Data Analytics

Report del progetto

Andrea Accornero [0001097521]
Alessandro Tocco [0001097880]

Alma Mater Studiorum Università di Bologna
Dipartimento di Informatica - Scienza e Ingegneria
Laurea Magistrale in Informatica

Contents

1 Introduzione

Il presente progetto si concentra sull'esecuzione di un'analisi di regressione utilizzando una varietà di modelli su un dataset composto dalle caratteristiche audio estratte da brani musicali. L'obiettivo primario è predire con precisione l'anno di pubblicazione di ciascuna canzone basandosi esclusivamente su queste caratteristiche audio.

1.1 Dataset

Il dataset fornito è estremamente ampio, comprendendo un totale di 252.175 righe e 91 colonne.

Righe	**Colonne**
252.175	91

Table 1: Carattestiche dataset

Ogni riga rappresenta una singola canzone, mentre le colonne contengono le diverse feature audio estratte per ciascuna traccia.

Year	**S0**	**S1**	**S2**	**S3**	**S4**	...	**S89**
2007	44.76752	114.82099	3.83239	27.99928	1.49153	...	31.32820
2004	52.28942	75.73319	11.35941	-6.20582	-27.64559	...	3.86143
2005	33.81773	-139.07371	134.19332	17.85216	63.47408	...	35.74749
1998	41.60866	3.17811	-3.97174	23.53564	-19.68553	...	3.60432
1987	44.49525	-32.25270	58.08217	3.73684	-32.53274	...	30.11015

Table 2: Dataset

La variabile "Year" indica l'anno di pubblicazione della canzone, mentre le colonne da "S0" a "S89" rappresentano le varie features audio estratte. Queste caratteristiche possono includere informazioni riguardanti l'ampiezza del segnale sonoro, la frequenza, la durata e altre proprietà acustiche rilevanti.

1.1.1 Data Visualization

Per avere una visione completa della distribuzione temporale delle canzoni nel dataset, è stato creato un istogramma utilizzando la libreria Matplotlib. Questo istogramma rappresenta la distribuzione degli anni di pubblicazione dei brani musicali presenti nel dataset. L'asse delle ascisse (x) mostra gli anni di pubblicazione, mentre l'asse delle ordinate (y) indica la frequenza con cui i brani sono stati pubblicati in ciascun anno. Questo grafico fornisce un'importante panoramica della tendenza temporale dei brani musicali nel dataset, permettendoci di identificare eventuali pattern o cambiamenti nel corso degli anni.

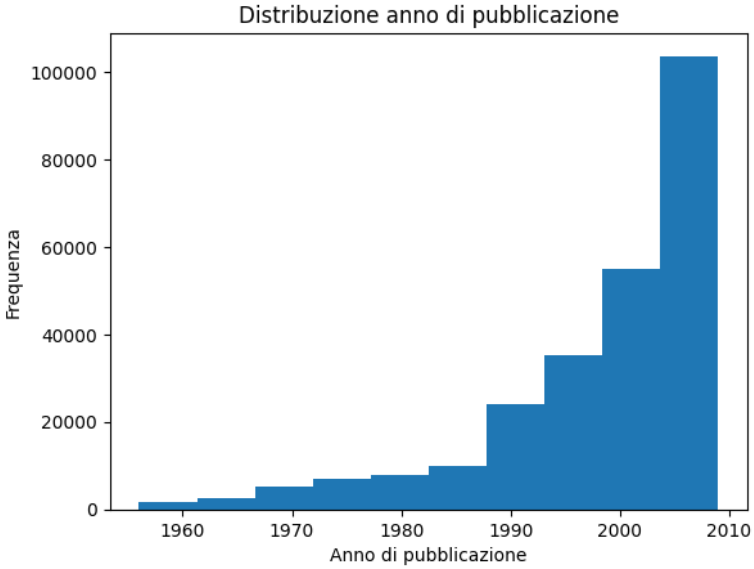


Figure 1: Distribuzione temporale delle canzoni

Inoltre, per comprendere meglio le relazioni tra le diverse caratteristiche presenti nelle canzoni e l'anno di pubblicazione, è stata condotta un'analisi esaustiva utilizzando tecniche di visualizzazione dei dati. Inizialmente, è stata esplorata la correlazione tra le varie feature presenti nel dataset mediante la creazione di una matrice di correlazione. Questa matrice fornisce un'indicazione visiva delle relazioni lineari tra le diverse caratteristiche. È stato osservato che l'intervallo delle feature dalla S12 alla S22 presenta una correlazione significativamente vicina a 1, suggerendo una forte relazione tra queste feature.

