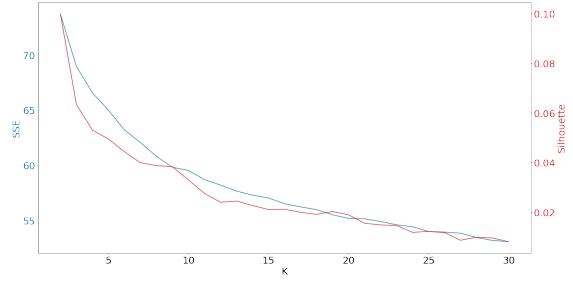


Figure 3.6: SSE/Silhouette curve

3.2.2 Hierarchical Clustering

Per il clustering gerarchico è stato utilizzato il metodo di Linkage Complete (reputato migliore rispetto al Single e Average sulla base dei dendrogrammi ottenuti) a cui è stata passata la cross-similarity matrix del dataset con metrica DTW e Sakoe Chiba band di 10. Il dendrogramma suggerisce un taglio a distanza 26 con la formazione di 4 cluster (figura 3.7).

I risultati ottenuti con questo tipo di clustering non apportano un grosso miglioramento rispetto ai precedenti con una silhouette di 0.045

3.2.3 Features-based Clustering

Inoltre si è provato il clustering features-based trasformando le time series in 15 feature (visibili in figura 3.9) che sono state normalizzate mediante lo Standard Scaler. La metrica utilizzata è stata l'Euclidea (risultata migliore dalle analisi rispetto alla Manhattan). Il clustering in questione restituisce i risultati migliori: eseguendo infatti il KMeans con $K=11$, come suggeriscono le curve dell'SSE e della Silhouette, si ottiene uno score di 0.16. Sebbene questo sia un valore più alto rispetto ai precedenti, non è un valore sufficiente per considerare quest'ultimo un buon clustering. Dall'analisi dei centroidi individuati si può notare come siano vicini tra loro (ad eccezione del cluster 4 che differisce sulle features max e kurtosis), a conferma del fatto che le time series vanno a formare un unico grande cluster la cui frammentazione comporta una diminuzione in termini di silhouette.

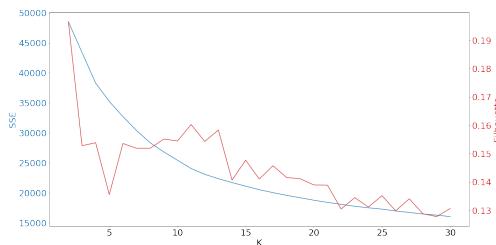


Figure 3.8: SSE/Silhouette curve

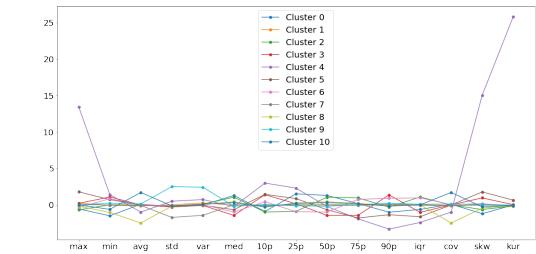


Figure 3.9: Centroidi KMeans

Inoltre le analisi precedenti sono state provate anche senza effettuare l'amplitude scaling cercando di preservare l'ampiezza delle tracce, ed utilizzando come approssimazione PAA che non richiede lo scaling; i risultati ottenuti sono però in linea con i precedenti e quindi non è stato ritenuto necessario riportarli.

3.3 Motifs and Anomalies

In questa sezione sono state invece studiate alcuni motif ed alcune anomalie presenti nel dataset. Ai fini dell'individuazione di motifs all'interno delle varie Time Series, sono state selezionate due tracce (track 2 e track 5) appartenenti allo stesso autore ('AWOL') e con lo stesso genere top ('Hip-Hop'). Rispetto ad entrambe le tracce è possibile osservare:

- I Motifs presenti sulle 2 Time Series (figure 3.10 e 3.11): essi rappresentano per definizione pattern frequenti simili tra loro (evidenziati con lo stesso colore). Ne sono presenti cinque diversi, in quanto nell'algoritmo è stato settato il parametro 'max_motifs' pari a 5. La window utilizzata in questo caso è anch'essa pari a 5, ossia si individuano i Motifs di lunghezza 5 Time Stamps. Come radius, è stato in questo caso utilizzato quello di default (radius = 2).
- Matrix Profile: essa rappresenta in ogni punto la distanza tra la sotto-seguenza (che si origina in quel punto) della Time Series e la più simile tra le altre sotto-seguenze, sempre all'interno della stessa time series; i minimi locali corrisponderanno, quindi, ai Motifs (distanze minime). Il relativo plot è visualizzabile in rosso. A questo proposito, nelle tabelle 3.1 e 3.2 si trovano gli indici iniziali dei motifs e le distanze. In entrambe le tabelle, essendoci solo 2 indici per ogni motif, si può dedurre come il radius pari a 2 non vada a catturare altre sotto-seguenze simili.

