Data Mining Project

RAVDESS Dataset Analysis

Author: *Biancamaria Bombino, Alex Degaudenzi, Sara Hakim*

Instructor:	*Guidotti, Spinnato*
Master's degree:	Data Science & Business Informatics
Date:	7 giugno 2023

Indice

1. Introduzione 2
2. Data Preparation 2
3. Imbalanced Learning 2
4. Dimensionality Reduction 5
5. Anomaly Detection 6
6. Advanced Classificators Methods 9
    6.1 Logistic Regression 9
    6.2 Support Vector Machine 11
    6.3. Neural Networks 12
    6.4 Ensamble Methods 14
    6.5 Gradient Boosting 16
7. Advanced Regression 16
8. Multiclass Emotions Experiments 17
9. Time Series Analysis 18
    9.1 Data Understanding and Preparation 18
    9.2 Clustering 18
    9.3 Motifs and Anomaly Discovery 21
    9.4 Classification 23
10. Explainability 28
11. Conclusioni Generali 29

RAVDESS Dataset Results

1. Introduzione

Il Ryerson Audio-Visual Database of Emotional Speech (RAVDESS) è un dataset composto da una serie di audio digitali, registrati da un numero finito di persone. In particolare, a 24 attori con accento americano è stato richiesto di pronunciare o cantare due frasi lessicalmente corrispondenti. Tali espressioni sono caratterizzate da determinate emozioni e sono enunciate con una certa intensità emozionale. In ambito scientifico, i file audio rappresentano un'innovazione nel settore della ricerca medica. Infatti, questi ultimi vengono impiegati per stimolare il sistema nervoso dei pazienti e analizzare i diversi effetti ottenuti su ogni individuo. Per effettuare le analisi, sono stati considerati due dataset distinti: il train set e il test set.

2. Data Preparation

Il dataset di train contiene 1828 records e 434 features, mentre il test set è formato da 624 records e 434 features. Attraverso diverse tecniche di visualizzazione è emersa la presenza di variabili contenenti esclusivamente valori costanti; in particolare: la variabile categorica *modality* e 51 features numeriche. Per tale motivo si è proceduto alla loro cancellazione in entrambi i dataset, in modo da ridurre le loro dimensioni ed evitare di analizzare dati inconcludenti. Inoltre, è stata eliminata la variabile categorica *actor* in quanto ciascun attore è caratterizzato da un codice identificativo differente, e tali codici sono divisi tra i due dataset a disposizione. Successivamente è stata calcolata la correlazione di Pearson tra gli attributi numerici ottenendo numerose correlazioni, e di conseguenza sono state eliminate 177 features aventi un indice di correlazione strettamente maggiore di 0.90 con altre variabili. Un esempio di alta correlazione è quello tra *frame_count* e le variabili corrispondenti alle lughezze dell'audio. Tra queste due *frame_count* è stata cancellata poichè ritenuta la meno utile per le future analisi. Tutte queste procedure sono state effettuate su entrambi i dataset di train e di test.

3. Imbalanced Learning

I dataset di riferimento sono quelli ottenuti durante la fase di Data Preparation riportata in Sezione 2. Sono state eseguite tutte le tecniche studiate di undersampling, oversampling ed infine il bilanciamento a livello di algoritmo. Le tecniche sono state applicate su due variabili categoriche: *emotion* ed *emotional intensity*. Come prima fase sono state normalizzate le variabile numeriche tramite la *Standard Scaler*, utilizzando la *fit trasform* sul Dataset di train e la *fit* sul Dataset di test.

Emotion

La variabile categorica *Emotion* è stata suddivisa in due categorie: *happy* contenente solo l'emozione felice e *not happy* che comprende tutte le restanti emozioni del dataset. Le due classi ottenute risultano avere uno sbilanciamento originale del 84,7% (*not happy*) e 15,3% (*happy*) nel dataset di train, e uno sbilanciamento simile risulta presente anche nel test set (*not happy* 84,6% e *happy* 15,4%). Nonostante le due classi siano già abbastanza disomogenee, si è provveduto a sbilanciarle ulteriormente per testare le tecniche con uno sbilanciamento più conforme alla realtà, portando la classe maggioritaria al 96% e la classe minoritaria al 4% in entrambi i dataset. Tale sbilanciamento è stato effettuato tramite il metodo *make_imbalance* della libreria *imblearn*. Per le tecniche di undersampling e oversampling (non random) si stratifica in base a *happy* o *not happy*, in modo da creare un sottoinsieme che avrà una distribuzione della variabile *happy* analoga alla popolazione iniziale. Questa tecnica permette di creare dei campioni più fedeli alla popolazione iniziale, ma allo stesso tempo è molto condizionata dalla scelta della variabile per cui stratificare. I risultati ottenuti con le diverse tecniche e utilizzando come classificatore sia il *Decision Tree* sia il *KNN*, sono stati analizzati in termini di *accuracy* e *f1-score*, e sono riportati nella seguenti tabelle 1, 2, 3, 4:

	Random OverSampler	Smote	Adasyn
Accuracy	0.85	0.74	0.81
f1 score classe not happy	0.92	0.85	0.89
f1 score classe happy	0.15	0.09	0.13

Figura 1: Decision Tree Oversampling

	Random UnderSampler	Condensed Nearest Neighbors	Tomek Links	Edit Nearest Neighbours	Cluster Centroids
Accuracy	0.53	0.80	0.94	0.92	0.24
f1 score classe not happy	0.69	0.89	0.97	0.96	0.35
f1 score classe happy	0.07	0.03	0.06	0.08	0.08

Figura 2: Decision Tree Undersampling

	Random OverSampler	Smote	Adasyn
Accuracy	0.93	0.77	0.77
f1 score classe not happy	0.96	0.87	0.87
f1 score classe happy	0.05	0.13	0.12

Figura 3: KNN Oversampling

	Random UnderSampler	Condensed Nearest Neighbors	Tomek Links	Edit Nearest Neighbours	Cluster Centroids
Accuracy	0.55	0.95	0.91	0.88	0.89
f1 score classe not happy	0.70	0.97	0.95	0.93	0.94
f1 score classe happy	0.09	0.06	0.08	0.06	0.14

Figura 4: KNN Undersampling

Analizzando i valori riportati nella tabelle in base alle metriche, in particolare la *f1-score*, è osservabile che la tecnica migliore risulta essere la Random Oversampling e il Condensed Nearest Neighbors per l'Undersampling. Entrambe queste tecniche conducono a risultati migliori quando il classificatore considerato è il KNN. Nel primo caso vengono aggiunti dati fittizzi assegnando ad ogni classe il seguente numero di dati: 1548. Nel caso dell'undersampling i dati vengono ridotti e si ottengono 247 dati per la classe *not happy* e 64 dati per la classe *happy*. Per la scelta dei migliori parametri utilizzati nei classificatori testati, è stata utilizzata la *GridSearchCV*, includendo la *RepeatedStratifiedKFold* con un numero di ripetizioni pari a 10 e un numero di split pari a 5. Nel caso del KNN per il Random Oversampling i parametri migliori restituiti sono: *n_neighbors=1, weights='distance', metric='manhattan'*. Per il CNN dell'Undersampling, i migliori parametri del KNN risultano essere: *n_neighbors=5, weights='distance', metric='manhattan'*. Nelle figure 5 e 6, riportiamo la visualizzazione dei dati tramite PCA e la ROC Curve ottenute dalle due tecniche migliori citate in precedenza; in cui la classe 0 corrisponde a *not happy* e la classe 1 corrisponde a *happy*.

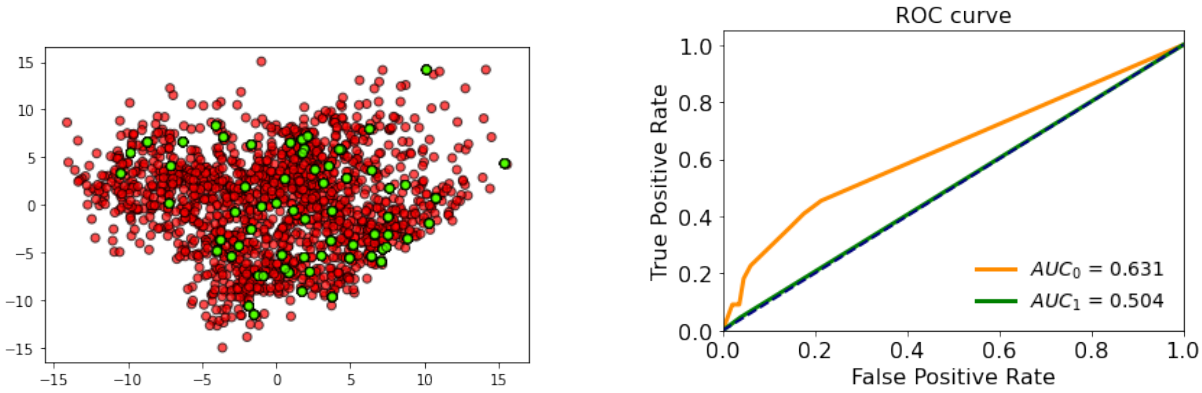

Figura 5: Visualizzazione PCA e ROC Curve Random Oversampling

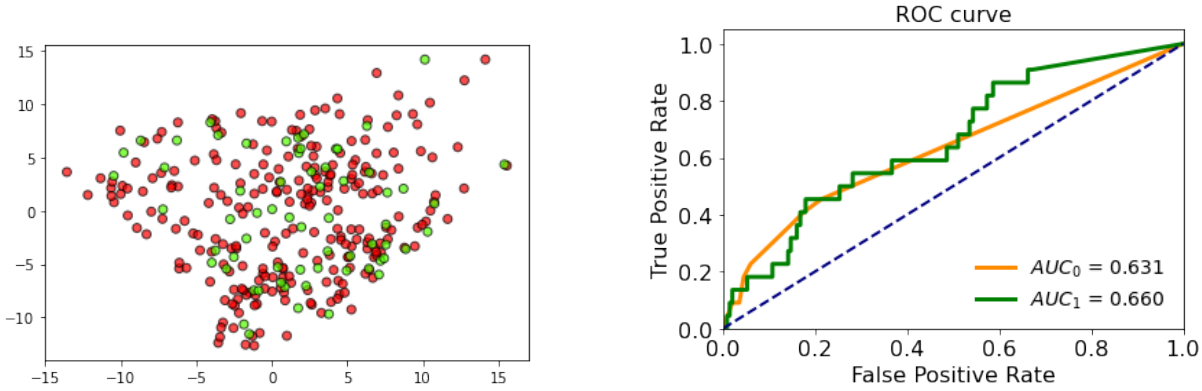

Figura 6: Visualizzazione PCA e ROC Curve CNN

Dalle ROC Curve raffigurate è possibile comprendere come il CNN conduce un test più informativo rispetto al Random Oversampling. Infatti quest'ultimo risulta avere un'area AUC pari al 0.5 e quindi non riesce a discriminare i dati caratterizzati dall'emozione felice. Tuttavia le AUC ottenute dalle roc curve di tutti i metodi testati risultano essere per la maggior parte inferiori di 0.7 e per tale motivo possiamo concludere che queste tecniche applicate al task *happy - not happy* eseguono test poco accurati. Infine è stata provata anche la procedura *class weight* del bilanciamento a livello dell'algoritmo, utilizzando come classificatore il Decision Tree e assegnando peso 1 alla classe *not happy* e peso 4 alla classe *happy* che risulta essere la meno frequente tra i records dei dataset a disposizione. Da questo esperimento sono stati ottenuti i seguenti valori per le metriche considerate: *Accuracy* pari a 0.92 e *F1-score* pari a 0.96 per *not happy* e 0.15 per *happy*.