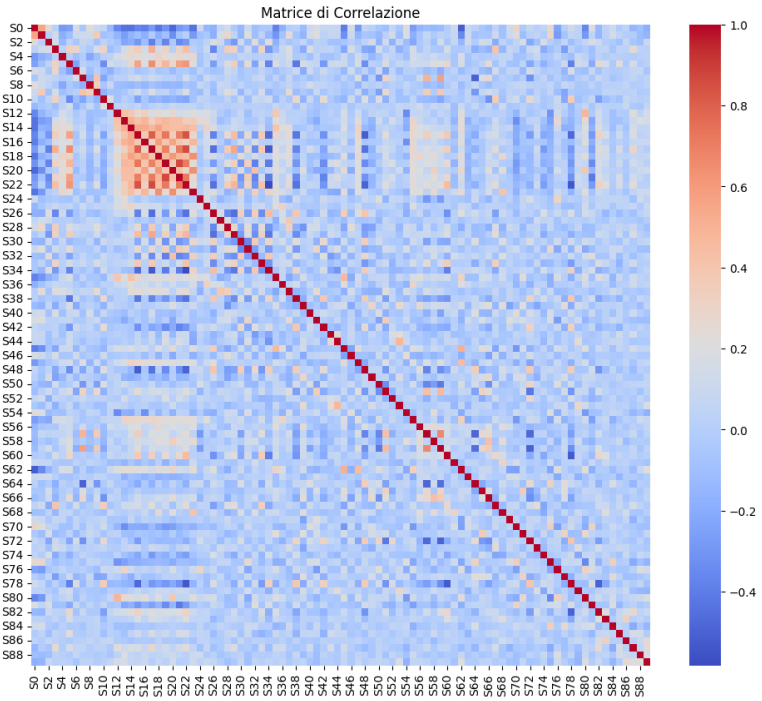


Figure 2: Matrice di correlazione

Per visualizzare più chiaramente questa relazione, è stato creato uno scatter plot che ha permesso di identificare la direzione e l'intensità della correlazione in questo intervallo.

Figure 3: Scatter plot dell'intervallo di features S12 e S22

Nello specifico, tutti i punti dello scatter plot sono concentrati nel quadrante in basso a sinistra del grafico. Questo andamento indica una forte correlazione negativa tra le feature rappresentate, il che significa che quando il valore di una feature aumenta, il valore dell'altra tende a diminuire e viceversa. Tale correlazione negativa suggerisce che le caratteristiche rappresentate da queste feature potrebbero essere in qualche modo "opposte" rispetto all'anno di pubblicazione delle canzoni.

Successivamente, è stata esaminata la distribuzione delle feature utilizzando boxplot e istogrammi. I boxplot hanno permesso di individuare eventuali outlier e valori estremi nelle distribuzioni delle feature, mentre gli istogrammi hanno fornito una visione più dettagliata delle distribuzioni stesse. Questo ha aiutato a comprendere meglio la variabilità dei dati e a identificare eventuali pattern o tendenze all'interno dell'intervallo delle feature considerate.

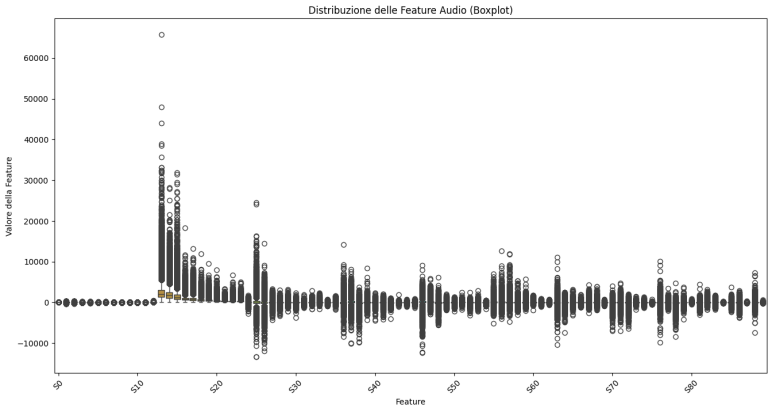


Figure 4: Box plot delle features

Nel boxplot delle features dalla S0 alla S12, si osserva una distribuzione concentrata intorno allo zero, suggerendo una bassa variabilità che potrebbe non essere rilevante per la predizione dell'anno di pubblicazione delle canzoni.

D'altra parte, le features dalla S13 alla S15 presentano valori molto elevati nel boxplot, ma mancano di valori negativi, indicando una predominanza di valori positivi che potrebbero essere importanti per la predizione. Tuttavia, la mancanza di variazioni negative potrebbe limitare la capacità del modello di apprendere relazioni complesse nei dati.

Le features dalla S16 fino alla fine mostrano una distribuzione più ampia dei valori nel boxplot, con presenza sia di valori positivi che negativi. Questo suggerisce una maggiore variabilità e bilanciamento rispetto alle altre features.

Per ottenere una visione più dettagliata di queste tre feature (S13, S14, S15), sono stati creati degli istogrammi che mostrano la distribuzione dei valori.