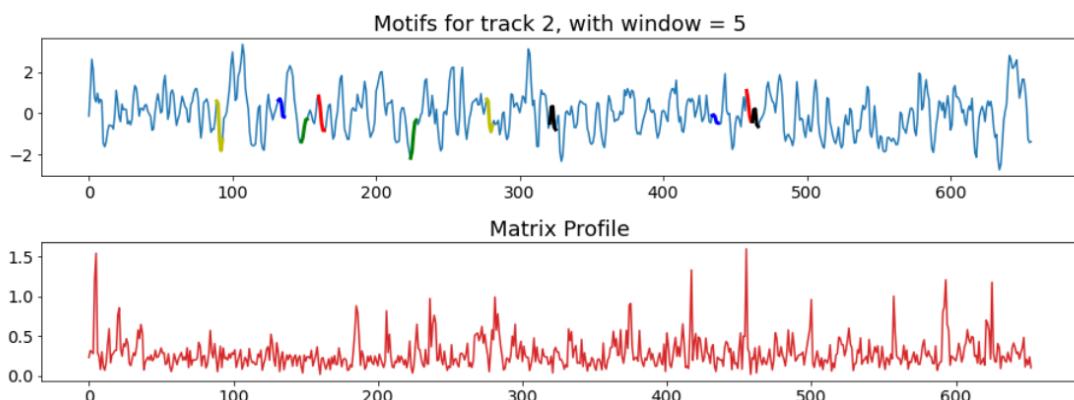


Figure 3.10: Motifs Track 2, $w = 5$

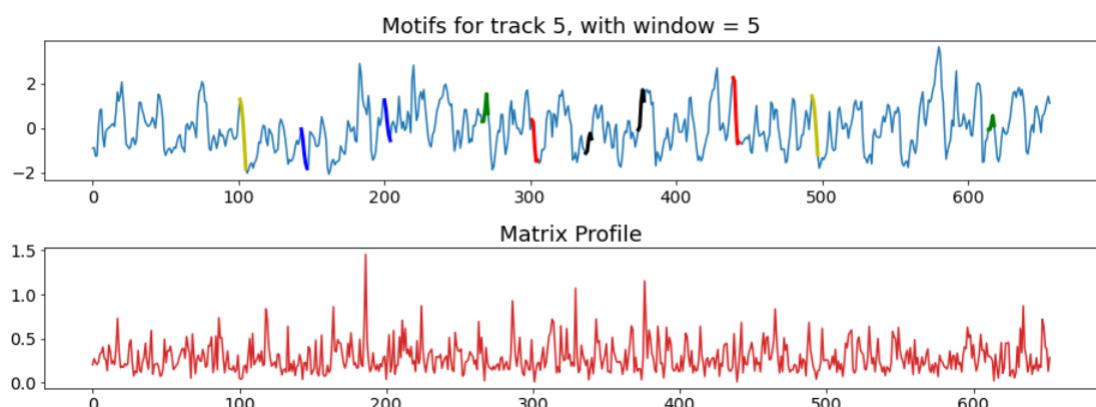


Figure 3.11: Motifs Track 5, $w = 5$

Motifs for Track 2 (w = 5)		
Motif	Top - Motifs Indexes	Distance
1 (In rosso)	[160, 458]	0.02
2 (In verde)	[148, 224]	0.04
3 (In nero)	[321, 462]	0.04
4 (In blu)	[132, 434]	0.06
5 (In giallo)	[89, 277]	0.06

Table 3.1: Motifs for Track 2, w = 5

Motifs for Track 5 (w = 5)		
Motif	Top - Motifs Indexes	Distance
1 (In rosso)	[301, 439]	0.01
2 (In verde)	[267, 614]	0.02
3 (In nero)	[338, 374]	0.03
4 (In blu)	[143, 200]	0.04
5 (In giallo)	[101, 493]	0.04

Table 3.2: Motifs for Track 5, w = 5

Aumentando il parametro della window da 5 a 10, è possibile notare un cambiamento nelle motifs con le distanze che aumentano (figure 3.12, 3.13 e tabella 3.3). Ciò che emerge da questa analisi è che, al diminuire della window, diminuiscono le distanze tra le sotto-sequenze: ciò indica che, utilizzando una window molto piccola, l'analisi perderebbe di significato in quanto il numero di Motifs con distanza minima sarebbe piuttosto elevato (e questo porterebbe a perdere di vista il concetto di Motif). Dalla tabella si nota anche che in questo caso, con w=10, il radius uguale a 2 va a catturare altre sottosequenze; questo può essere spiegato dall'aumento della distanza, rispetto alle motif con w=5, che porta a ricercare altre sotto-sequenze in un intorno maggiore.