Emotional Intensity

Un'ulteriore task effettuato con le tecniche di Imblanced Learning è stato svolto con la variabile target *emotional intensity* che risulta essere pressochè bilanciata. Utilizzando le solite procedure viste per la variabile *emotion*, è stato effettuato lo sbilanciamento portando la classe *strong* ad essere la classe minoritaria passando da una distribuzione iniziale del 46.2% al 4%, mentre la classe *normal* rappresenta la classe mag-

gioritaria e si passa da una distribuzione dei dati del 53.8% al 96%. In seguito sono state applicate tutte le tecniche di Undersampling e Oversampling, con entrambi i classificatori Decision Tree e KNN. In questo caso, la tecnica migliore in termini di *F1-score* e *accuracy* risulta essere la *Cluster Centroids* con il KNN. Tramite quest'ultima otteniamo l'*accuracy* pari a 0.93, e l'*F1-score* della classe *not happy* pari a 0.96 e di *happy* uguale a 0.24.

Comprensioni finali

Da queste analisi si è compreso che aumentare lo sbilanciamento del dataset non porta ad un miglioramento delle prestazioni. Infatti dopo aver bilanciato il dataset abbiamo ottenuto risultati leggermente migliori, in particolare per l'*F1-score*. Quest'ultima risulta essere pari a 0 per la classe *happy* con questo sbilanciamento, e perciò la precisione dei classificatori considerati è pessima ed essi non riescono a prevedere bene la classe minoritaria.

4. Dimensionality Reduction

In questa sezione sono state applicate le tecniche di riduzione sui dataset ripuliti, ottenuti in fase di *Data Preparation*. Le variabili numeriche sono state normalizzate tramite la *StandardScaler*. Anche per questo task è stata utilizzata la *GridSearchCV* assistita da una *5-fold Cross Validation* ripetuta 10 volte per ottimizzare i parametri dei classificatori *Decision tree* e *KNN*. Il problema della riduzione della dimensionalità è stato affrontato attraverso i vari approcci della *Feature Selection* e *Feature Projection*. Queste metodologie sono state applicate ai dataset in modo da ridurli ed ottenere lo stesso numero e le stesse features sia per il train set sia per il test set; e in seguito le loro perfomance sono state valutate attraverso i due classificatori considerati. Nelle figure 7 e 8 sono esposti i risultati ottenuti attraverso queste tecniche. Da queste tabelle, possiamo notare come l'*Univariate Feature Selection* con l'utilizzo del metodo *Select K-Best* (che rimuove tutto tranne le k caratteristiche di punteggio più alto) risulta essere la miglior tecnica di Feature Selection in termini di *F1-Score e Accuracy* per la riduzione della dimensionalità. I valori migliori di queste metriche sono stati ottenuti dal classificatore KNN. Per quanto riguarda le tecniche di Feature Projection, la migliore risulta essere la *Principal Component Analysis*. Per quest'ultima la selezione del numero di componenti (pari a 39 dopo la riduzione) è stata effettuata utilizzando principalmente il metodo degli autovalori, che mantiene le componenti principali i cui autovalori sono maggiori di uno. In seguito è stata valutata la PCA ed essa restuisce perfomance migliori a livello di *Accuracy e F1-score* con il Decision Tree.

	Variance Threshold	Select K-best	Recursive Feature Elimination (RFE)	Select from model
Accuracy	0.78	0.78	0.76	0.76
f1 score classe not happy	0.87	0.87	0.86	0.86
f1 score classe happy	0.27	0.33	0.25	0.19

Figura 7: Features selection

	Random subspace projection	PCA
Accuracy	0.77	0.79
f1 score classe not happy	0.86	0.88
f1 score classe happy	0.20	0.21

Figura 8: Features projection

In aggiunta, è stato deciso di effettuare un'ulteriore esperimento con la dimensionality reduction utilizzando il dataset di train sbilanciato nella fase di *Imbalanced Learning* e ribilanciato con una delle migliori

tecniche ottenute precedentemente, ovvero il Condensed Nearest Neighbor. In questo modo otteniamo un dataset di train composto da circa 311 records. E' stata scelta una procedura di Undersampling poichè in questo modo viene evitata la creazione di dati fittizi che sarebbe avvenuta in caso di Oversampling e che avrebbe potuto deviare le analisi eseguite. Applicando le stesse tecniche di riduzione della dimensionalità citate in precedenza, risulta essere la *Variance Threshold* la miglior tecnica. Con quest'ultima si ottengono con il Decision Tree valori di Accuracy pari a 0.84 e di F1-score pari rispettivamente a 0.92 per la classe *not happy* e 0.12 per *happy*. Concludendo si può notare come i valori migliori per la dimensionality reduction vengono ottenuti quando il dataset utilizzato è quello originale solamente ripulito.

5. Anomaly Detection

In questa Sezione sono stati individuati l'1% di outliers, applicando più metodi di diverse famiglie per la ricerca di anomalie. Le tecniche utilizzate per il rilevamento dei valori anomali sono: *HBOS, Elliptic Envelope, KNN, DBSCAN, LOF, ABOD, LODA, Likelihood Approach, Isolation Forest e Extended Isolation Forest*. Per le tecniche basate sulla distanza, le features numeriche sono state normalizzate tramite la tecnica *StandardScaler*.

Anomaly detection - Dataset originale ripulito

Le analisi sono state effettuate sul dataset di train ottenuto in fase di Data Preparation. Inizialmente è stata definita una funzione per trovare gli outliers basata su IQR. Questa procedura ritorna il numero di possibili outliers per ogni variabile, e tra i risultati ottenuti si osserva che la variabile *lag1_sum_w1* presenta il maggior numero di outliers pari a 734. Rappresentando tali outliers nel boxplot si osserva che questi si presentano quando la feature considerata assume valori maggiori di 1 e minori di -1 e sono presenti maggiormente quando la variabile *emotion* è diversa da *happy*. Tuttavia questa funzione ha evidenziato più dell'1% delle anomalie, poiché l'IQR considera ogni feature individualmente e non giudica l'anomalia di un record nell'intero contesto. Per le successive indagini, è stato invece utilizzato negli algoritmi un parametro di contaminazione pari a 0.01 in modo da selezionare solo una frazione di punti dati anomali pari all'1% dei dati totali. Tuttavia, alcuni metodi tipo il DBSCAN non comprendono questa opzione e per tale motivo è stato eseguito l'hyper-parameter tuning per trovare la miglior configurazione dei parametri. I metodi HBOS, ELLIPTIC ENVELOPE, KNN, LOF, LODA e ABOD evidenziano esattamente 19 outliers. Per quanto riguarda l'ISOLATION FOREST abbiamo utilizzato sia la versione di *sklearn* sia della libreria *eif*. Quest'ultima consente di costruire anche una foresta di isolamento estesa. La costruzione delle foreste è avvenuta prendendo in considerazione gli stessi valori per i parametri: *ntrees* ovvero il numero di alberi da utilizzare per l'adattamento della foresta e pari a 600, e *sample_size* il numero di righe da sottocampionare nella creazione di ciascun albero pari a 256. Quest'ultimo deve essere inferiore al numero di osservazioni nel set di dati. Da questo studio è risaltato che con *sklearn* l'isolation forest evidenzia 19 outliers, mentre con *eif* non viene catturato nessun outlier né con la foresta normale né con l'estesa. Di conseguenza si deduce che *sklearn* permette di rimarcare meglio le anomalie. Nella figura 9 vengono riportate una parte delle immagini raffiguranti la visualizzazione degli outliers in 3D Scatter Plot usufruendo della PCA come tecnica di dimensionality reduction.

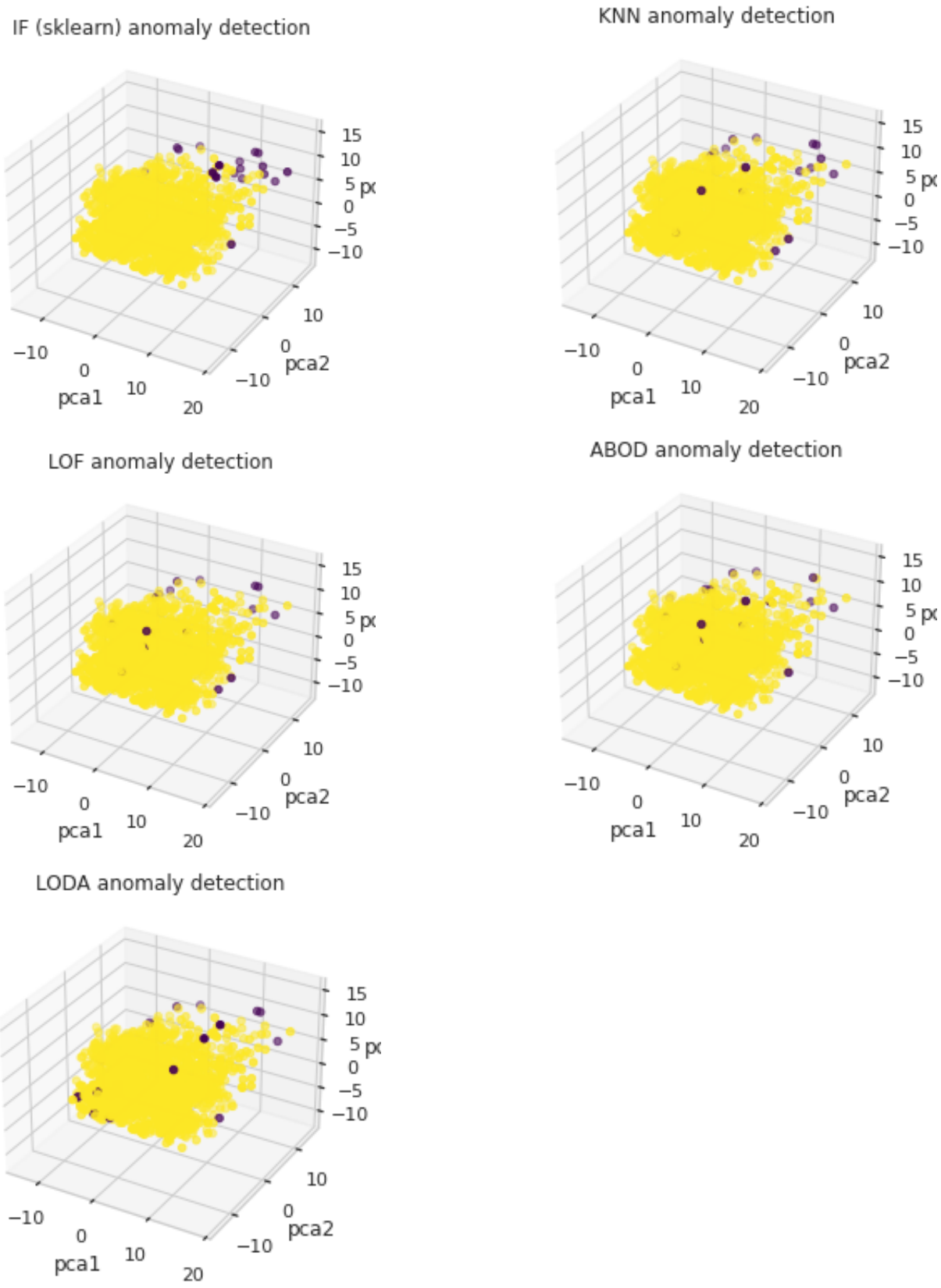

Figura 9: Scatter Plot dei metodi di Anomaly Detection: Isolation Forest, KNN, LOF, ABOD e LODA. In giallo sono rappresentati gli inliers e in viola l'1% degli outliers.