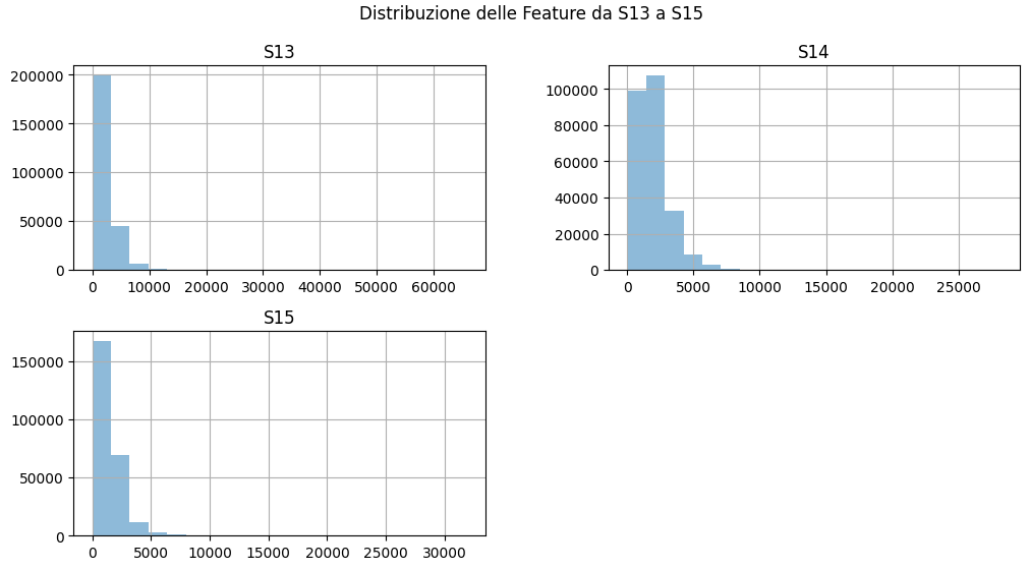


Figure 5: Istrogrammi delle features S13, S14 e S15

Gli istogrammi confermano che queste feature spresentano una distribuzione con barre relativamente alte per i valori più bassi, che diminuiscono man mano che si procede verso valori più alti. Questo suggerisce la presenza di numerosi punti dati con valori bassi, ma con una diminuzione significativa della frequenza per valori più alti, indicando una coda lunga verso destra della distribuzione.

2 Metodologia

Per affrontare l'obiettivo di predire accuratamente l'anno di pubblicazione delle canzoni basandosi sulle caratteristiche audio estratte, è essenziale adottare un approccio metodologico chiaro e ben strutturato.
In questo capitolo, esamineremo le principali fasi della metodologia impiegata, focalizzandoci sulla pulizia dei dati e sulle diverse tecniche di pre-processing utilizzate per preparare il dataset al calcolo con i vari modelli selezionati.

2.1 Pulizia Dati

Durante la fase di preparazione e pulizia dei dati, sono state eseguite diverse operazioni per garantire l'integrità e la qualità del dataset. Inizialmente, è stato eseguito un processo di filtraggio dei duplicati, il quale ha identificato la presenza di 52 righe duplicate nel dataset. Queste righe sono state prontamente rimosse, eliminando potenziali fonti di inconsistenza e sovrastima nella fase successiva di analisi.

Successivamente, è stata dedicata particolare attenzione alla gestione dei dati mancanti. Nonostante l'accurata verifica, non sono stati individuati valori mancanti durante l'analisi dei dati.

2.2 PreProcessing

Durante la fase di preprocessing dei dati, sono state utilizzate due tecniche di scalatura: lo StandardScaler e il MinMaxScaler. Entrambe queste tecniche sono utilizzate per trasformare le feature in modo che abbiano una scala specifica, ma operano in modi leggermente diversi.

2.2.1 Standard Scaler

Lo StandardScaler standardizza le feature sottraendo loro la media e dividendo per la deviazione standard di ciascuna feature. In pratica, questo significa che la media di ogni feature diventa zero e la deviazione standard diventa uno. Questo processo garantisce che le feature abbiano una distribuzione con media zero e deviazione standard unitaria, il che può essere utile per algoritmi di apprendimento automatico che si basano su questa assunzione, come ad esempio la regressione lineare. Inoltre, lo StandardScaler non riduce l'intervallo delle feature, ma le rende più facilmente confrontabili tra loro.

2.2.2 MinMaxScaler

Il MinMaxScaler, d'altra parte, normalizza le feature portandole tutte nell'intervallo specificato, di solito tra 0 e 1. Questo avviene tramite la trasformazione lineare delle feature in modo che il valore minimo diventi 0 e il valore massimo diventi 1. Questa tecnica è utile quando si desidera mantenere la relativa importanza delle feature e garantire che siano tutte comprese nella stessa scala. Tuttavia, il MinMaxScaler può non essere adatto quando le feature hanno deviazioni standard molto diverse o contengono valori anomali, in quanto può portare a una riduzione della varianza delle feature.