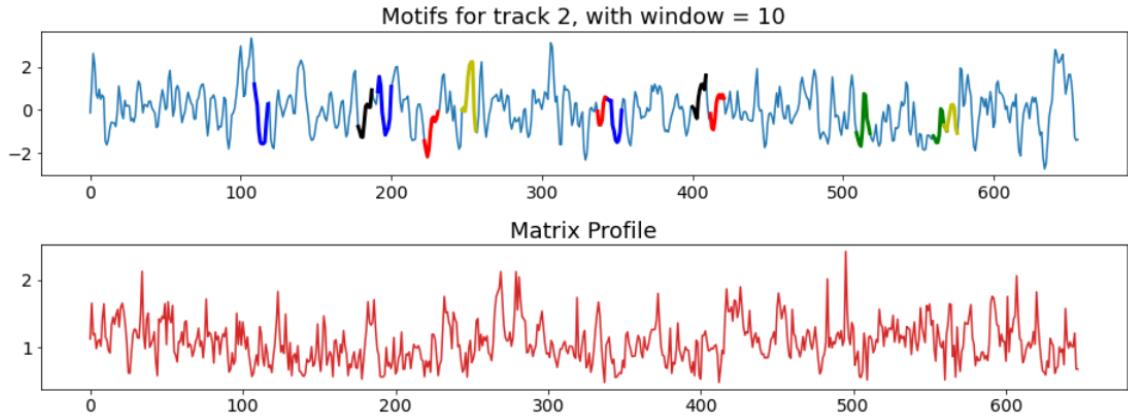


Figure 3.12: Motifs for Track 2, w = 10

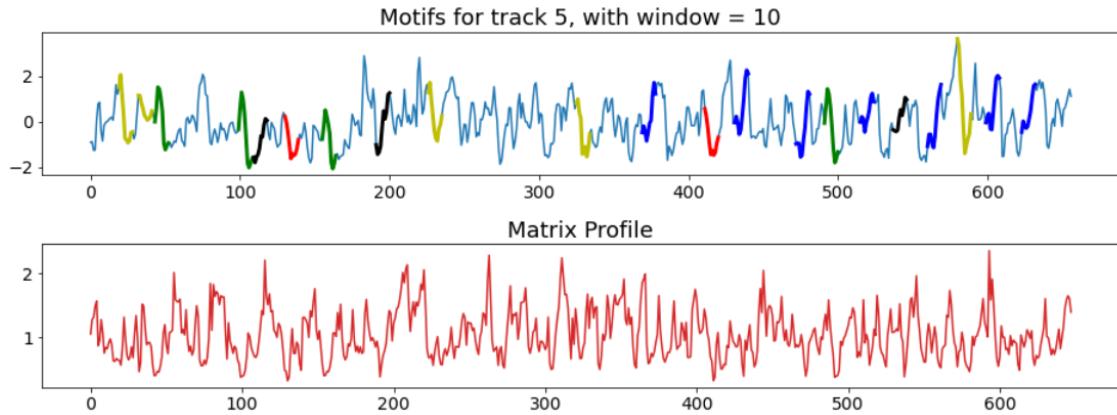


Figure 3.13: Motifs for Track 5, w = 10

Motifs for Track 2 (w = 10)		
Motif	Top - Motifs Indexes	Distance
1 (In rosso)	[222, 337, 412]	0.48
2 (In verde)	[509, 560]	0.52
3 (In nero)	[178, 400]	0.53
4 (In blu)	[109, 191, 344]	0.55
5 (In giallo)	[247, 567]	0.56

Table 3.3: Motifs for Track 2, w = 10

Motifs for Track 5 (w = 10)		
Motif	Top - Motifs Indexes	Distance
1 (In rosso)	[130, 411]	0.32
2 (In verde)	[43, 99, 155, 491]	0.38
3 (In nero)	[191, 537]	0.39
4 (In blu)	[431, 472, 515, 560, 599, 623]	0.4
5 (In giallo)	[32, 226, 325, 580]	0.47

Table 3.4: Motifs for Track 5, w = 10

Sempre attraverso le stesse 2 tracce audio, sono state anche ricercate le anomalie; esse sono state individuate utilizzando le stesse matrix profile realizzate per i motif con w=5 e sono qui riportate in figura 3.14 e in figura 3.15. Coerentemente con quanto esposto sopra, in corrispondenza dei picchi della Matrix profile si trovano le anomalies. I parametri utilizzati sono 'ex_zone' = 3 e k = 5.