Come ulteriore esperimento è stato utilizzato il *Likelihood Approach* che assegna un punteggio che indica quanto un record è anomolo. Per selezionare l'1% dei valori, vengono identificati i 19 outliers con il punteggio più elevato.

Voto di maggioranza

Il risultato finale del rilevamento delle anomalie è stato ottenuto tenendo conto dei punteggi ottenuti da ciascun metodo esaminato. Sulla base dei risultati forniti da ciascun algoritmo, è stata effettuata una votazione a maggioranza per determinare se un dato è un outlier o inlier. L'insieme di queste tecniche di anomaly detection, ha assegnato l'etichetta *"outlier"* a un record solo se la maggioranza dei suoi componenti (ovvero gli algoritmi testati) era d'accordo con tale assegnazione. Questo approccio ha portato ad identificare sempre 19 anomalie. In questa maggioranza ovviamente non sono compresi criteri quali il *Likelihood Approach*, ma comunque successivamente l'esito riportato da quest'ultimo è stato confrontato con i voti di maggioranza e sono state trovate varie corrispondenze.

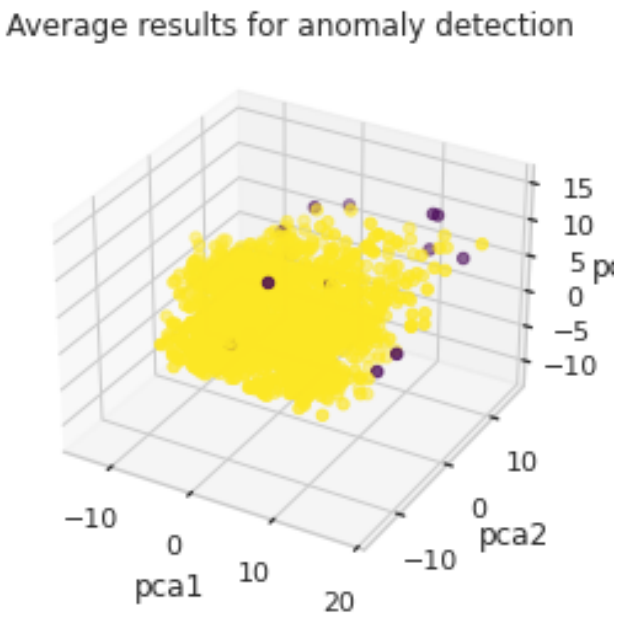

Figura 10: Scatter Plot Top 1% outliers - Voto di Maggioranza. Outliers in viola e Inliers in giallo.

Eliminazione outliers

I valori anomali sono stati gestiti attraverso l'eliminazioni di essi solo da dataset di train, in quanto risultano essere in numero esiguo. Inoltre, non sono stati ritenuti particolarmente interessanti. Infatti, la maggior parte di questi outliers si presentano quando l'emozione non è *happy*, ma corrisponde a *fearful* o *calm*. Infine si è notato che la maggior parte di queste anomalie si ha in situazioni in cui la variabile *lag1_sum_w1* si trova in un intervallo di valori per cui sono state individuate anomalie attraverso il box plot precedente.

Anomaly Detection - Dataset ridotto con PCA

Gli stessi esperimenti sono stati effettuati sul dataset di train ridotto ottenuto applicando la tecnica PCA di dimensionality reduction. Questo studio è stato condotto per facilitare la comprensione degli outliers e magari scoprire più particolari delle cause di eventuali anomalie, avendo in questo caso meno features. Ma anche con il dataset ridotto a 39 attributi, l'1% degli outliers ottenuti con le diverse tecniche utilizzate risultano essere pochi ovvero 21. Ma comunque i 2 dati anomali aggiunti non portano a conclusioni interessanti, e sembra non esserci nessuna particolare correlazione con le 39 features di questo dataset. L'unica differenza rilevante consiste nell'Isolation Forest della libreria *eif* che con questo dataset riesce a trovare 19 outliers con la foresta normale e 21 con quella estesa.

6. Advanced Classificators Methods

Tutti gli studi condotti con i metodi di classificazione avanzata hanno come protagonista i dataset originali, solamente ripuliti tramite la Data Preparation. Per il dataset di train è stato cosiderato il dataset senza outliers. I dati sono stati sempre normalizzati con la *StandardScaler*. I criteri di classificazione sono stati allenati sul train set per poi eseguire la previsione con il test set. Per questi esperimenti si è deciso di svolgere un nuovo task binario, in cui le 8 emozioni sono state suddivise in positive e negative, escludendo *surprised* e *neutral* in quanto non appartenenti a nessuna delle due categorie stabilite. Per tale motivo è stata aggiunta la feature *positive* in cui le emozioni categorizzate come positive assumono valore 1 mentre quelle identificate come negative valore 0. Questo nuovo task è stato scelto in prosieguo al task *happy - not happy*, poiché con gli esperimenti eseguiti attraverso quest'ultimo è stato rilevato che l'emozione *happy* non viene predetta facilmente probabilmente per via dei pochi records caratterizzati da essa. Essendo questa un'emozione positiva si è voluto generalizzare questa suddivisione col fine di raggiungere esiti migliori, e concentrarsi sulla comprensione della positività e/o della negatività trasmessa dagli audio vocali attraverso lo studio delle emozioni.

6.1 Logistic Regression

Le features prese in considerazione inizialmente per la risoluzione della Logistic Regression sono le caratteristiche originali dell'audio ossia: *sum, std, kur*. Per il caso della regressione *Univariata*, la variabile indipendente scelta in questo insieme è la *curtosi*, che risulta essere la feature che conduce a punteggi migliori per le metriche considerate, come visualizzabile in tabella 11. Con tale attributo inoltre, l'*accuracy* aumenta fino a 0.56 e l'AUC incrementa a 0.59.

Class	Precision	Recall	F1 score
negativa	0.74	0.48	0.58
positiva	0.44	0.70	0.54

Figura 11: Logistic Regression - KUR

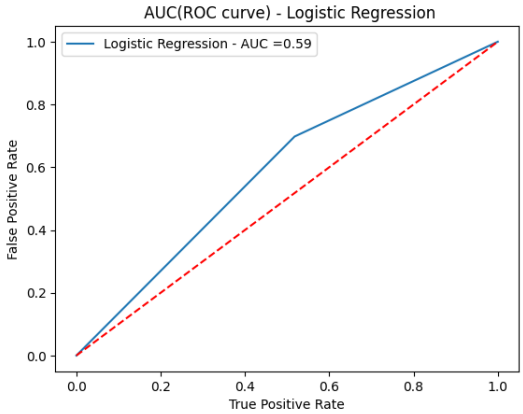

Figura 12: ROC Curve - KUR

Per effettuare un confronto, è stata eseguita anche la logistic regression con più variabili indipendenti, utilizzando sempre le caratteristiche dell'audio originale. I risultati riportati in tabella 13, risultano essere migliori del task univariato. Dalla figura 14 è possibile notare anche un'aumento dell'AUC della ROC Curve rispetto alla precedente.

Class	Precision	Recall	F1 score
negativa	0.76	0.54	0.63
positiva	0.46	0.70	0.56

Figura 13: Logistic regression - Features Audio Originali

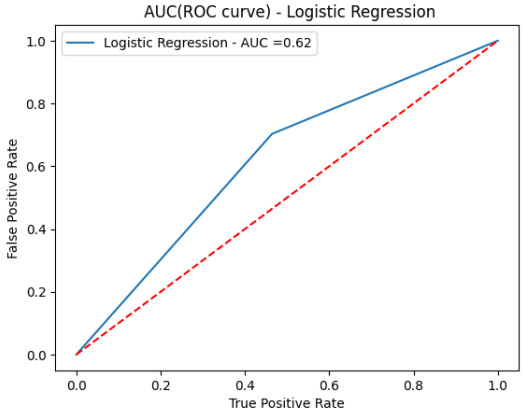

Figura 14: ROC Curve - Features Audio Originali

Un ulteriore indagine è stata effettuata utilizzando tutte gli attributi numerici del dataset come variabli indipendenti per predire la solita variabile dipendente *positive*. Come si evince dalle figure 15 e 16 si hanno performance più soddisfacenti rispetto alle due prove riportate precedentemente, con *accuracy* pari a 0.67 e ROC Curve con un'AUC del 0.67. Infine, tramite la *Permutation Feature Importance* è stato possibile comprendere il contributo relativo che ogni funzionalità apporta a una stima. Dalla permutation importance sull'insieme delle original features dell'audio risulta che la variabile in testa è la KUR. Mentre dalla permutation importance su tutti gli attributi la caratteristica che risalta è *zc_q75_w4*.

Class	Precision	Recall	F1 score
negativa	0.78	0.67	0.72
positiva	0.53	0.67	0.59

Figura 15: Logistic regression - All Features

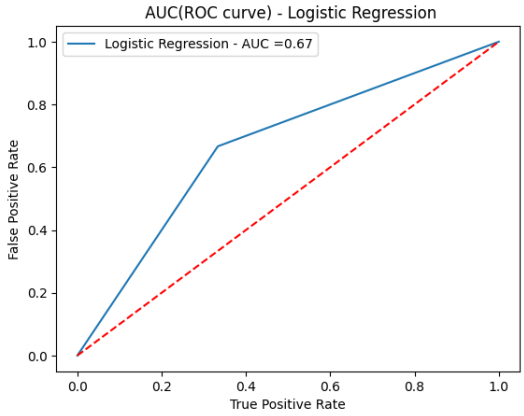

Figura 16: ROC Curve - All Features

Concludendo è possibile confermare come la regressione logistica che considera come variabili indipendenti tutte le features numeriche, risulta essere la più efficace per risolvere il problema di regressione analizzato. Questo probabile è collegato al numero di variabili esplicative scelte, e probabilmente con un numero basso il modello non riesce a spiegare bene la variabile dipendente *positive*.

6.2 Support Vector Machine

In questa Sezione abbiamo sperimentato l'SVM lineare e non lineare per il compito di riconoscimento delle emozioni *positive* e *negative*. Gli iperparametri ottimali sono stati ottenuti sempre tramite l'impiego della *GridSearchCV*.

Linear SVM

La *GridSearchCV* per questo task di SVM lineare ha restituito come parametri ottimali: *'C': 1.0, 'tol': 0.0001*. Con questi valori sono stati ottenuti i risultati riportati in tabella 17. L'accuracy ottenuta è pari al 0.70, ma quest'ultima non viene considerata la metrica ideale per la valutazione poichè il task *positive - negative* analizzato risulta essere sbilanciato (554 contro 972).

Class	Precision	Recall	F1 score
negativa	0.76	0.78	0.77
positiva	0.60	0.58	0.59

Figura 17: Linear SVM - Results

Di conseguenza sono state valutate metriche quali l'AUC che riporta un valore pari a 0.75 come visto in figura 19. Tramite questa metrica si possono comprendere meglio anche informazioni relative a eventuali errori. In questo caso avendo ottenuto un valore di AUC maggiore di 0.70, è possibile affermare che il test eseguito con il Linear SVM è moderatamente accurato. Infine in figura 18 è possibile visualizzare la Lift Curve che permette di visualizzare il rapporto di miglioramento cumulativo in base alla percentuale di dati

classificati. In questo caso il modello ha un lift maggiore del 2.0 a 0.1 circa dei dati, quindi i risultati del modello sono quasi migliori nel primo 10% dei dati per la classe 1 rispetto ai risultati senza un modello.