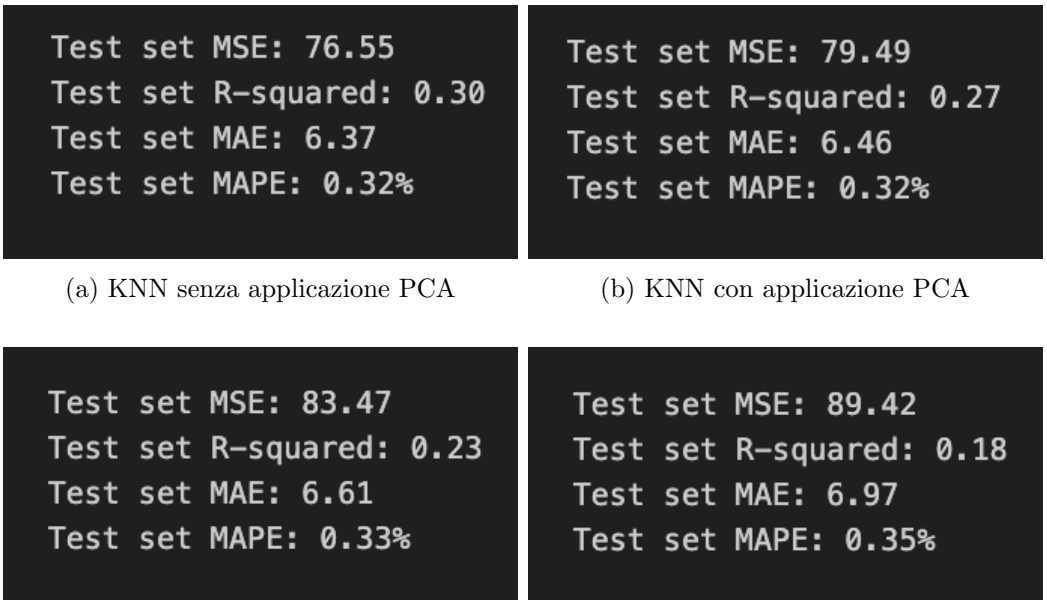
2.2.3 Funzione di PreProcessing

Per automatizzare il processo di preprocessamento dei dati, è stata sviluppata una funzione denominata preprocessTrain. Questa funzione consente di applicare diverse tecniche di normalizzazione e scalatura, inclusi lo StandardScaler e il MinMaxScaler, al set di dati di addestra-

mento e di test. La funzione restituisce il set di dati preprocessato pronti per l'addestramento dei modelli di regressione.

2.3 PCA

Durante l'analisi, è stata tentata l'implementazione della Principal Component Analysis (PCA) per ridurre la dimensionalità delle feature, passando da 90 a 56 features mantenendo il 95% della varianza. Tuttavia, questo approccio non ha migliorato le prestazioni predittive dei modelli di regressione rispetto alle feature originali. Di conseguenza, si è deciso di non adottare la PCA nella fase finale dell'analisi, preferendo mantenere le feature originali per garantire risultati più affidabili e rappresentativi.



(a) KNN senza applicazione PCA (b) KNN con applicazione PCA

(c) LR senza applicazione PCA (d) LR con applicazione PCA

Figure 6: Esempio applicazione PCA

2.4 Outlier

Durante la fase di esplorazione dei dati, abbiamo utilizzato il valore z-score con una soglia di 3 e la tecnica degli intervalli quartili per individuare gli outlier nel dataset. Tuttavia, l'applicazione di tali metodi ha comportato la rimozione di un considerevole numero di dati, sollevando preoccupazioni sulla possibile perdita di significatività del nostro set di dati.

Di conseguenza, abbiamo optato per suddividere le features in quartili, seguendo la tecnica del quartile. Successivamente, abbiamo esaminato diverse strategie per gestire gli outlier identificati, valutando sia la rimozione che la sostituzione con la media e la mediana. Tuttavia, alla luce delle incertezze riguardo al significato specifico di ciascuna osservazione, abbiamo deciso di preservare gli outlier all'interno del dataset. Questa scelta è stata fatta per evitare una potenziale alterazione delle caratteristiche intrinseche del nostro insieme di dati.

2.5 Allenamento

Considerando che il nostro progetto si concentra sulla risoluzione di un problema di regressione, abbiamo adottato l'approccio dell'apprendimento supervisionato. Questo metodo implica l'utilizzo di un insieme di dati precedentemente etichettato per addestrare i modelli di

regressione al fine di predire l'output desiderato.

Per assicurarci di valutare accuratamente le prestazioni dei modelli, abbiamo suddiviso il nostro dataset in tre parti: un set di addestramento, composto dall'80% dei dati, e due set di validazione e test, rappresentanti il restante 20%. Questa divisione ci ha consentito di addestrare i modelli sulla maggior parte dei dati disponibili, mentre allo stesso tempo valutiamo le loro prestazioni su insiemi di dati indipendenti durante le fasi di validazione e test.

3 Implementazione

Il progetto è strutturato attorno a tre funzionalità fondamentali, ciascuna progettata per esplorare e sfruttare diverse tecniche di apprendimento.

La prima parte del progetto si concentra sull'applicazione di tecniche di Machine Learning supervisionato tradizionali. In particolare, abbiamo esaminato modelli come la Regressione Lineare (LR), il Random Forest Regressor (RF), il Support Vector Regressor (SVR) e il K-Nearest Neighbors Regressor (KNR). Lo scopo di questa fase è valutare le performance dei modelli classici nel panorama del machine learning tradizionale, al fine di identificare il modello che offre le migliori prestazioni sui dati.

La parte centrale è dedicata all'utilizzo di tecniche di Machine Learning supervisionato basate su reti neurali, con un'architettura Feed-Forward. L'obiettivo di questa fase è valutare se l'implementazione delle reti neurali possa portare a miglioramenti nelle performance di previsione rispetto ai modelli tradizionali.

La parte finale dello studio si concentra sull'utilizzo di tecniche di Machine Learning supervisionato con modelli deep learning progettati specificamente per dati tabulari. Sono stati esplorati approcci come TabNet e TabTransformer, per comprendere come modelli di deep learning possano gestire dati strutturati come quelli presenti nel nostro dataset.