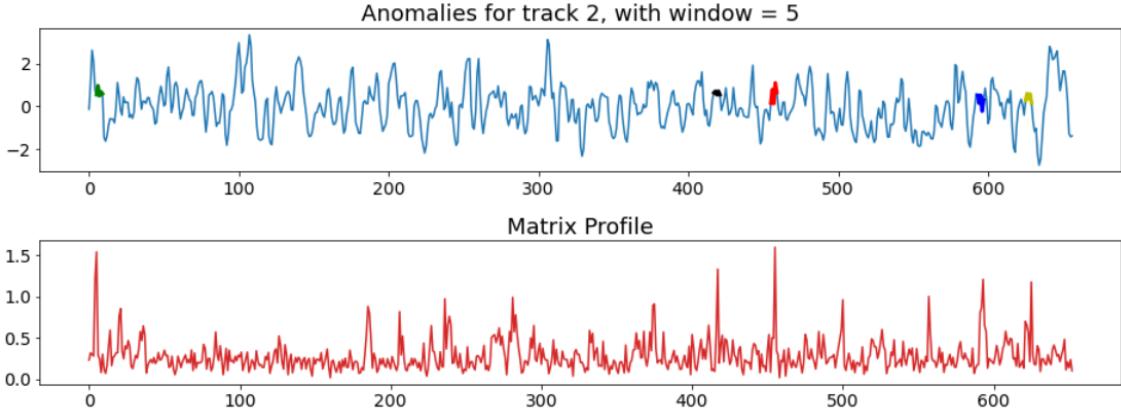


Figure 3.14: Anomalies for Track 2, w = 5

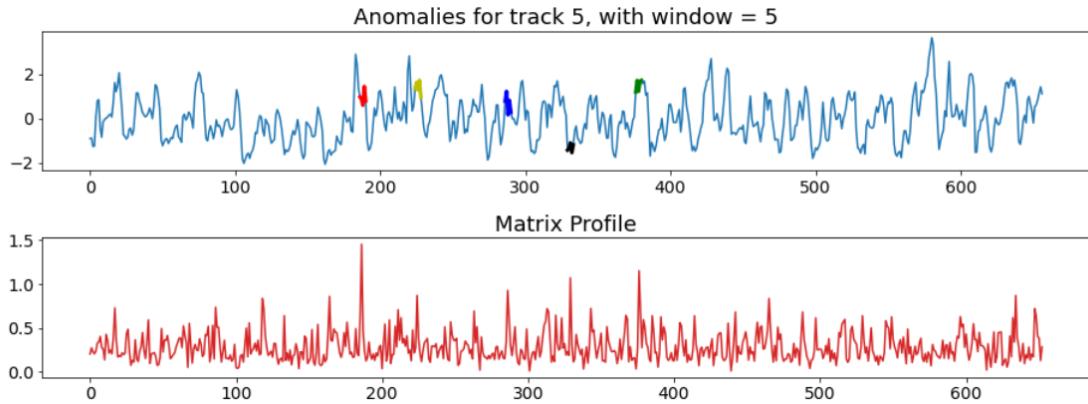


Figure 3.15: Anomalies for Track 5, w = 5

3.4 Time Series Classification

In questa sezione del progetto è stato realizzato il task dellla classificazione utilizzando le time series a disposizione. Anche qui l'obbiettivo è stato quello di riconoscere il genere top di una traccia audio, considerando però solamente 3994 traccie appartenenti a 4 generi differenti. Come si può osservare dalla tabella 3.5, il dataset a disposizione, a cui è stato applicato l'amplitude scaling, risulta bilanciato per le 4 classi.

Esistono diversi metodi per effettuare una classificazione basata sulle time series; il più caratteristico è sicuramente individuare le shapelet ed eseguire una classificazione basata su di esse. Per individuare le shapelet è stata utilizzata la libreria 'tslearn', che mette a disposizione un metodo per creare le shapelet andando direttamente a realizzare un modello attraverso la Logistic Regression. Questo metodo permette quindi sia di avere a disposizione delle shapelet, sia di avere già una prima classificazione basata su di esse.

E' però necessario settare alcuni parametri, tra cui la lunghezza ed il numero di shapelet da realizzare: per questa scelta, è stato utilizzato un altro metodo della libreria 'tslearn', che attraverso una funzione euristica, restituisce il numero di shapelet e la loro lunghezza, basandosi sui dati a disposizione. Anche in questo metodo però, è presente un parametro con cui selezionare il numero voluto di shapelet di lunghezza diversa. Diversi risultati derivati da questo metodo euristico sono stati testati in una grid search, assieme ad altri parametri necessari per la classificazione. Tutti i parametri provati, con i migliori risultati, sono riportati in tabella 3.6, mentre le performance della Logistic Regression sono riportate in tabella 3.7.

Param	Range	Best
n_shapelets_per_size	{[32: 6], {32: 6, 64: 6} {32: 6, 64: 6, 96: 6}, {32: 6, 64: 6, 96: 6, 128: 6]}	{32: 6, 64: 6, 96: 6}
optimizer	['Adam', 'sgd']	Adam
max_iter	150	150
batch_size	[30,100]	30
weight_regularizer	[1, 0.1, 0.01, 0]	0.01

Table 3.6: Parametri Logistic Regression

	Accuracy	F1	Recall	Precision
Train	0.52	0.51	0.52	0.52
Test	0,47	0,47	0,48	0,47

Table 3.7: Performance Logistic Regression

Grazie alla grid search, è stato quindi possibile appurare che la miglior classificazione avviene andando a considerare 18 shapelet totali: 6 di lunghezza 32, 6 di lunghezza 64 e 6 di lunghezza 96. Nelle figure 3.16, 3.17 e 3.18 sono riportate queste 18 shapelet (divise per una migliore visualizzazione), confrontandole con la time series della traccia audio di indice 2.

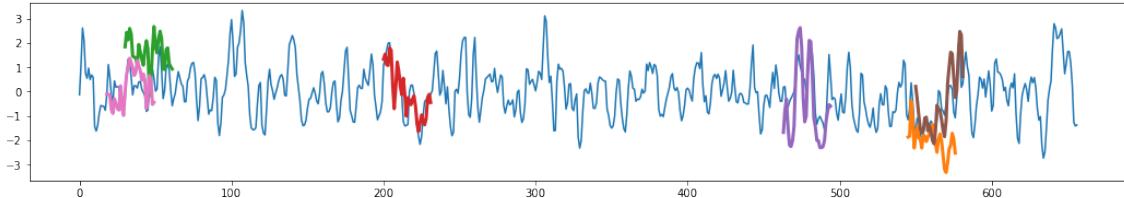


Figure 3.16: Le 6 shapelet di lunghezza 32 create, confrontate con la time series della traccia audio di indice 2

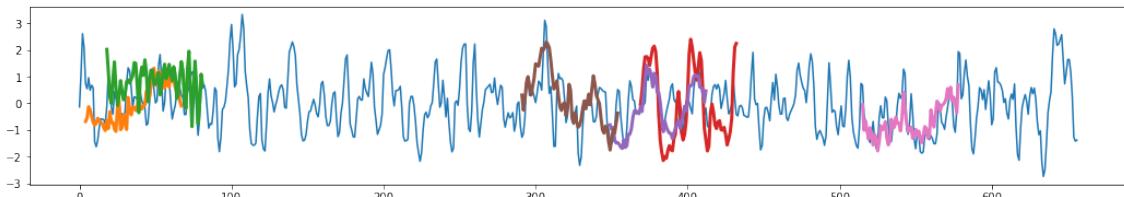


Figure 3.17: Le 6 shapelet di lunghezza 64 create, confrontate con la time series della traccia audio di indice 2