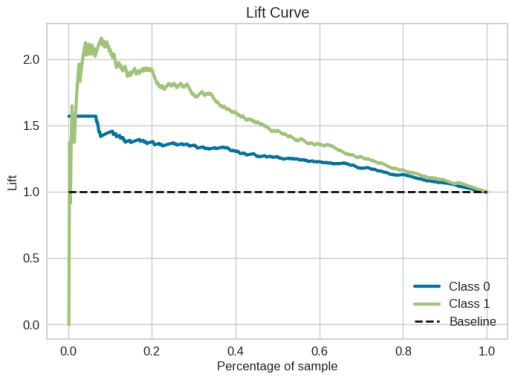

Figura 18: Linear SVM Lift Curve

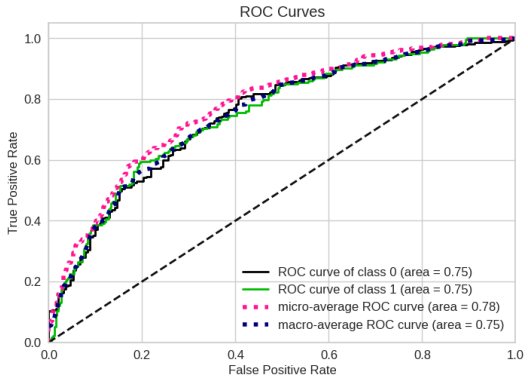

Figura 19: Linear SVM ROC Curve

Non Linear SVM

Anche per questo task è stata utilizzata la *GridSearchCV* per ottenere la miglior configurazione degli iper-parametri, la quale ha restituito come miglior parametri: *'C': 10.0, 'gamma': 0.01, 'kernel': 'rbf'*. Con questi parametri, si ottengono i valori delle metrice riportate in figura 20 e una *accuracy* del 0.70 che rimane inviariata rispetto al Linear SVM. Mentre per l'area AUC della ROC Curve si subisce un calo del valore che diventa pari a 0.73.

Class	Precision	Recall	F1 score
negativa	0.74	0.82	0.78
positiva	0.61	0.51	0.56

Figura 20: Non Linear SVM - Results

Successivamente sono stati testati ulteriori Kernel al fine di ottenere performance più soddisfacenti. Con il kernel *lineare* e *poly* è stata ottenuta la stessa *accuracy*, mentre il *sigmoid* non raggiunge il 0.65. Per quanto riguarda l'*f1-score* tutti i kernel hanno prestazioni leggermente inferiori al *RBF* restituito con l'hyper tuning dei parametri. Concludendo, il Linear SVM e il Non Linear SVM conducono a risultati simili per il task binario *positive - negative*, anche se i valori di alcune metriche utilizzate per le valutazioni risultano essere leggermente superiori nel caso lineare che è preferibile anche per via del suo costo minore dal punto di vista computazionale.

6.3. Neural Networks

In questa Sezione sono descritti i risultati ottenuti dalle analisi effettuate con le Neural Networks. Per ovviare il problema dell'overfitting c'è bisogno di controllare il numero di hidden layers ed effettuare una

corretta regolarizzazione. Per tale motivo viene eseguita la scelta degli iperparametri e per selezionare i migliori viene adoperata nuovamente la *GridSearchCV*.

Single Perceptron

Per quanto riguarda il Single Perceptron, in figura 21 sono rappresentati i risultati ottenuti con gli iperparametri scelti che conducono ad una ROC Curve con AUC pari a 0.75. Come si può notare dalla tabella, l'algoritmo è stato eseguito anche dopo aver bilanciato il dataset con l'Oversampling e l'Undersampling. Queste prove sono state condotte per comprendere la variazione delle metriche di valutazione e delle performance del modello in base al bilanciamento della variabile target. Gli esiti hanno dimostrato che l'algoritmo ritorna valori migliori quando il dataset è più bilanciato.

	parametri	Accuracy	Precision	Recall	F1 score
single perceptron	alpha:0.001 penality:l1 tol:0.1	0.61	0.60	0.61	0.60
single perceptron con oversampling	alpha:0.001 penality: l1 e tol: 0.1	0.66	0.65	0.66	0.65
single perceptron con undersampling	alpha: 0.003 penality: l1 tol:0.01	0.68	0.67	0.69	0.67

Figura 21: Single Perceptron Performance

Multilayer Perceptron

Per il Multilayer Percpetron sono state eseguite le stesse procedure del modello precedente. Le metriche e i parametri ottenuti sono visibili nella tabella 22. Sono state applicate le tecniche di Undersampling e Oversampling anche in questo caso, e le performance del classificatore aumentano nuovamente quando le due classi risultano bilanciate.

	parametri	Accuracy	Precision	Recall	F1 score
multilayer Perceptron	hidden layer sizes:(128, 64, 32), learning rate: adaptive, momentum: 0.9	0.68	0.67	0.68	0.67
multilayer Perceptron con oversampling	hidden layer sizes: (128, 64, 32), learning rate: adaptive, momentum: 0.9	0.70	0.68	0.69	0.68
multilayer Perceptron con undersampling	hidden layer sizes: (128, 64, 32), learning_rate: adaptive, momentum: 0.9	0.68	0.68	0.69	0.68

Figura 22: Multilayer Perceptron Performance

In generale i numeri ottenuti appaiono molto simili tra il Single e il Mulilayer Perceptron. Di quest'ultimo, in particolare, risultano essere superiori i valori ottenuti dopo l'Oversampling.

Deep Neural Networks

Il classificatore è stato implementando delineando una architettura costituita da più livelli formati da un numero differente di neuroni per ogni livello. La funzione di attivazione dei livelli è la *relu*. Quest'ultima è la funzione di attivazione migliore e più avanzata rispetto al *sigmoid* e *TanH* perché tutti gli svantaggi come il *Vanishing Gradient Problem* sono completamente rimossi. Con questo modello si ottengono valori di *accuracy* pari a 0.70 e di *F1-score* pari a 0.75 per la classe *negative* e 0.60 per la classe *positive*. Infine sono state effettuate prove modificando la struttura del modello con parametri di regolarizzazione quali

L2 e DROPOUT. Con questi metodi sono stati ottenuti risultati meno significativi rispetto a quelli già riportati, poiché l'*accuracy* subisce un decremento e la *loss* un incremento. Questo porta a concludere che con i parametri di regolarizzazione c'è un calo delle performance e quindi la rete neurale non modella bene i dati di addestramento.

6.4 Ensamble Methods

In questa sezione sono stati approfonditi i seguenti modelli: *AdaBoost, Random Forest e Bagging* per la classificazione binaria delle emozioni suddivise in positive e negative. La configurazione dei migliori iperparametri per ogni modello viene restituita ulteriormente dalla *GridSearchCV*. I risultati ottenuti e gli iperparametri ritenuti ottimi per i modelli testati sono riportati nella figura 23.

	parametri	Accuracy	Precision	Recall	F1 score
Adaboost	learning rate: 0.1, n estimators: 5	0.65	0.60	0.57	0.56
Random Forest	criterion: gini,max_depth:10,min samples leaf: 1,min samples split: 5, n estimators: 100	0.70	0.67	0.64	0.65
Bagging	base estimator:None, n estimators=100, random state:42	0.70	0.67	0.65	0.66

Figura 23: Ensamble Methods Performance

Per l'***Adaboost*** otteniamo un valore AUC pari a 0.58, quindi il modello non è ritenuto abbastanza informativo per questo task poiché evidentemente non discrimina molto bene le emozioni positive dalle negative. Effettuando la tecnica della *Feature Importance* è stato possibile comprendere che ci sono solo due variabili importanti nel predire la variabile target, ovvero *mfcc_q05_w3 e stft_sum_w2*. Il ***Bagging*** è stato testato con differenti tecniche di classificazione ovvero: *GaussianNB, Decision Tree, KNN, Logistic Regression, Nessun Stimatore, SVC e Random Forest.* Nella figura 30 sono riportate le ROC Curve ottenute utilizzando diversi stimatori di base da adattare a sottoinsiemi casuali del set di dati. Le performance migliori sono ottenute quando il Bagging viene eseguito impostando come estimatore base *None*. In questo caso di default viene considerato un Decision Tree e l'*accuracy* risulta essere 0.70. Il bagging peggiora notevolmente quando viene considerato l'SVC o la Logistic Regression, che ritornano un'*accuracy* pari a 0.62 e un valore di AUC pari a 0.60 e 0.61.

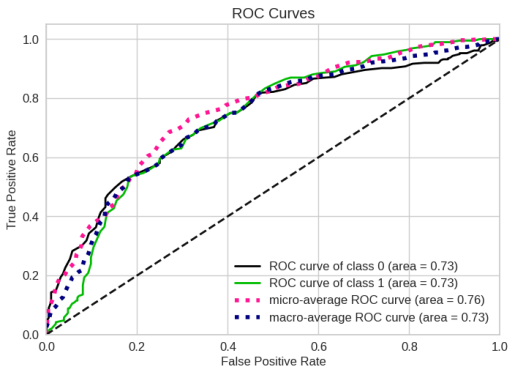

Figura 24: Stimatore di Default - Decision Tree

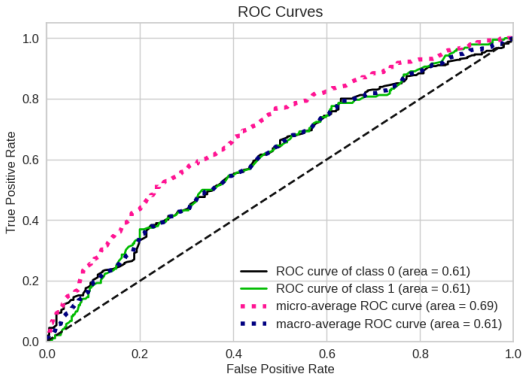

Figura 25: Random Forest

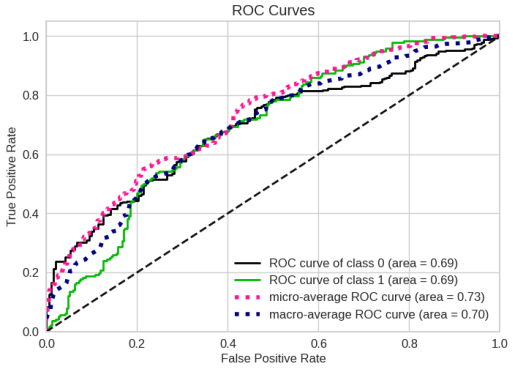

Figura 26: GaussianNB

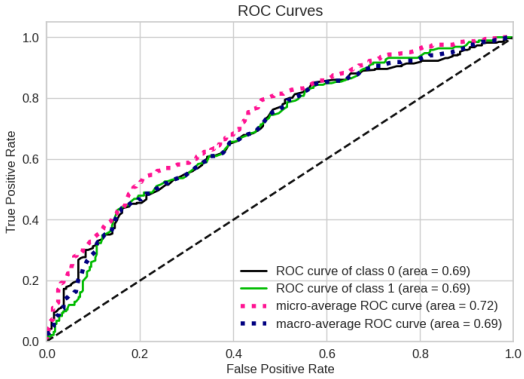

Figura 27: KNN

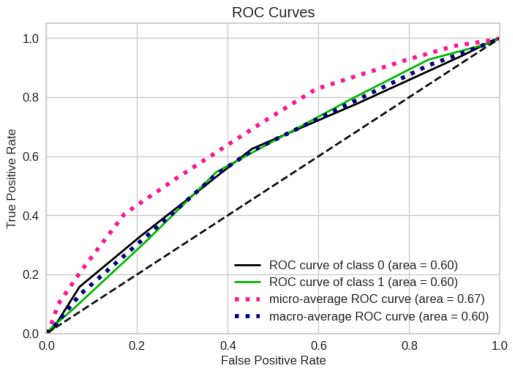

Figura 28: SVC

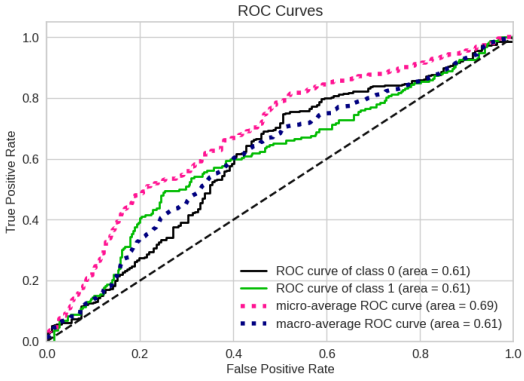

Figura 29: Logistic Regression

Figura 30: Bagging Different Estimators - ROC Curve

La maggior parte degli studi effettuati sul bagging portano a concludere che indipendentemente dallo stimatore considerato, spesso risulta un valore molto alto della metrica *specificity* che indica la percentuale di elementi negativi del testing set che sono stati classificati come negativi; e uno molto basso della *recall*. Questo ci permette di constatare che le emozioni *positive* non vengono predette correttamente e questo probabilmente è causato dallo sbilanciamento del dataset. Per la ***Random Forest*** sono stati ottenuti valori molto simili al bagging con lo stimatore base. L'unica differenza è che l'AUC riportata da questa tecnica

risulta essere leggermente maggiore del bagging, ovvero pari a 0.74. Inoltre la percentuale di elementi positivi del testing set che sono stati classificati come positivi risulta essere 0.46 e quindi superiore rispetto alle precedenti. Infine, anche per la Random Forest è stata eseguita la *Feature Importance*, i cui risultati evidenziano *mfcc_q05_w3* come attributo più rilevante.

