

Alma Mater Studiorum – Università di Bologna  
Laurea Magistrale in Informatica  
Corso di Data Analytics

# PROGETTO DATA ANALYTICS

---

Martina Daghia - 0001097932 - martina.daghia@studio.unibo.it  
Martina Zauli - 0001097933 - martina.zauli@studio.unibo.it

# Indice

## 1. Introduzione

## 2. Metodologia

- Data Acquisition
- Data Visualization
- Data Preprocess

## 3. Implementazione

- Modelli Scikit-learn
- Rete Neurale Feed-Forward
- Modelli Tabulari

## 4. Risultati

# Introduzione

---

**Task:** regressione

**Variabili input:** N feature audio di una canzone

**Variabile output:** anno in cui è stata pubblicata la canzone



**Obiettivo:** predire l'anno di pubblicazione di una canzone basandosi sulle feature della sua traccia audio

# Data Acquisition

Dataset utilizzato:

- *train.csv*
- **252.175 righe e 91 colonne**

*canzone*

*features  
audio*

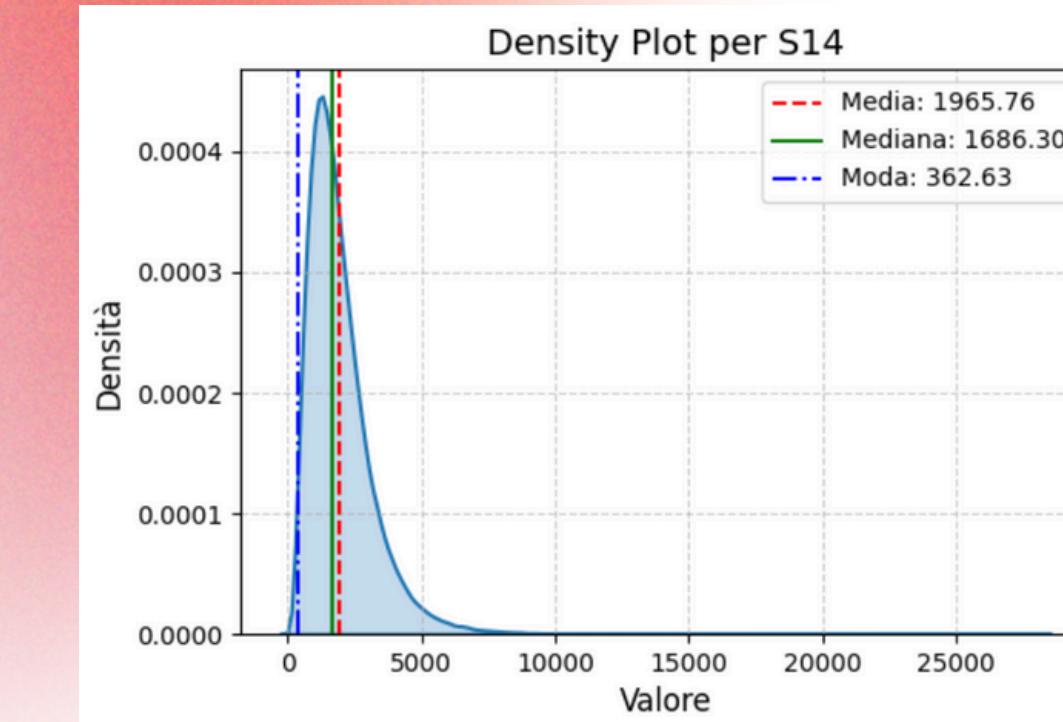
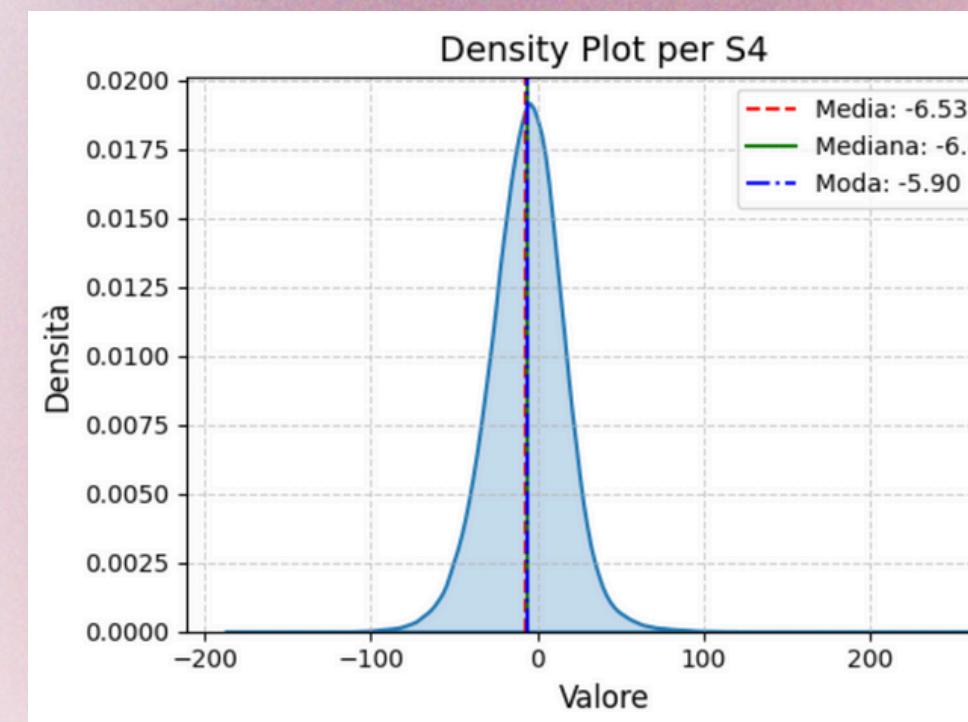