Genere Top	Time Series
Rock	999
Experimental	999
Electronic	999
Hip-Hop	997

Table 3.5: Distribuzioni dei generi top tra le time series

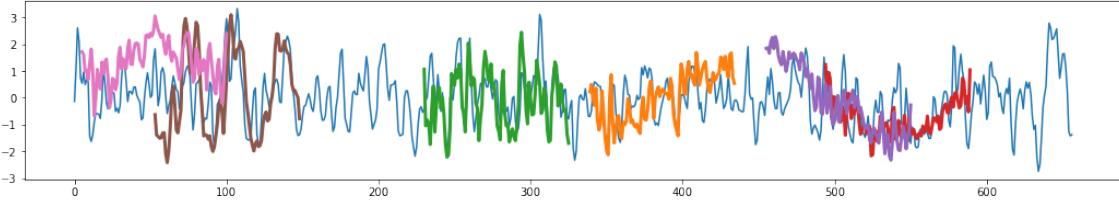


Figure 3.18: Le 6 shapelet di lunghezza 96 create, confrontate con la time series della traccia audio di indice 2

Grazie a queste 18 shapelet, è stato poi possibile creare un dataframe con 18 attributi per ogni traccia audio: ogni valore corrisponde alla distanza tra la time series della traccia ed una delle diverse shapelet. A questo nuovo dataframe sono stati poi applicati i metodi di classificazione KNN, Decision Tree e SVM, con i parametri e le performance riportati nelle tabelle 3.8, 3.9, 3.10, 3.11, 3.12 e 3.13.

Param	Range	Best
n_neighbors	range(3,70)	43
weights	['uniform', 'distance']	uniform
p	[1,2]	1

Table 3.8: Parametri KNN

	Accuracy	F1	Recall	Precision
Train	0,54	0,53	0,54	0,55
Test	0,48	0,48	0,49	0,49

Table 3.9: Performance KNN

Param	Range	Best
criterion	['gini', 'entropy']	entropy
min_samples_split	range(10,70,10)	10
min_samples_leaf	range(10,70,10)	60
max_depth	range(1, 20)	5

Table 3.10: Parametri Decision Tree

	Accuracy	F1	Recall	Precision
Train	0,52	0,52	0,52	0,53
Test	0,45	0,45	0,46	0,46

Table 3.11: Performance Decision Tree

Param	Range	Best
C	[0.0001, 0.001, 0.1, 1, 100, 1000, 10000]	1
kernel	['rbf', 'linear']	rbf
gamma	['scale', 'auto']	auto

Table 3.12: Parametri SVM

	Accuracy	F1	Recall	Precision
Train	0.55	0.54	0.55	0.55
Test	0.49	0.48	0.49	0.49

Table 3.13: Performance SVM

Come ultimo metodo di classificazione, si è provato ad estrarre le feature, intese come valori statistici (sono le stesse estratte per il clustering), dalle time series. In questo modo è stato ottenuto un dataframe di 15 colonne a cui è stato applicato il Decision Tree. Con questo procedimento è stata quindi eseguita una classificazione sulle time series senza utilizzare le shapelet. In tabella 3.14 sono riportati i parametri selezionati, mentre in tabella 3.15 sono riportati le performance del modello.

Param	Range	Best
criterion	['gini', 'entropy']	gini
min_samples_split	range(10,70,10)	10
min_samples_leaf	range(10,70,10)	20
max_depth	range(1, 20)	3

Table 3.14: Parametri Decision Tree sulle features estratte

	Accuracy	F1	Recall	Precision
Train	0.39	0.38	0.39	0.42
Test	0.35	0.33	0.35	0.36

Table 3.15: Performance Decision Tree sulle features estratte

I 5 metodi di classificazione visti non sembrano, dai risultati, riuscire a discriminare in modo efficace il genere di una traccia audio. Infatti nessuno dei 5 riesce a raggiungere un'accuracy maggiore di 0.50, a fronte di un valore di assegnamento random di 0.25 (dataset bilanciato con 4 label).

Modulo 4

4.1 Sequential Pattern Mining

Nella seguente sezione verrà descritta l'analisi dei sequential pattern individuati nel dataset di Time Series utilizzato nel Modulo 3. E' stato opportuno però, per una migliore efficienza computazionale, utilizzare solo un sottoinsieme di 400 Time Series, equivalenti a 100 per ognuno dei 4 generi presenti (Rock, Hip-Hop, Electronic e Experimental).

Anche in questo caso è stato effettuato l'amplitude scaling necessario per trasformare le Time Series con SAX. Ottenute le Time Series, approssimate con 50 segmenti e dimensione dell'alfabeto 8 (parametri utilizzati anche per il clustering), si è proceduto all' individuazione dei sequential pattern. In figura 4.1 è mostrato come varia la support del più frequente sequential pattern al crescere della lunghezza della sequenza; il risultato è ottenuto utilizzando il parametro filter della funzione topk di PrefixSpan. Si passa dalla 1-sequence composta dal solo simbolo 3 con support 400, e cioè presente in tutte le Time Series, alla 10-sequence più frequente con support 212. Si è inoltre notato che la presenza dei simboli nel dataset è sbilanciata (tabella 4.1). Ciò è una diretta conseguenza dell'approssimazione che tenderà ad assegnare ai segmenti i simboli appartenenti alla fascia centrale. In figura 4.2 è riportato l'esempio di una Time Series con l'approssimazione in SAX (le linee tratteggiate verdi sono i breakpoint che delimitano i simboli).