In ciascuna di queste tre fasi, il dataset è stato inizialmente suddiviso in set di addestramento (80%) e di test (20%). Successivamente, il set di addestramento è stato ulteriormente suddiviso in set di addestramento finale (80%) e di convalida (20%). Le metriche utilizzate per valutare le prestazioni dei modelli includono il **Mean Squared Error (MSE)**, il **Mean Absolute Error (MAE)**, il coefficiente **R-squared (R2)** e il **Mean Absolute Percentage Error (MAPE)**.

I modelli finali tuttavia saranno addestrati sull'intero file *train.csv*.

3.1 Modelli Scikit-learn

3.1.1 Linear Regression

Per valutare l'efficacia del modello `Linear Regression`, sono state considerate tre modalità di preprocessing: nessun preprocessing, standardizzazione e Min-Max. Per ciascuna modalità, è stato eseguito un processo di addestramento e convalida del modello di regressione lineare. Questo processo ha coinvolto la cross-validation, che ha suddiviso i dati in cinque fold per garantire una valutazione robusta delle prestazioni del modello.

Durante la cross-validation, è stata calcolata la media del coefficiente R-squared su tutti i fold, fornendo una stima dell'accuratezza predittiva del modello per ciascuna modalità di preprocessing. In questo modo, è stata identificata la modalità di preprocessing che massimizzava il coefficiente R-squared, indicando quindi la migliore tecnica da utilizzare per ottenere le prestazioni ottimali del modello.

E' stata identificata come migliore tecnica di preprocessing, relativa al modello, il MinMax Scaler. Successivamente, sono state valutate le prestazioni del modello utilizzando diverse metriche, tra cui il Mean Squared Error (MSE), il Mean Absolute Error (MAE), il coefficiente

R-squared (R2) e il Mean Absolute Percentage Error (MAPE).

3.1.2 Random Forest Regressor

Per valutare l'efficacia del modello `Random Forest Regressor` sono state considerate tre modalità di preprocessing: nessun preprocessing, standardizzazione e Min-Max.
Inoltre, sono stati considerati i seguenti iperparametri:

- numero di estimatori (50, 100, 150)
- profondità massima degli alberi (nessuna, 3, 5, 7, 9)

Un approccio di cross-validation con una K-Fold di 5 è stato implementato per valutare le performance del modello con diverse tecniche di preprocessing e identificare la migliore configurazione degli iperparametri con un calcolo della media del coefficiente R-squared su tutti i fold, fornendo una stima dell'accuratezza predittiva del modello.
Come migliore configurazione è stata indentificata la seguente:

- preprocessing : standard
- numero di estimatori : 50
- profondità massima degli alberi : nessuna

Successivamente, sono state valutate le prestazioni del modello utilizzando diverse metriche, tra cui il Mean Squared Error (MSE), il Mean Absolute Error (MAE), il coefficiente R-squared (R2) e il Mean Absolute Percentage Error (MAPE).

3.1.3 Support Vector Regressor

Per valutare l'efficacia del modello `Support Vector Regressor`, sono state considerate tre modalità di preprocessing: nessun preprocessing, standardizzazione e Min-Max. Per ciascuna modalità, è stato eseguito un processo di addestramento e convalida del modello di regressione lineare. Questo processo ha coinvolto la cross-validation, che ha suddiviso i dati in cinque fold per garantire una valutazione robusta delle prestazioni del modello.

Dato il grande volume di dati nel dataset e i lunghi tempi di calcolo richiesti per la ricerca degli iperparametri ottimali, abbiamo scelto di utilizzare il modello con la sua configurazione predefinita.

Durante la cross-validation, è stata calcolata la media del coefficiente R-squared su tutti i fold, fornendo una stima dell'accuratezza predittiva del modello per ciascuna modalità di preprocessing. In questo modo, è stata identificata la modalità di preprocessing che massimizzava il coefficiente R-squared, indicando quindi la migliore tecnica da utilizzare per ottenere le prestazioni ottimali del modello.

E' stata identificata come migliore tecnica di preprocessing, relativa al modello, lo standard. Successivamente, sono state valutate le prestazioni del modello utilizzando diverse metriche, tra cui il Mean Squared Error (MSE), il Mean Absolute Error (MAE), il coefficiente R-squared (R2) e il Mean Absolute Percentage Error (MAPE).

3.1.4 K-Nearest Regressor

Per valutare l'efficacia del modello `KNN Regressor` sono state considerate tre modalità di preprocessing: nessun preprocessing, standardizzazione e Min-Max, inoltre sono stati considerati i seguenti iperparametri:

- n_neighbors : 5, 7, 9, 11, 15, 20, 25, 30, 35, 40,45

Un approccio di cross-validation con una K-Fold di 5 è stato implementato per valutare le performance del modello con diverse tecniche di preprocessing e identificare la migliore configurazione degli iperparametri con un calcolo della media del coefficiente R-squared su tutti i fold, fornendo una stima dell'accuratezza predittiva del modello Come migliore configurazione è stata indentificata la seguente:

- preprocessing : MinMax
- n_neighbors : 40

Successivamente, sono state valutate le prestazioni del modello utilizzando diverse metriche, tra cui il Mean Squared Error (MSE), il Mean Absolute Error (MAE), il coefficiente R-squared (R2) e il Mean Absolute Percentage Error (MAPE).