6.5 Gradient Boosting

In questa Sezione viene adoperato il metodo di classificazione *Gradiente Boosting Machines* e viene messo a confronto con altre tecniche appartenenti a questa famiglia, ovvero *XGBoost, LighGBM, CatBoost, HistGradientBoosting*. Il frutto di questi esperimenti è esposto nella tabella 31. Da quest'ultima è visibile come tutte queste tecniche conducono a valori affini delle metriche di valutazione.

	Accuracy	Precision	Recall	F1 score
Gradient Boosting	0.70	0.67	0.67	0.67
XGBoost	0.70	0.68	0.66	0.67
CatBoost	0.70	0.67	0.65	0.66
Light GBM	0.71	0.69	0.65	0.66
HISTGBC	0.69	0.67	0.66	0.66

Figura 31: Gradient Boosting Methods Performance

Conclusions

Riassumento questa Sezione dei classificatori avanzati, è possibile affermare che, nonostante l'*accuracy* non rappresenti la metrica ideale per il task binario *positive-negative* poiché non equamente bilanciato; è stata comunque presa in considerazione perché determina la percentuale di previsioni corrette effettuate dal modello. In base a questo indice il modello ***LightGBM***, appartenente alla famiglia dei *Gradient Boosting Classificator*, è quello con *accuracy* superiore tra tutti i classificatori avanzati esaminati. Questo era prevedibile per via della struttura del *LightGBM* che consente una migliore accuratezza rispetto a qualsiasi altro algoritmo, poiché produce alberi molto più complessi seguendo l'approccio diviso in base alla foglia piuttosto che in base al livello. Valutando una metrica più tangibile, come l'AUC che è adatta per set di dati sbilanciati, risulta che i modelli più convenienti sono la ***Random Forest*** con valore 0.74, il ***Single Perceptron*** con valore 0.75 e il ***Linear SVM*** con valore 0.75.

7. Advanced Regression

Il dataset utilizzato per questo task è quello originale solamente ripulito tramite Data preparation. Per il dataset di train è stato cosiderato il dataset senza outliers. I dati sono stati sempre normalizzati con la *StandardScaler*, mentre per le variabili categoriche è stata definita una codifica *LabelEncoding*. La regressione avanzata è stata eseguita tramite *Random Forest e Gradient Boosting* e per entrambi la variabile target

numerica considerata è ***KUR***, la cui previsione si ottiene dalle restanti variabili indipendenti presenti nel set di dati. I parametri ottimali sono stati ottenuti in questo caso tramite la *RandomizedSearchCV*, e nella figura 32 sono trascritte le metriche e i rispettivi punteggi ottenuti con le due tecniche esaminate. Da questi punteggi è possibile visualizzare che il Gradient Boosting risulta preferibile in quanto presenta uno scarto quadratico medio più basso e un coefficiente di determinazione più alto.

	parametri	scarto quadratico medio	coefficiente di determinazione
Random Forest	min samples leaf: 5 min samples split:5	10.06	0.74
Gradient Boosting	n estimators:100, loss:squared error, learning rate: 0.1, min samples split: 2, min samples leaf: 1	9.40	0.76

Figura 32: Advanced Regression Performance

8. Multiclass Emotions Experiments

In questa Sezione sono stati sperimentati alcuni dei classificatori avanzati utilizzando la variabile *emotion* come target, ed è stata definita una codifica *Label Encoding* per tale variabile ottenendo 8 classi corrispondenti ad ogni emozione. Il dataset utilizzato per questo task è lo stesso delle precedenti analisi e i dati sono stati sempre normalizzati con la *StandardScaler*. Nella tabella 33 sono esposte le performance di tutti i classificatori testati e si può affermare che il più proficuo per questo task è la *Deep Neural Network*. Per definire i parametri della Rete Neurale è stato inizialmente definito un modello baseline utile a comparare le performance dei vari modelli testati. La rete neurale del modello baseline è composta da due hidden layer, il primo composto da 128 neuroni e il secondo da 64, entrambi con funzione di attivazione *ReLu*. L'output layer conta otto neuroni, uno per ogni classe da predire, con funzione di attivazione *softmax*. Sulla stessa infrastruttura di rete sono stati testati tre metodi di regolarizzazione, *L1, L2 e Dropout*. I metodi *L1 e L2* sono stati applicati su entrambi gli hidden layer con il valore *lambda = 0.01*, mentre il *Dropout* è stato effettuato aggiungendo un layer di dropout del 10% dei neuroni dopo entrambi gli hidden layer. I risultati migliori sono stati raggiunti grazie alla regolarizzazione *L2*, che è stata quindi utilizzata per creare un altro modello i cui parametri sono stati ottenuti tramite l'utilizzo della *RandomizedSearchCV* con *CV=5*.

	DEEP NEURAL NETWORK	RANDOM FOREST	BAGGING	ADABOOST	GRADIENT BOOSTING	XGBOOST	LIGHTGBM	CATBOOST
Accuracy	0.49	0.42	0.42	0.48	0.44	0.45	0.47	0.49
f1 score	0.47	0.39	0.37	0.47	0.44	0.44	0.46	0.47
Precision	0.48	0.43	0.42	0.48	0.45	0.46	0.48	0.48
Recall	0.50	0.41	0.40	0.48	0.44	0.44	0.47	0.48

Figura 33: Advanced Classificators Performance - Multiclass Emotions

Per quanto riguarda questo task multiclasse è stato notato che tutti i classificatori provati non hanno condotto a risultati soddisfacenti, infatti i valori delle metriche considerate sono sempre minori di 0.5.

9. Time Series Analysis

9.1 Data Understanding and Preparation

In questa fase di Data Understanding e Preparation, sono stati scaricati i dataset di test e di train, e sono stati preparati i dati per eseguire i task successivi. Ogni record è stato preprocessato con la libreria *Librosa*, tramite cui sono state estratte le time series. I dati sono stati normalizzati attraverso la libreria *TimeSeriesScalerMinMax*. Inoltre, essendo presenti parti "vuote" in testa e in coda nelle time series, queste parti sono state ritagliate. Infine, i dataset sono stati compressi per attuire il problema di complessità computazionale riscontrato durante l'esecuzione dei vari task.

9.2 Clustering

In questa Sezione vengono riportati i risultati ottenuti durante la fase di Clustering. Il dataset utilizzato è quello descritto precedentemente e l'algoritmo principalmente sperimentato è il *k-means*. Infine, è anche illustrato l'esperimento effettuato con il *clustering gerarchico*.

K-Means

Questo algoritmo è stato applicato sia sul dataset non approssimato, sia sui dataset approssimati tramite SAX e PAA. La metrica di distanza utilizza è l'Euclidea. Per gli esperimenti con l'approssimazione è stato scelto un numero di simboli e segmenti pari a 100. Il numero di clusters k è 8. Tale numero è stato scelto poiché il dataset contiene file audio parlati e cantanti, che cercano di trasmettere una emozione tra le 8 messe a disposizione per essere interpretate dagli attori. In questo modo si cerca di capire se il K-Means sia in grado di separare i dati in base alle emozioni, assegnandoli ai rispettivi cluster. Per quanto riguarda la prova effettuata con il K-Means senza approssimazione, è stato possibile notare che alcuni clusters riescono ad individuare un'unica emozione: il numero 0 presenta solo l'emozione *neutral*, il numero 1 l'emozione *angry*, il numero 2 *fearful* e infine il numero 5 *angry*. Nei restanti cluster possiamo notare che sono emerse tutte le emozioni in modo abbastanza omogeneo; solo nel cluster 6 sembra prevalere maggiormente l'emozione *calm*. Nella tabella 34 sono mostrate le percentuali della presenza delle emozioni per ogni cluster.

	calm	sad	fearful	disgust	surprised	happy	neutral	angry
cluster 0	0%	0%	0%	0%	0%	0%	100%	0%
cluster 1	0%	0%	0%	0%	0%	0%	0%	100%
cluster 2	0%	0%	100%	0%	0%	0%	0%	0%
cluster 3	16,15%	13,77%	13,54%	4,75%	10,21%	14,96%	8,79%	17,81%
cluster 4	11,00%	13,36%	15,60%	8,78%	6,95%	17,03%	8,12%	19,13%
cluster 5	0%	0%	0%	0%	0%	0%	0%	100%
cluster 6	33,33%	18,67%	14,00%	10,00%	10,00%	7,33%	4,00%	2,67%
cluster 7	15,92%	18,77%	16,73%	8,57%	6,73%	15,51%	6,94%	10,81%

Figura 34: Composizione Clusters del K-Means Senza Approssimazione

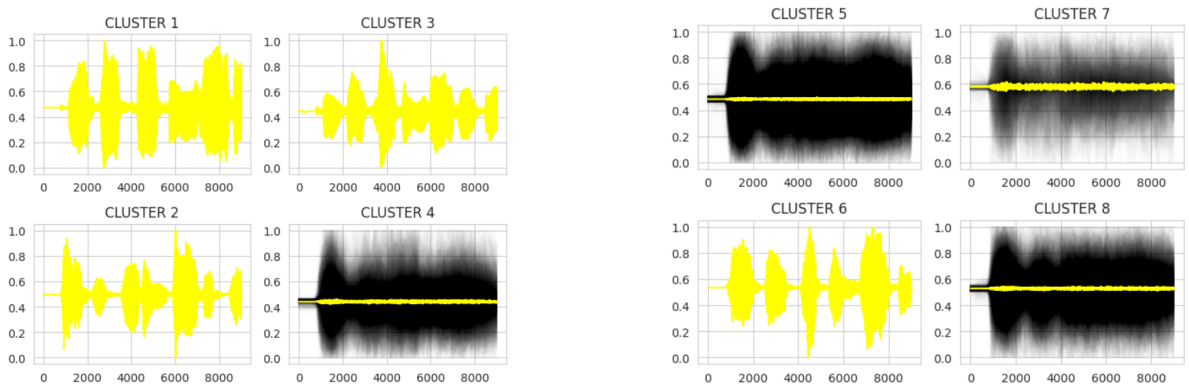

Figura 35: Centroidi del clustering K-means sulle time series senza approssimazione