Year	S0	S1	S2	S3	...	S89
2007	44.76752	114.82099	3.83239	27.99928	...	31.32820
2004	52.28942	75.73319	11.35941	-6.20582	...	3.86143
2005	33.81773	6.18222	9.45456	17.85216	...	35.74749
1998	41.60866	3.17811	-3.97174	23.53564	...	3.60432
1987	44.49525	-32.25270	58.08217	3.73684	...	30.11015

# Data Visualization

- Distribuzione delle canzoni per anno
- Distribuzione delle feature audio (media, mediana, moda)

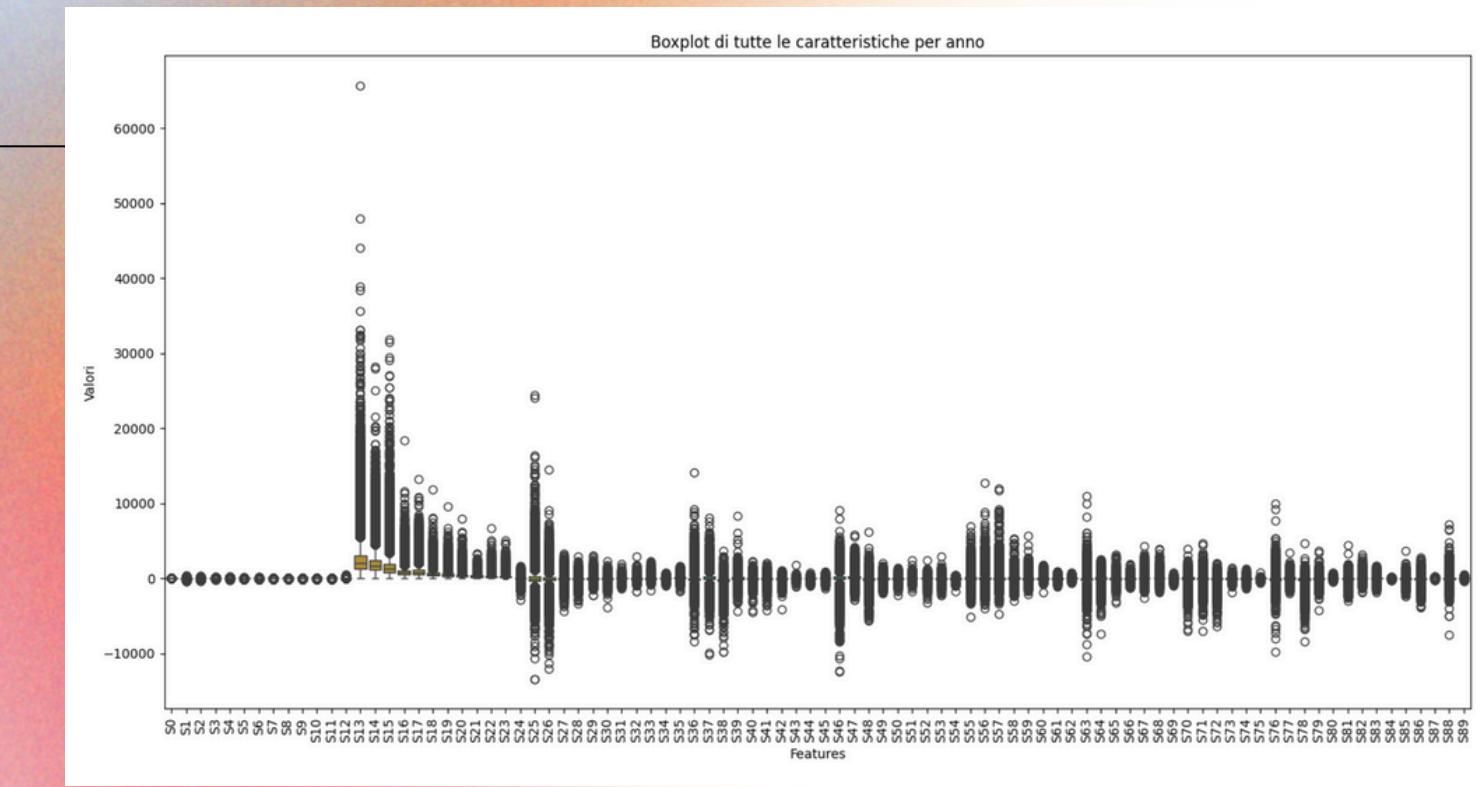
2000 - 2009	139.859 canzoni
1990 - 1999	68.371 canzoni



# Data Visualization

## Boxplot:

- Visualizzare la distribuzione delle singole feature audio
- Identificare la presenza di outliers