3.2 Reti Neurali

La funzionalità relativa alle Reti Neurali prevedeva l'implementazione di una Rete Neurale Feed-Forward.

La rete è composta da tre strati: uno strato di input con dimensioni corrispondenti al numero di features nel dataset, due strati nascosti con rispettivamente 64 e 32 neuroni, e uno strato di output con un singolo neurone per la previsione dell'anno di uscita della canzone. La scelta di utilizzare uno strato di output singolo è coerente con la natura della regressione, dove stiamo cercando di prevedere un valore continuo.

Per quanto riguarda la funzione di attivazione, è stata selezionata la funzione ReLU (Rectified Linear Unit) tra i due strati nascosti. La ReLU è comunemente utilizzata nelle reti neurali Feed Forward per introdurre non-linearità, consentendo alla rete di apprendere relazioni complesse nei dati. La rete ha un singolo neurone nell'ultimo strato, che restituisce l'output previsto. Poiché si tratta di un problema di regressione, la rete restituisce un singolo valore numerico che rappresenta l'anno previsto.

La perdita utilizzata per l'ottimizzazione è il Mean Squared Error (MSE), poiché si adatta bene a problemi di regressione, misurando la discrepanza quadratica media tra le previsioni della rete e i valori reali dell'anno di uscita delle canzoni. Per quanto riguarda l'ottimizzatore, è stato scelto l'Adam con un tasso di apprendimento iniziale di 0.0001. L'Adam è noto per la sua efficacia in ottimizzazione e adattamento automatico del tasso di apprendimento.

Abbiamo utilizzato i DataLoader di PyTorch per gestire il caricamento efficiente dei dati durante l'addestramento.

Infine, abbiamo valutato le performance del modello prima e dopo l'addestramento utilizzando metriche significative come il Mean Squared Error (MSE), il Mean Absolute Error (MAE), il Mean Absolute Percentage Error (MAPE), e il coefficiente R-squared (R2) per ottenere una comprensione completa delle capacità predittive della rete neurale Feed Forward.

3.3 Tabular

La terza fase del nostro progetto è dedicata all'esplorazione e all'implementazione di tecniche di Machine Learning supervisionato specificamente progettate per dati tabulari. Questo approccio è particolarmente rilevante per il nostro Dataset, composto da feature audio strutturate. Nel contesto di questa funzionalità, abbiamo adottato il modello TabNet e TabTransformer, architetture avanzate basate su reti neurali progettata per affrontare compiti di regressione su dati tabulari.

3.3.1 TabNet

La configurazione del modello TabNet è stata personalizzata attraverso diverse opzioni. Ad esempio, abbiamo regolato il numero di decisioni il numero di attenzioni con valore 16, il numero di passaggi a 3, e il valore di gamma a 1 per ottimizzare le prestazioni del modello nel contesto specifico del nostro problema di regressione.

Successivamente, abbiamo utilizzato la libreria 'pytorch-tabular' per definire la configurazione del modello, specificando dettagli come la variabile target, le colonne continue e altre opzioni specifiche del modello. Abbiamo definito anche configurazioni per l'ottimizzazione, il training e l'esperimento.

Il modello TabNet è stato quindi creato e addestrato utilizzando il set di addestramento, con un numero massimo di epoche, un batch size di 256, e una pazienza di 10 epoche per l'early stopping. Durante il processo di addestramento, il numero di workers nei DataLoader è stato aumentato per migliorare l'efficienza computazionale.

Per valutare le performance del modello, abbiamo utilizzato il set di train, applicando le metriche specificate in fase di configurazione. Il processo di valutazione ha incluso il calcolo dell'MSE, MAE, MAPE e il coefficiente R-squared.

3.3.2 TabTransformer

La parte finale della nostra indagine si concentra sull'impiego del modello TabTransformer.

La configurazione del modello TabTransformer è stata definita attraverso la libreria 'pytorch-tabular', specificando la variabile target e le colonne continue presenti nel nostro Dataset.

Il processo di addestramento del modello è stato personalizzato attraverso configurazioni quali il batch size (256), il numero massimo di epoche (100), la pazienza per l'early stopping (10), e l'acceleratore utilizzato (CPU). Abbiamo disabilitato la ricerca automatica del tasso di apprendimento e, successivamente, abbiamo addestrato il modello utilizzando il set di addestramento e la convalida.

Per valutare le performance del modello, abbiamo utilizzato il set di train, applicando le metriche specificate in fase di configurazione. Il processo di valutazione ha incluso il calcolo dell'MSE, MAE, MAPE e il coefficiente R-squared.

4 Risultati

I seguenti risultati di prova sono stati calcolati splittando il file "train.csv" in fase di allenamento (80%), e di convalida e test (20%), in modo tale da verificare la bontà dei modelli.