Simbolo	Support	Simbolo	Support
[3]	400	[6]	385
[2]	396	[1]	382
[5]	394	[7]	301
[4]	394	[0]	228

Table 4.1: Support Singleton

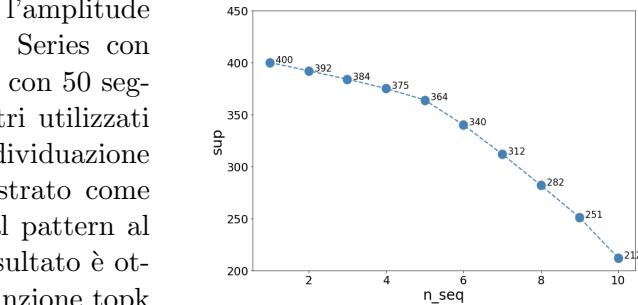


Figure 4.1: Max support n-sequence con n in(1,10)

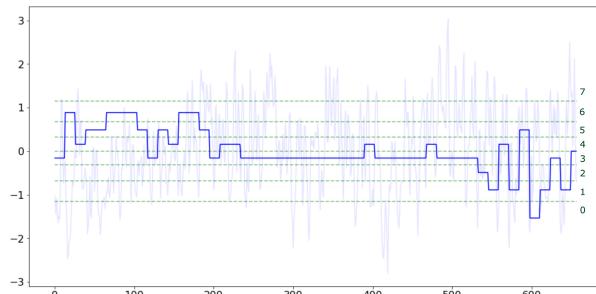


Figure 4.2: Rappresentazione SAX Time Series

Prendendo in considerazione le top 5 8-sequence, si è osservato in quali Time Series fossero presenti, svolgendo un'analisi volta a capire se ci fossero dei pattern prevalenti in alcuni generi. In tabella sono mostrati i sequential pattern trovati e la rispettiva support. Dal barplot si può notare come per ogni pattern (asse x) la distribuzione dei generi delle Time Series contenenti quel pattern rimanga pressochè la stessa; il genere Hip-Hop sembra essere più caratterizzato dai sequential pattern trovati rispetto a Rock ed Experimental, anche se la support di questi ultimi rimane intorno a 60 su 400.

P	Pattern	Support
P1	[3, 3, 3, 3, 3, 3, 3, 3]	282
P2	[3, 3, 3, 3, 4, 4, 3, 3]	276
P3	[3, 3, 3, 3, 3, 3, 4, 4]	273
P4	[3, 3, 3, 3, 3, 4, 3, 3]	273
P5	[3, 3, 3, 4, 3, 3, 3, 3]	273

Table 4.2: Top 5 8-sequence

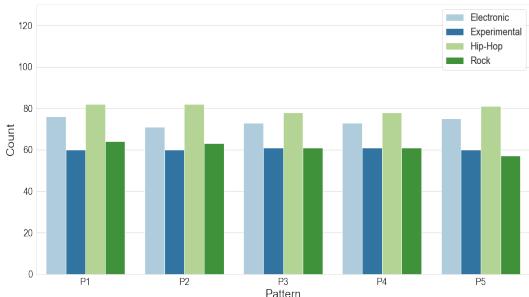


Figure 4.3: Distribuzione generi sequential pattern

4.2 Advanced Clustering

In questa sezione sono state confrontate tra di loro, tecniche di clustering tradizionali ed avanzate, al fine di individuare meglio le strutture nascoste nei dati. A tal proposito è stato ripreso il dataset utilizzato nei moduli 1 e 2, privo degli outliers (eliminati nel modulo 1). Il dataset è quindi composto da 20 attributi numerici continui e 24752 record rappresentanti tracce audio. Prima delle analisi i dati sono stati normalizzati utilizzando il metodo dello standard scaler.

4.2.1 K-Means vs X-Means

Inizialmente sono stati provati i due metodi K-Means e X-Means. Al fine di individuare il giusto numero di K per il K-Means, sono stati provati i valori in un range da 2 a 50 e per ogni k è stato calcolato il SSE e la Silhouette. Come si può vedere dalla figura 4.4, la silhouette ha valori molto bassi e decresce all'aumentare di K. Riportando i dati in due dimensioni con PCA ed eseguendo un clustering con K=2 (figura 4.5), si può notare come l'individuazione di 2 cluster comporti la divisione dell'unico grande agglomerato di punti.