Avendo ottenuto un il valore di *support* basso relativo ai cluster che individuano un'unica emozione, ed essendocene due che individuano la stessa emozione *angry*; sono stati ritenuti migliori i risultati ottenuti sempre con il k-means ma usando le approssimazioni SAX e PAA. Con il dataset approssimato tramite la SAX è risultato un solo cluster contentente un'unica emozione *sad*, ma con un *support* comunque molto basso. Invece gli altri cluster ottenuti sono caratterizzati da tutte le emozioni in modo abbastanza omogeneo. Nel cluster numero 2 prevale maggiormente la *calma*, con una percentuale pari a 33,11%. Per quanto riguarda gli esperimenti sul dataset approssimato con la PAA, i risultati riportati in tabella 36 mostrano che nessun cluster racchiude un'unica emozione e inoltre quest'ultime sono presenti in misura simile. Solo il cluster 3 riporta una presenza notevolmente maggiore di un'unica emozione, in questo caso *calm*, rispetto ai restanti. Nel cluster 2 invece si può notare la quasi assenza di *disgust* indicata da una percentuale molto inferiore rispetto alle altre, pari a 0,85%.

	calm	sad	fearful	disgust	surprised	happy	neutral	angry
cluster 0	12,10%	14,01%	15,28%	7,64%	7,64%	16,24%	9,87%	17,19%
cluster 1	14,00%	20,40%	18,00%	9,60%	3,60%	14,00%	5,20%	15,20%
cluster 2	19,50%	11,01%	19,50%	0,85%	8,47%	14,40%	7,63%	18,64%
cluster 3	37,11%	17,53%	12,37%	10,31%	10,31%	5,15%	4,12%	3,10%
cluster 4	10,91%	12,32%	17,95%	8,80%	8,10%	19,72%	8,45%	13,73%
cluster 5	15,25%	14,91%	11,20%	6,44%	11,20%	15,60%	8,81%	16,61%
cluster 6	25,00%	20,00%	16,46%	9,14%	10,36%	10,36%	6,10%	2,44%
cluster 7	10,13%	14,05%	13,40%	8,50%	5,88%	17,32%	7,51%	23,20%

Figura 36: Composizione Clusters del K-Means con PAA

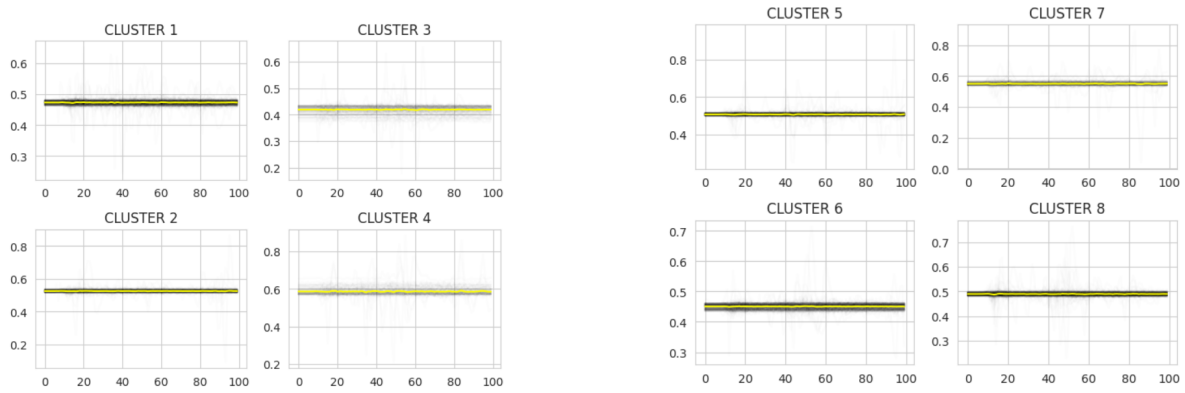

Figura 37: Centroidi del clustering K-means sulle time series con PAA

Concludendo, è possibile affermare che i risultati migliori basati sulla differenziazione delle emozioni e sui valori di support correlati sono ricavati effettuando l'approssimazione con la PAA. Quindi, in generale, il k-means non riesce a separare perfettamente gli audio per ogni emozione. Questo probabilmente è dovuto anche all'utilizzo della metrica Euclidea che, nonostante sia la metrica più utilizzata e più intuitiva per le serie temporali, è comunque una distanza sensibile al rumore e incapace di gestire i *local time shifting*. Quest'ultima significa che i segmenti simili sono fuori fase e sono quindi sensibili a piccole traslazioni lungo l'asse del tempo.

Clustering Gerarchico

Come secondo algoritmo di clustering per le time series, è stato preso in considerazione il clustering gerarchico. La metrica utilizzata è nuovamente l'Euclidea, e i metodi utilizzati sono: *single, complete e ward*. Tra questi ultimi, quello che ha condotto al raggiungimento di cluster più strutturati è il *ward*, e in figura 38 è raffigurato il dendogramma ottenuto.

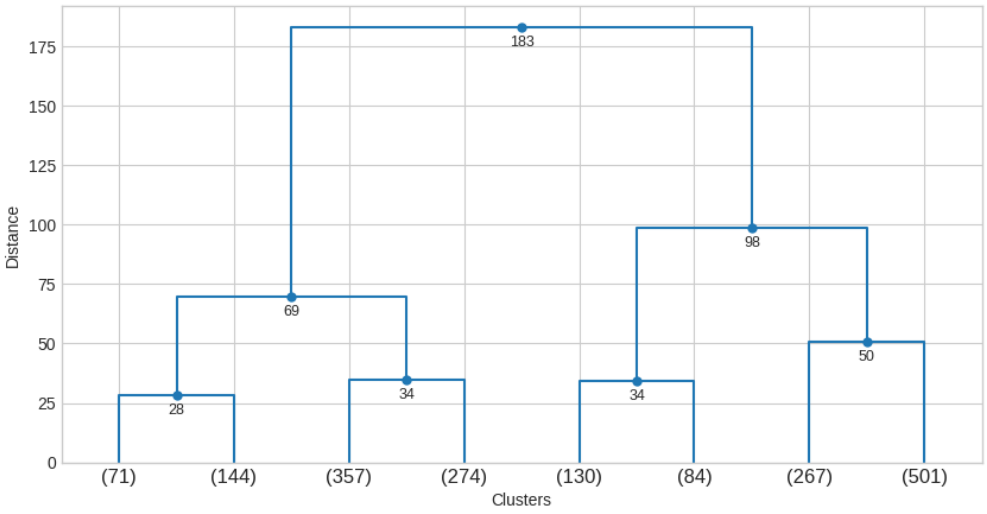

Figura 38: Clustering Gerarchico TS - Metodo Ward

Questo dendrogramma permette di visualizzare come i cluster cambiano in funzione della distanza massima che viene applicata ai dati e che rappresenterà la soglia. Dal grafico è quindi possibile dedurre che il numero di cluster ottimale è pari a 2.

9.3 Motifs and Anomaly Discovery

In questa Sezione vengono estratti i motifs e le anomalie presenti nelle time series. E' stato deciso di effettuare questo lavoro principalmente sui dati caratterizzati dall'emozione *calm e angry*, questo poiché risultano essere le emozioni che maggiormente prevalgono nei cluster analizzati nella sezione precedente. I risultati dei Motifs analizzati rappresentano quelli caratterizzati da una presenza maggiore di modelli precedentemente sconosciuti e che compaiono frequentemente nelle serie di dati temporali. Il Matrix Profile è stato testato variando la dimensione della finestra temporale, sperimentando sia valori di finestra bassi e sia valori alti. Gli esiti ritenuti migliori sono stati ottenuti con una finestra pari a 10. Di seguito, nelle figure 39 e 40, vengono riportati solo una parte dei risultati inerenti all'emozione *calm*, quelli considerati appunto più interessanti.

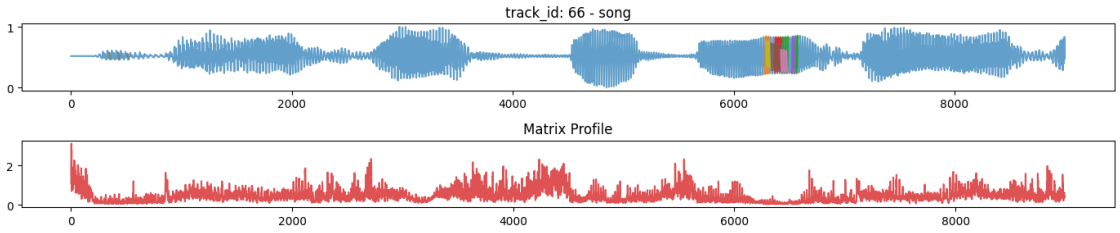

Figura 39: Motifs e Matrix Profile dell'emozione CALM: Song

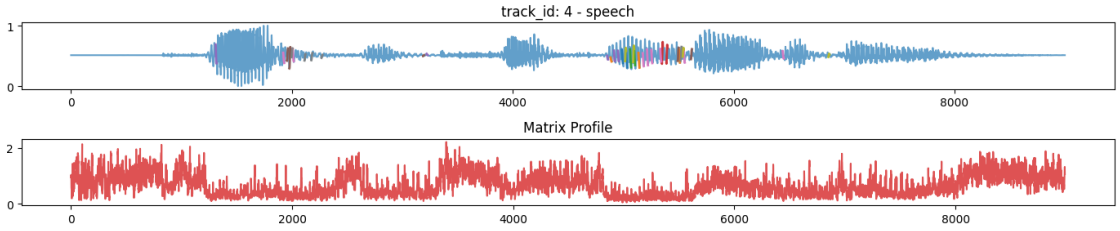

Figura 40: Motifs e Matrix Profile dell'emozione CALM: Speech

L'audio con vocal channel *song*, mostra i motifs all'indice: [6537, 6559], [6286, 6308], [6439, 6461, 6483, 6571], [6375, 6397, 6419], [6525, 6547], [6341, 6363, 6385, 6407], [6430, 6452, 6474], [344, 426, 510], [6295, 6317]. Per questi le distanze sul profilo della matrice sono: [0.024747352895742158, 0.02587259699640924, 0.027087079761547332, 0.03195784986093826, 0.03308605162833605, 0.03496916438387599, 0.037836221787341175, 0.03793434718546223, 0.04087619608318804]. L'audio con vocal channel *speech*, invece, presenta i motifs all'indice: [4988, 5026], [4885, 5138, 5504], [5062, 5098], [5350, 5391], [1309, 3220, 4854, 4916, 4952], [1955, 1977, 3188, 5493, 5613], [1921, 2007, 5177, 5215, 5254, 5335, 5548, 6444], [1997, 2114, 2176, 2268, 5519], [5017, 5090, 5539, 6858]. Per questi ultimi le distanze sul profilo della matrice sono: [0.05510046346935064, 0.09196980243165864, 0.0921459545456981, 0.09261331919927743, 0.09858043017394863, 0.10245275291878594,

0.1036772594918119, 0.10529735390781866, 0.10913563311610573]. La Matrix Profile è stata adoperata anche per rilevare le discrepanze nelle serie temporali. Al fine di rilevare le anomalie, la zona di esclusione è stata impostata a 5, ovvero la metà della finestra temporale utilizzata. Per l'emozione *calm*, non sono state rilevate discordanze evidenti, ma è stato notato che la maggior parte, sia per *speech* sia per *song*, presentano anomalie nelle zone iniziali della serie temporale. In figura 41 sono riportati alcuni esempi.

Figura 41: Anomalie relative all'emozione CALM

Gli stessi esperimenti sono stati condotti focalizzando l'attenzione sui dati caratterizzati dall'emozione *angry*. I risultati sono raffigurati nelle immagini 42 e 43.