Per ogni modello, vengono presentati i valori delle metriche ottenute dalle migliori configurazioni discusse nel capitolo precedente. Inoltre, per i modelli Scikit-learn, sono inclusi due grafici: uno che confronta i valori predetti con quelli reali e un altro che illustra la distribuzione degli errori di previsione.

4.1 Linear Regression

MSE	R-squared	MAE (%)	MAPE
83.46	0.23	6.60	0.33

Table 3: Risultati LR

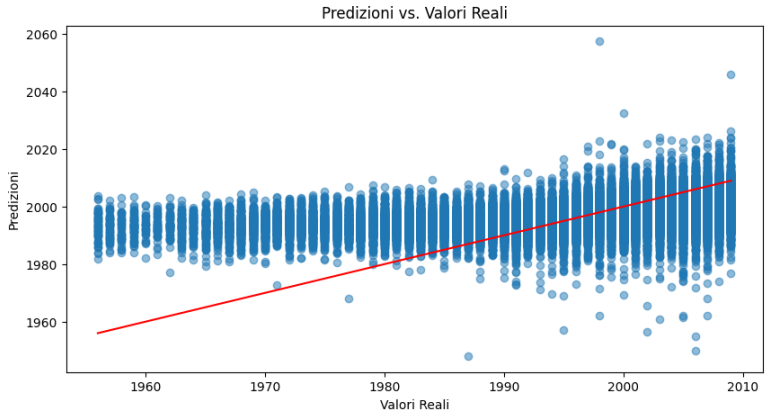


Figure 7: Valori previsione vs valori reali Linear Regression

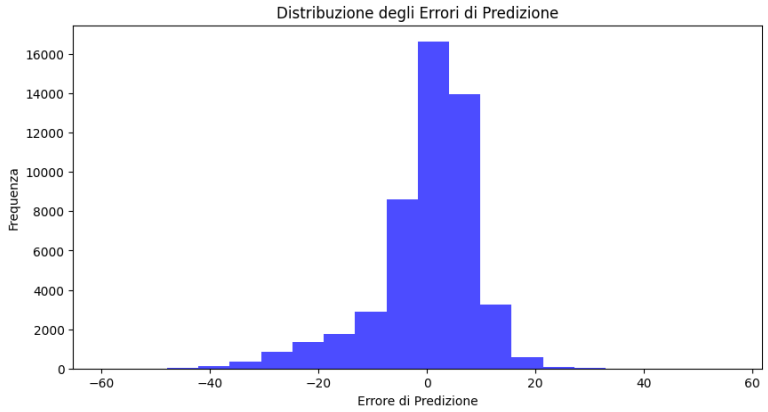


Figure 8: Distribuzione errori di Predizione Linear Regression

4.2 Random Forest Regression

MSE	R-squared	MAE (%)	MAPE
79.90	0.28	6.43	0.32

Table 4: Risultati RF

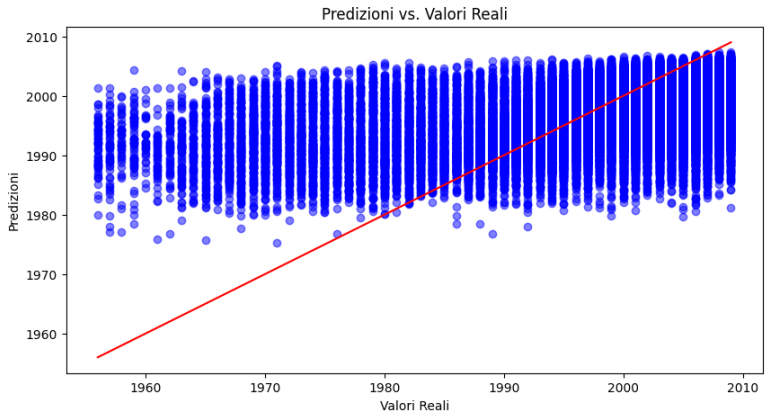


Figure 9: Valori previsione vs valori reali Random Forest Regression

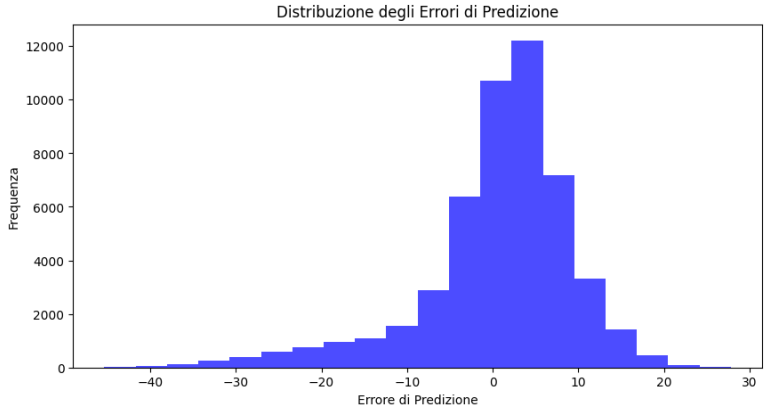


Figure 10: Distribuzione errori di Predizione Random Forest sRegression

4.3 Support Vector Regressor