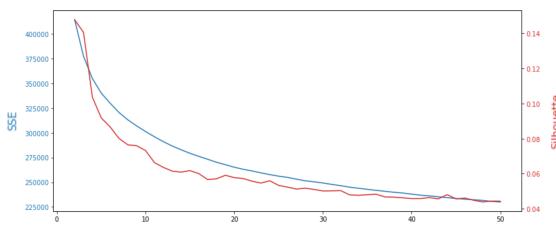


Figure 4.4: SSE/Silhouette curve

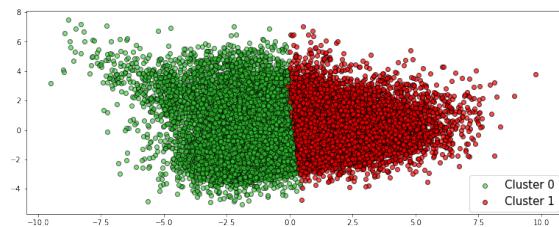


Figure 4.5: PCA K-Means con k=2

Successivamente è stato testato sugli stessi dati il metodo X-Means che utilizza il Bayesian Information Criterion (BIC) per bisezionare i cluster. Il miglior valore di BIC viene ottenuto con k-max=39. Il valore della Silhouette con K=39 precipita però a 0.05, indicando di fatto una divisione in cluster errata. Per i ragionamenti già effettuati per il K-Means, anche in questo caso sembra che nessuna divisione sia la miglior soluzione.

In figura 4.6 è riportato il clustering ottenuto fissando K-MAX=10; la divisione dell'agglomerato, con i limiti della visualizzazione ottenuta con PCA, non sembra la scelta migliore, e questo viene anche confermato dal valore della silhouette pari a 0.08.

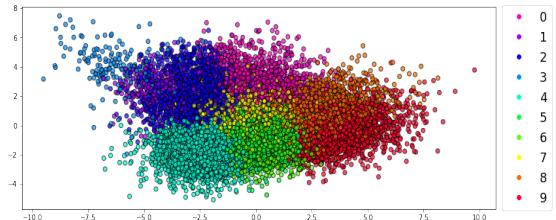


Figure 4.6: PCA X-Means con k=10

4.2.2 DbScan vs Optics

Come seconda analisi è stato eseguito un clustering dei dati in base alla densità, sono stati quindi utilizzati i metodi DbScan e Optics. Per quanto riguarda il DbScan, inizialmente è stato ricercato un range di valori per epsilon andando a calcolare la distanza del k-esimo vicino di ogni punto.

Variando il valore di k, è stato osservato un range di epsilon tra 4 e 5 (Figura 4.7). Successivamente sono stati testate alcune combinazioni di epsilon e Min_Samples al fine di ottenere un valore di silhouette elevato. Dal grafico di figura 4.8 si nota che per epsilon=4 e Min_Samples=20 si ottiene una silhouette pari a 0.41. Questo valore di silhouette è il più alto ottenuto e va a creare un grande cluster con 24570 record e 182 noise points.

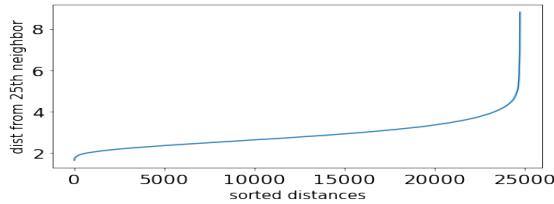


Figure 4.7: Distanza di ogni punto dal suo 25esimo punto più vicino

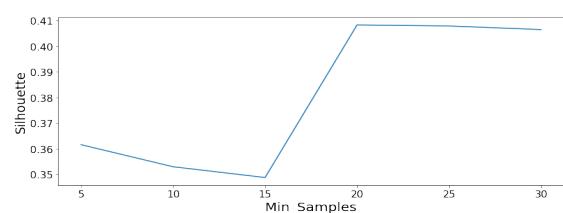


Figure 4.8: Variazione della silhouette con epsilon=4

Per Optics invece non è stato necessario scegliere il valore di epsilon che è stato fissato con valore 'infinito'. Sono stati provati alcuni valori diversi di Min_Samples, in particolare in un range da 5 a 35; dai diversi grafici del Reachability Plot prodotti, non sembrano però esserci differenze. Il clustering automatico eseguito con il metodo 'XI' restituisce sempre un solo cluster con tutti i record. Analizzando il Reachability Plot riportato in figura 4.9, è evidente la presenza di un solo cluster; una threshold intorno al valore di 5 potrebbe individuare alcuni noise points. Questo andrebbe a produrre un risultato molto simile a quello ottenuto con il DbScan.

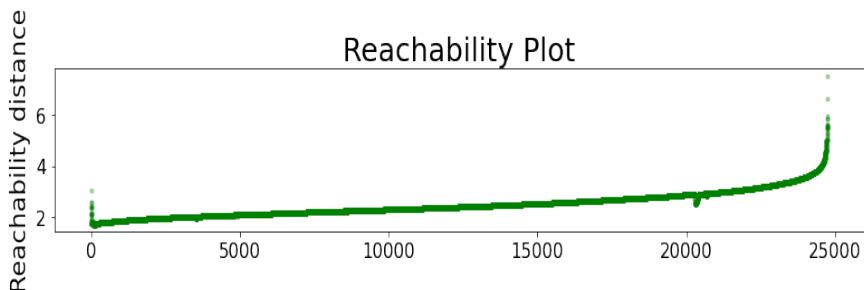


Figure 4.9: Reachability Plot con Min_Samples=35

4.3 Transactional Clustering

Dopo aver utilizzato le tecniche di clustering per gli attributi numerici, è stato qui realizzato anche il clustering con attributi categorici. Gli attributi sono stati selezionati dal sub-dataset 'tracks'; è stato inizialmente ridotto il dataframe andando a considerare solo 8000 record, quelli cioè con attributo ('set', 'subset') = 'small', e successivamente si è passati alla selezione degli attributi tentando di individuare quelli con un minor numero di null value. La scelta è ricaduta su 5 attributi: ('album', 'title'), ('album', 'type'), ('artist', 'name'), ('track', 'genre_top') e ('track', 'license'). E' stato poi deciso di ridurre i record a 7764 a causa di alcuni null value presenti in ('album', 'type') e in ('track', 'license'). Esaminando questi attributi attentamente, è stato inoltre notato che ('artist', 'name') e ('album', 'title') presentano rispettivamente 2306 e 2417 valori diversi; un range così ampio potrebbe non essere l'ideale per il clustering.