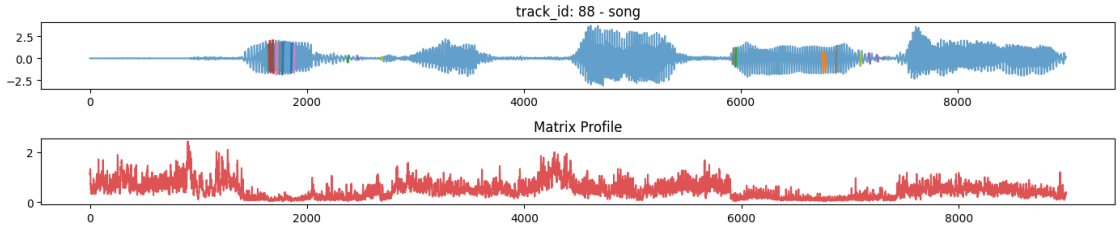

Figura 42: Motifs e Matrix Profile dell'emozione ANGRY: Song

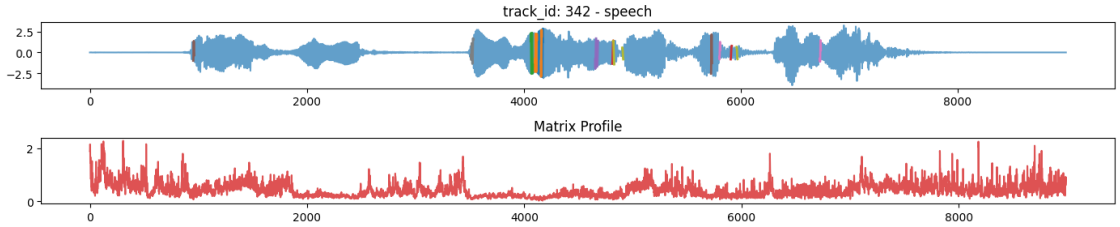

Figura 43: Motifs e Matrix Profile dell'emozione ANGRY: Speech

L'audio con vocal channel *song*, mostra i motifs all'indice:[1773, 1853], [6752, 6774], [2376, 5949], [1659, 1686], [2465, 7182, 7255], [1646, 1673], [1720, 1880], [1746, 5926, 6874, 7310], [2683, 7101]. Le relative distanze sulla matrix profile sono: [0.01745052427612542, 0.021166409966030083, 0.02606962857961187, 0.030362458381052237, 0.041648208274877654, 0.04447615115974383, 0.04482371622784094, 0.04524328299899506, 0.04746313511740382]. Mentre per l'audio *speech* i motifs corrispondono all'indice: [4135, 4169], [4101, 4112, 4157], [4063, 4074], [4816, 5907], [4657, 4674], [953, 5726], [5802, 6729], [3512, 3524], [4824, 4908, 5961]. Le relative distante invece sono: [0.016924983642761852, 0.04370111540800433, 0.05263885820429882, 0.05532413579046073, 0.057336437417495285, 0.059429186960014466, 0.06470725608282508, 0.06662978231311871, 0.07108512522432511]. L'analisi delle anomalie con l'emozione *angry* hanno condotto ai risultati mostrati in figura 44 . Anche in questi casi le discordanze sono presenti maggiormente in corrispondenza della zona inziale delle time series.

Figura 44: Anomalie relative all'emozione ANGRY

9.4 Classification

In questa Sezione vengono illustrati gli algoritmi di classificazione utilizzati per prevedere i valori delle variabili categoriche, ponendo particolare attenzione sugli attributi *emotion* e *vocal_channel*. Per questo task sono state classificate le time series originali, ma successivamente, per mitigare il problema relativo alla complessità computazionale, è stata effettuata la classificazione anche sulle time series approssimate tramite la tecnica *Symbolic Aggregate Approximation (SAX)*. L'allenamento dei modelli è stato eseguito sul dataset di Train, mentre le previsioni sono state effettuate sul dataset di Test.

Time Series senza approssimazione - EMOTION

Il dataset utilizzato in questa Sezione è lo stesso ricavato durante la fase di Data Preparation. Come primo passo è stata conseguita una trasformazione tramite il metodo *Label Encoding* per la variabile *emotion* considerata. I classificatori utilizzati in questa fase sono: *Rocket, MiniRocket e KNN* con la metrica Euclidea. Entrambi i classificatori *Rocket* e *MiniRocket* utilizzano il RidgeClassifierCV. Tra questi ultimi, il classificatore che ha permesso di ottenere performance migliori in termini di *accuracy* e *f1_score* risulta essere il *MiniRocket*. Gli esiti di quest'ultimo sono osservabili in figura 45.

	Precision	Recall	f1-score	Accuracy
Angry	0.56	0.83	0.67	
Calm	0.65	0.61	0.63	
Disgust	0.54	0.60	0.57	
Fearful	0.43	0.27	0.33	
Happy	0.52	0.50	0.51	
Neutral	0.70	0.54	0.61	
Sad	0.52	0.59	0.56	
Surprised	0.49	0.38	0.42	
				0.55

Figura 45: Risultati Mini Rocket

Per quanto riguarda il classificatore *Rocket*, invece, è stato ottenuto un valore di accuracy pari a 0.33, quindi inferiore al valore ricavato con il precedente metodo. Anche l'f1_score risulta essere minore, assumento un valore pari a 0.33. Un'ulteriore prova è stata condotta con il KNN. Quest'ultimo ha riportato risultati

poco soddisfacenti rispetto agli altri due metodi, infatti le metriche ottenute risultano essere molto bassa, ovvero un'accuracy pari a 0.17 e l'f1_score è pari a 0.11. Le performance risultanti da questi tre modelli, sono probabilmente una conseguenza dovuta anche alla scelta della metrica Euclidea.

Time Series con SAX - EMOTION

In questa Sezione sono stati effettutati esperimenti impiegando l'approssimazione delle time series tramite la tecnica *Symbolic Aggregate Approximation (SAX)*. Il classificatore considerato è il KNN e sono stati eseguiti diversi test con l'utilizzo di differenti metriche, ovvero:

- *DTW*
- *DTW sakoechiba*
- *DTW itakura*

Questa scelta progettuale è stata effettuata per poter comprendere se l'utilizzo di una differente metrica potesse condurre ad esiti ottimi rispetto a quelli raggiungi con la distanza Euclidea. Tra tutti questi tentantivi, gli esiti considerati migliori sono stati ricavati adoperando il KNN con l'ausilio della metrica *DTW itakura*, come è possibile notare in figura 46.

	Precision	Recall	f1-score	Accuracy
Angry	0.21	0.31	0.25	
Calm	0.50	0.22	0.30	
Disgust	0.12	0.12	0.12	
Fearful	0.28	0.24	0.26	
Happy	0.22	0.32	0.26	
Neutral	0.28	0.27	0.28	
Sad	0.28	0.24	0.26	
Surprised	0.25	0.21	0.23	
				0.25

Figura 46: Risultati KNN con metrica DTW itakura

Tuttavia è stato esaminato che anche tramite l'impiego delle restanti distanze, si giunge a performance simili. Infatti, con la metrica *DTW* si ottiene un'accuracy pari a 0.23 e f1_score pari a 0.21, quindi leggermente inferiore ai precedenti risultati. Mentre con la *DTW sakoechiba* il valore di accuracy è 0.24 e di f1_score è 0.22. Dunque, si può affermare che tutti e tre gli esperimenti riportano risultati migliori rispetto agli esiti ottenuti con l'utilizzo del KNN e della metrica Euclidea.

Time Series senza approssimazione - VOCAL CHANNEL

Un secondo task di classificazione è stato effettuato sulla variabile target *vocal_channel*. Anche in questo caso sono stati utilizzati gli stessi classificatori analizzati per la feature *emotion*. Lo scopo di tale tentativo è il voler scrutare se è possibile ricavare performance più soddisfacenti, essendo questa una variabile categorica binaria. Il dataset utilizzato è sempre lo stesso ottenuto dalla fase di Data Preparation. Come primo passo è stata effettuata sempre una trasfromazione tramite il metodo *Label Encoding* per l'attributo categorico *vocal_channel*. Analizzando i risultati ottenuti, è possibile confermare che anche in questo caso il classificatore che ha restituito performance migliori in termini di *accuracy* e *f1_score* risulta essere il *MiniRocket*. Infatti è possibile notare in figura 47, che sono presenti valori di f1_score elevati per entrambe le classi e pari a 0.98 e 0.99. Invece l'accuracy risulta essere 0.99. Dal confronto con il *MiniRocket* eseguito per la feature *emotion*, possiamo attestare che questo classificatore riporta risultati nettamente superiori con la variabile target *vocal_channel*.

	Precision	Recall	f1-score	Accuracy
Song	0.98	0.98	0.98	
Speech	0.99	0.99	0.99	
				0.99

Figura 47: Risultati MiniRocket

Tramite il *Rocket*, invece, sono stati ottenuti comunque valori buoni di f1_score pari a 0.93 e accuracy pari a 0.93. Successivamente è stato testato il KNN con la metrica Euclidea, e questo ha riportato esiti poco esaurienti rispetto ai precedenti, infatti l'f1_score risulta essere uguale a 0.38 e l'accuracy pari a 0.58. Infine, per questa feature binaria, è stato eseguito anche il *Feature-Based Classifier*. Quest'ultimo ha restituito valori di accuracy e f1_score pari a 0.77. Anche questi esperimenti confermano che la classificazione con *vocal_channel* conduce a risvolti più soddisfacenti.

Time Series con SAX - VOCAL CHANNEL

Anche in questo caso è stata effettuata un'approssimazione sulle times series analoga a quella effettuata per il task eseguito sulla variabile *emotion*, utilizzando sempre la tecnica *SAX*. Il classificatore testato è sempre il KNN, variando differenti metriche:

- DTW
- DTW sakoechiba
- DTW itakura

Questo esperimento ha riportato performance migliori, in particolare quando viene adoperata la metrica *DTW itakura*. Tali risultati sono rappresentati nella tabella 48.

	Precision	Recall	f1-score	Accuracy
Song	0.75	0.78	0.76	
Speech	0.83	0.81	0.82	
				0.80

Figura 48: Risultati KNN con metrica DTW itakura

Invece per la *DTW* sono stati ottenuti un'*accuracy* del 0.76 e *f1_score* pari a 0.75. Mentre per la *DTW sakoechiba* l'*accuracy* e la *f1_score* sono entrambe pari a 0.78.

Conclusioni

Riassumendo, è possibile notare che il task di classificazione funziona meglio quando viene eseguito sulla variabile binaria *vocal_channel*, questo probabilmente perchè il numero di classi è inferiore e il numero di esempi per ogni classe di conseguenza è maggiore. In entrambi gli esperimenti condotti, ovvero quello con *emotion* e quello con *vocal_channel*, nel caso delle time series non approssimate il classificatore più efficiente risulta essere sempre il *MiniRocket*. Questo era prevedibile poichè, almeno rispetto al *Rocket*, esso è una riformulazione (quasi) deterministica di quest'ultimo che è più veloce su set di dati più grandi e vanta comunque una precisione approssimativamente equivalente. Quindi anche se i valori ottenuti da questi due algoritmi si discostano di poco, è comunque preferibile il *MiniRocket* poichè migliora l'efficienza computazionale. Per le prove condotte con l'approssimazione SAX e il KNN con differenti varianti della DTW, è possibile attestare che sono stati ricavati esiti migliori rispetto alla stessa procedura con la metrica Euclidea. Questo era prevedibile poiché la Dynamic Time Warping consente confronti "molti a uno" per creare il miglior allineamento possibile, sfruttando le distorsioni temporali tra di loro, mentre l'Euclidea consente il confronto punto "uno a uno". Tra le varianti, quella che ha condotto a performance migliori è la *itakura*. Infine, è possibile constatare che in generale il *MiniRocket* è l'esperimento che conduce a performance maggiormente elevate rispetto anche al KNN con la DTW itakura e l'approssimazione SAX. Infatti bisogna tener conto che SAX riflette solo la caratteristica del valore medio del segmento e perciò potrebbe perdere informazioni importanti in un segmento, vale a dire l'andamento della variazione del valore nel segmento. Tale mancanza può causare in alcuni casi una classificazione errata, poiché la rappresentazione SAX potrebbe non essere in grado di distinguere diverse serie temporali con valori medi simili ma trend differenti.