MSE	R-squared	MAE (%)	MAPE
77.41	0.29	5.79	0.29

Table 5: Risultati Support Vector Regressor

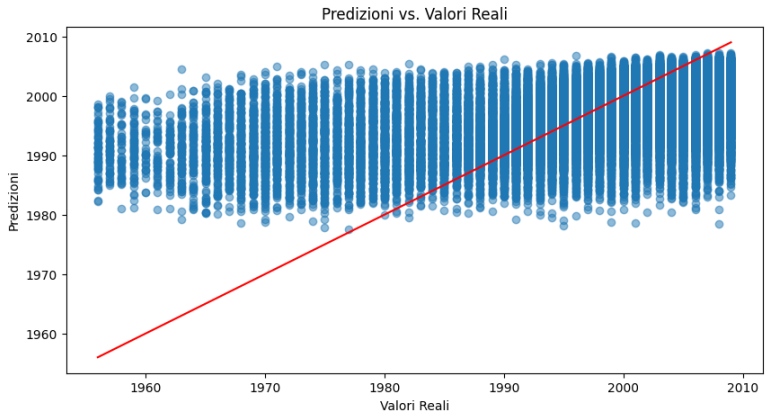


Figure 11: Valori previsione vs valori reali Support Vector Regressor

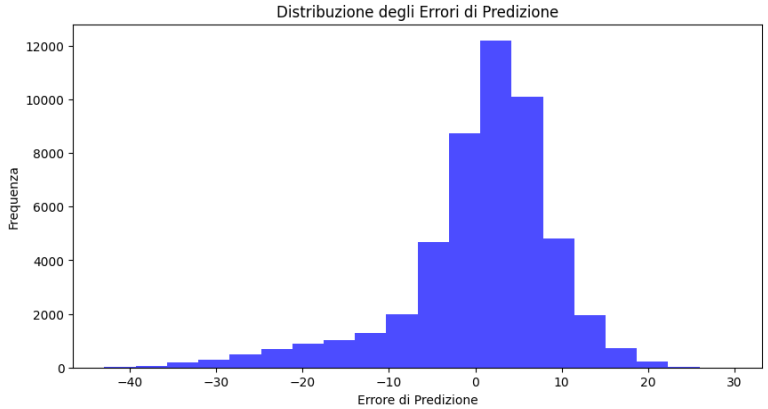


Figure 12: Distribuzione errori di Predizione Support Vector Regressor

4.4 K-Nearest Regressor

MSE	R-squared	MAE (%)	MAPE
76.55	0.30	6.37	0.32

Table 6: Risultati K-Nearest Regressor

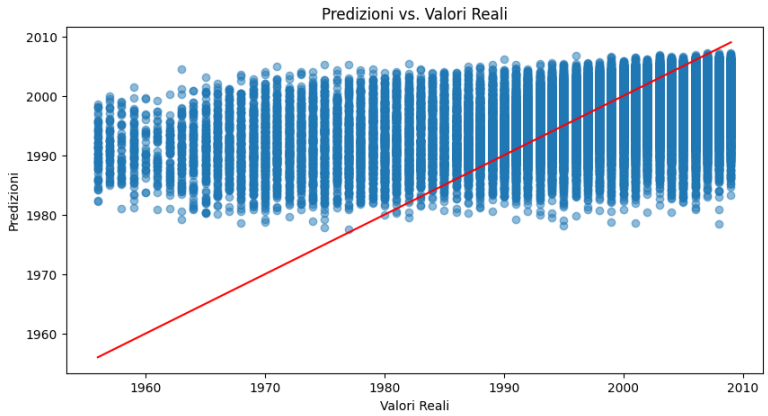


Figure 13: Valori previsione vs valori reali K-Nearest Regressor

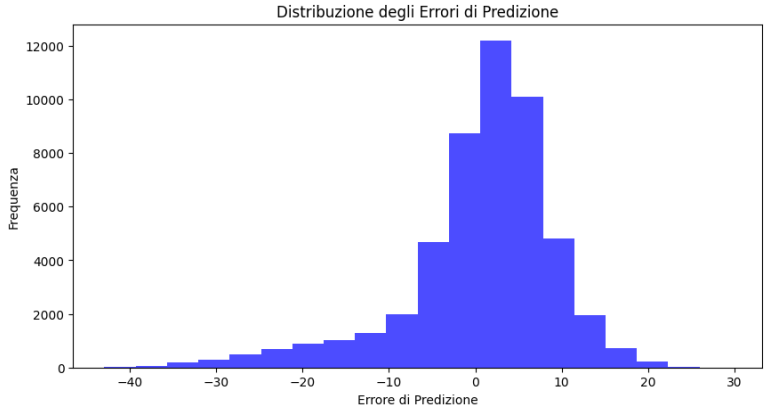


Figure 14: Distribuzione errori di Predizione K-Nearest Regressor

4.5 Rete Neurale Feed-Forward

MSE	R-squared	MAE (%)	MAPE
79.09	0.27	6.32	0.31

Table 7: Risultati reti Neurali

4.6 Tab Net

MSE	R-squared	MAE (%)	MAPE
85.49	0.20	6.72	0.33

Table 8: Risultati TabNet

4.7 Tab Transformer

MSE	R-squared	MAE (%)	MAPE
84.16	0.22	6.60	0.33

Table 9: Risultati TabTransformer