4.3.1 K-Modes

Il clustering su questi dati è stato eseguito con la tecnica del K-Modes. Utilizzando l'algoritmo con numero di inizializzazioni diverse uguale a 5 e massimo 200 iterazioni per ogni k, è stato cercato il miglior numero di cluster k per questi dati. La ricerca è stata fatta calcolando ogni volta il 'cost' (cioè la somma dei mismatch, sugli attributi, dei record all'interno di ogni cluster rispetto

all'oggetto rappresentativo del cluster) e la silhouette, calcolata utilizzando la Jaccard distance. In figura 4.10 è riportato il grafico con le curve del cost e della silhouette al variare del numero di cluster tra 2 e 50. Dal grafico si osserva che il valore del cost tende a diminuire aumentando k, con la curva che diventa meno inclinata con k circa 10. La silhouette invece varia anche se sembra avere un trend crescente con l'aumento dei cluster. In ogni caso non supera lo 0.1, per i valori di K qui testati, indicando un clustering non buono. Questo risultato della silhouette può essere spiegato dall'ampio range di valori di alcuni attributi considerati; aumentando la divisione in cluster si avrà sempre più attributi simili all'interno del cluster e quindi un miglioramento della silhouette calcolata con la Jaccard distance. Per verificare questo trend, è stato calcolato anche il cost e la silhouette per valori di k fino a 500 (solo per multipli di 5). In figura 4.11 si può osservare il trend della silhouette che cresce sempre di più arrivando quasi a 0.30.

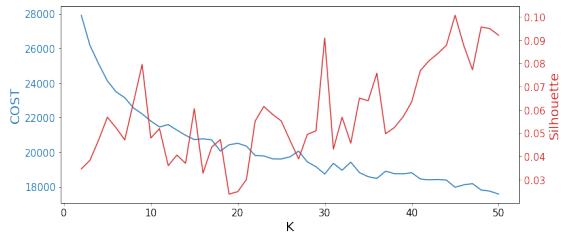


Figure 4.10: Cost e Silhouette al variare di K da 2 a 50

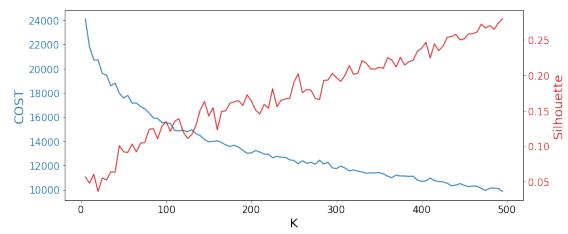


Figure 4.11: Cost e Silhouette al variare di K da 5 a 500

Nonostante questi risultati è stato deciso di utilizzare un k=9; questa scelta è stata fatta in base al fatto che un elevato numero di cluster porterebbe a dei risultati poco significativi. Con k=9 la silhouette è 0.08, non un buon valore ma motivato da quanto riportato precedentemente. In tabella 4.3 sono riportati i 9 object rappresentativi dei cluster ottenuti. Da un'analisi visiva si può notare subito l'assenza del genere top 'Instrumental', l'unico genere che risulta mancante degli 8 presenti nei dati. Si ripetono 2 volte invece i generi 'International' e 'Folk': troviamo in entrambi i casi una variazione poi sul tipo di album. Anche su (track, license) troviamo alcuni valori comuni a cluster diversi, mentre su (artist, name) e (album, title), i due attributi che presentano un ampio range di valori, non compaiono mai valori in comune tra coppie di cluster.

	Record	(track, genre_top)	(artist, name)	(album, title)	(album, type)	(track, license)
1	1622	Pop	TRG Banks	Carcrashlander Instrumentals	Album	A-NC 3.0 International
2	1327	Rock	Scott Holmes	Drop a 4 note Soundtrack	Album	A-NC-ShareAlike 3.0 International
3	1022	Experimental	Blue Dot Sessions	This Is The End, Beautiful Friend	Album	A-NC
4	1107	Hip-Hop	The Impossebulls	#JustBecause	Album	A-NC- ShareAlike
5	758	Electronic	Jason Shaw	Audionautix: Tech, Urban, Dance	Album	A-NC-ShareAlike 3.0 International
6	427	Folk	Alash Ensemble	Live on WFMU's Transpacific Sound Paradise, Jan 22, 2011	Radio Program	A-NC-No Derivative Works 3.0 United States
7	394	International	The Sounds of Taraab	Live at the 2013 Golden Festival	Live Performance	A-NC-Share Alike 3.0 United States
8	592	Folk	Big Blood	Pas de tigre en afrique ?	Album	A-NC-Share Alike 3.0 United States
9	515	International	Sláinte	Live at the 2014 Golden Festival	Album	A-NC-NoDerivatives 3.0 International

Table 4.3: Object rappresentativi dei 9 cluster ottenuti