SHAPELETS

In questo paragrafo vengono riportati gli esperimenti di classificati eseguiti con le Shapelets, sia sulla variabile target *emotion* sia su *vocal_channel*. I classificatori utilizzati in entrambi i casi sono: *Shapelet Model* e *Shaplet-distances-based*.

Shapelets - EMOTION

Inizialmente è stato testato il *Shapelets model*, e tramite la funzione *grabocka_params_to_shapelet_size_dict* sono stati individuati 7 shapletets di lunghezza 800. Successivamente è stata eseguita la classificazione che ha restituito risultati in termini di accuracy pari a 0.14 e f1-score uguale a 0.06. Dunque, quest'ultimo, non risulta essere il modello ideale per questa analisi. Successivamente è stato applicato il *Shapelet-distances-based Classifier* che ha restituito metriche vantaggiose rispetto al precedente, ed è possibile notare ciò attraverso i risultati esposti in figura 49.

	precision	recall	f1-score	accuracy
angry	0.19	0.35	0.25	
calm	0.22	0.28	0.25	
disgust	0.12	0.04	0.07	
fearful	0.16	0.14	0.13	
happy	0.25	0.21	0.18	
neutral	0.25	0.06	0.10	
sad	0.38	0.27	0.32	
surprised	0.00	0.00	0.00	
				0.20

Figura 49: Risultati Shaplet-distances-based Classifier

Tale tabella, pone in evidenza come l'emozione *surprised* è associata a punteggi pari a 0 per ogni metrica considerata. Essendo quindi che la f1_score assume punteggi superiori solo quando sia la precision sia la recall sono caratterizzate da valori elevati, in questo caso è pari a zero poichè evidentemente il classificatore non riesce a categorizzare correttamente tale emozione e non sono presenti previsioni positive corrette.

Shapelets - VOCAL CHANNEL

Le stesse procedure eseguite precedentemente per *emotion*, sono state riapplicate alla variabile *vocal_channel*, giungendo in questo caso a risultati più convenienti. Per quanto riguarda il *Shapelets Model*, la classificazione è stata effettuata sempre su shapelets di lunghezza pari a 800. Le metriche analizzate restituiscono valori di accuracy pari a 0.58 e di f1-score uguali a 0.37. Mentre per lo *Shapelet-distances-based Classifier* sono stati ottenuti esiti più vantaggiosi, visualizzabili in tabella 50.

	precision	recall	f1-score	accuracy
song	0.64	0.67	0.66	
speech	0.75	0.73	0.74	
				0.70

Figura 50: Risultati Shaplet-distances-based Classifier

Dunque, si può asserire che anche con le shapelets, le performance migliori sono ottenute con la variabile target *vocal channel*, e probabilmente ciò è in parte dovuto alla tipologia del task di classificazione che in questo caso è binario e quindi presenta un numero inferiore di classi.

10. Explainability

In questa sezione si fornisce una spiegazione sulle ragioni per cui il classificatore *Random Forest* descritto precedentemente, classifica le istanze di prova in un determinato modo. Per fare ciò, è stato costruito e addestrato un classificatore *Random Forest* utilizzando gli stessi parametri e partizioni impiegati nella Sezione 6.4.

LIME

L'algoritmo LIME è utilizzato per spiegare le previsioni del modello di apprendimento automatico. Le spiegazioni dovrebbero aiutare a capire perché il modello si comporta in quel determinato modo. Quindi questo algoritmo utilizza una regressione logistica per spiegare il comportamento della scatola nera del classificatore adoperato. Attraverso le figure 51 e 52 viene raffigurata la spiegazione per due istanze selezionate casualmente, classificate correttamente dal modello, appartenenti alle 2 classi di test analizzate. Nell'immagine 51 è riportata l'interpretazione Lime di un'istanza classificata come *positive*. Il modello è sicuro al 63% che si tratti di un'emozione positiva. In particolare i valori *mfcc_std e sft_min_w3* aumentano le possibilità dell'emozione di essere classificata come positiva.

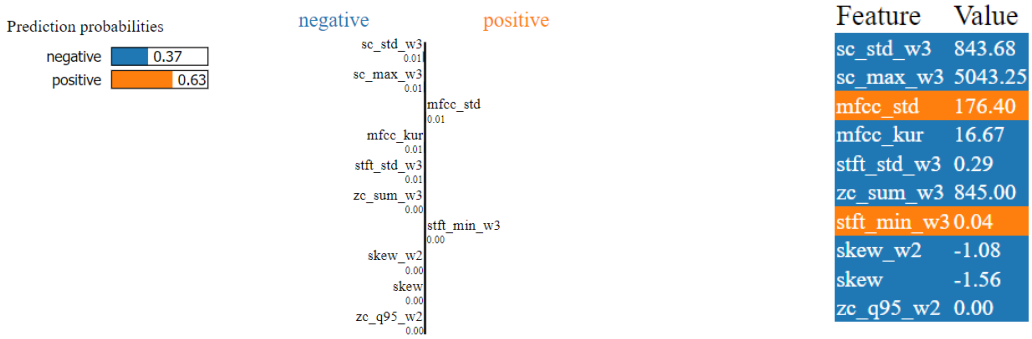

Figura 51: Interpretazione Lime per un'emozione positiva

Nell'illustrazione 52 è invece rappresentata l'interpretazione Lime di un'istanza catalogata come *negativa*. In questo caso è possibile notare come il modello è sicuro al 73% che si tratti di un'emozione negati-

va. Inoltre, diverse variabili quali *sc_std_w3, kur_w3, sc_max_w3, mfcc_mean, mfcc_mean, mfcc_mean_w3, stft_mean_w3, zc_kur_w4 e zc_skew_w4* incrementano le possibilità di questa istanza di essere etichettata come negativa.

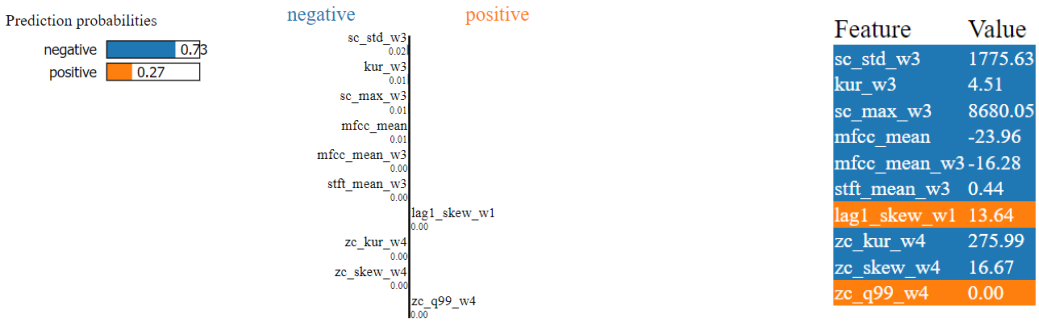

Figura 52: Interpretazione Lime per un'emozione negativa

Concludendo è possibile affermare, attraverso il confronto delle due spiegazioni, che quando la variabile *sc_std_w3 e sc_max_w3* assumono valori più bassi, al file audio viene associata tendenzialmente un'emozione positiva; viceversa se queste feature assumono valori più alti l'audio tende ad essere associato ad emozioni negative.

11. Conclusioni Generali

In questo progetto è stata concessa l'opportunità di lavorare con un ampio set di dati che ha messo a disposizione varie alternative per risolvere differenti compiti di classificazione e clustering. Inizialmente, nella Sezione 1. Introduzione, è stata fornita una piccola descrizione per introdurre il dataset originale, ovvero il RAVDESS ed eventuali suoi possibili impieghi nel mondo reale. Nella Sezione 2. Data Preparation è riportata la fase di preparazione dei dati, in cui è stata eseguita la pulizia e comprensione degli elementi presenti nei dataset di train e test. Successivamente segue la fase 3. Imbalanced Learning, in cui sono stati analizzati gli effetti negativi che i dataset sbilanciati possono avere sui task di classificazione. Di conseguenza sono state migliorate le prestazioni di classificatori semplici come Decision Tree e KNN, applicando e confrontando le differenti tecniche di Oversampling e Undersampling. Questo task di Imbalanced Learning è stato svolto sia sulla variabile target *Emotional Intensity*, sia sulle *emotion* creando un task binario *happy - not happy*. Nella Sezione 4. Dimensionality Reduction sono state applicate le tecniche di riduzione della dimensionalità dei dati. I metodi utilizzati appartengono sia alle procedure di Feature Selection sia alle tecniche di Features Projection. In prosieguo, nella parte 5. Anomaly Detection, sono state rilevate e rimosse l'1% delle anomalie usando i metodi: DBSCAN, HBOS, ELLIPTIC ENVELOPE, KNN, LOF, LODA, ABOD, ISOLATION FOREST (con due differenti librerie) e LIKELIHOOD APPROACH. Nella Sezione 6. Advanced Classificators Methods sono riportati i metodi di classificazione avanzata applicati ad un nuovo task bianario *positive - negative*. I classificatori analizzati sono: Logistic Regression, SVM lineare e non lineare, Neural Network e Deep Neural Network, gli Ensamble Methods (Adaboost, Bagging e Random Forest) e infine il Gradient Boosting. Nella Sezione 7. Advanced Regression è spiegato il compito di regressione avanzata, mentre nella parte 8. Multiclass Emotions Experiments sono riportati ulteriori

esperimenti aggiuntivi di classificazione avanzata condotti sulla variabile *emotion*. Nella Sezione 9. Time Series Analysis vengono riportate le analisi attuate sulle Time Series in cui inizialmente c'è stata una fase di Data Preparation, e poi è stato ultimato il task di clustering attraverso l'impiego del K-Means e del Clustering Gerarchico; in modo da comprendere la capacità e l'efficienza di questi algoritmi nel differenziare le otto emozioni. Sono stati esaminati anche i motifs e le anomalie presenti nelle time series, in modo da intuire la presenza di modelli precedentemente sconosciuti e che compaiono frequentemente nelle serie di dati temporali. Successivamente sono stati riportati gli esperimenti eseguiti per il task di classificazione sulle variabili target *emotion* e *vocal_channel*. I metodi utilizzati sono: KNN con distanza Euclidea e con le varie tipologie della DTW, il Rocket e il MiniRocket. Gli esperimenti con il KNN sono stati realizzati nuovamente dopo aver effettuato l'approssimazione SAX. Al termine di questa parte, è stata effettuata la classificazione anche attraverso le Shapelets. Nella Sezione 10. Explainability è riportata una delle tecniche utilizzate per aiutare a comprendere il perché un modello di intelligenza artificiale genera certe decisioni, grazie alla descrizione del suo funzionamento. L'algoritmo preso in considerazione per tale intuizione è il LIME. In questo caso specifico, la suddetta analisi è stata condotta per fornire una chiara spiegazione del motivo per cui la Random Forest produce una certa etichetta di classe.