### DA S13 A S19

- Distribuzione solo positiva
- Numerosi outliers

### DA SO A S12

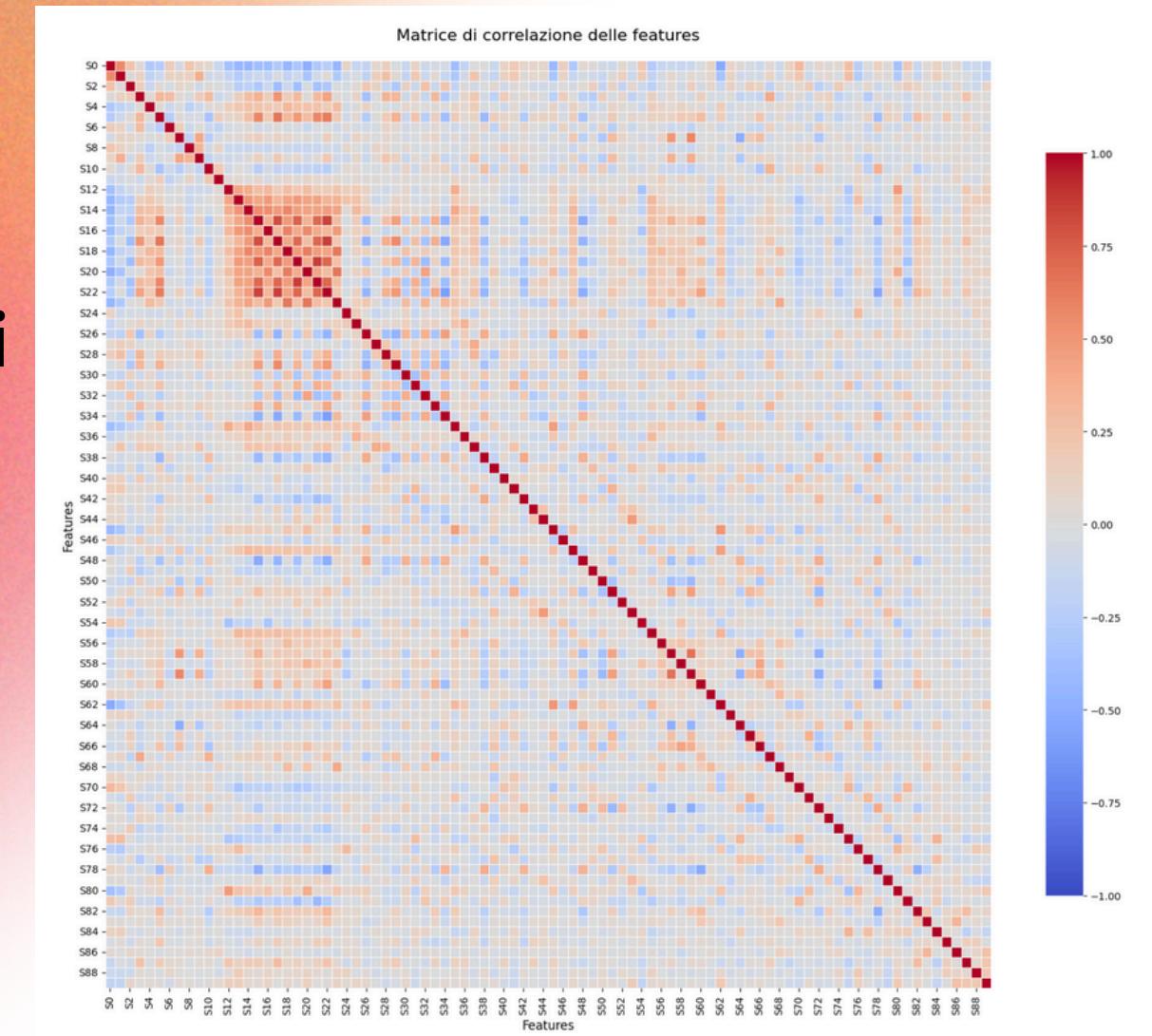
- Distribuzione più compatta (intorno allo zero)
- Bassa variabilità

### ALTRÉ FEATURES

- Valori positivi e negativi
- Distribuzione più ampia
- Maggiore variabilità

# Data Visualization

- **Matrice di correlazione:**
    - analizzare le relazioni tra le features audio
  - Da S12 a S23: correlazione **elevata** con info **ridondanti**
    - ridurre dimensionalità con **PCA**
  - Maggioranza features **non** presentano **forti correlazioni**
    - informazioni uniche utili ai modelli di ML
- Mantenuta rappresentazione completa del dataset



# Data Preprocess

---

- O valori **mancanti**
- 52 valori **duplicati** (rimossi)
- **Outliers**
  - influenzano le prestazioni dei modelli
  - sostituiti con la mediana per migliorare la qualità dei dati per i modelli

Colonna	Numero outliers	Percentuale outliers	Numero nuovi outliers	Percentuale nuovi outliers
S0	5306	2.10%	1492	0.59%
S1	7313	2.90%	2684	1.06%
S2	7484	2.97%	2657	1.05%
...	...	...	...	...
S88	19031	7.55%	9973	3.96%
S89	19330	7.67%	10960	4.35%

# Data Preprocess

---

## Standard Scaler:

- per garantire che tutte le feature abbiano **media 0 e deviazione standard 1** (distribuzione uniforme)
- **migliore** performance rispetto al Min-Max Scaler:
  - outliers ancora presenti dopo la rimozione, potrebbero influenzare la scala dei dati
  - per alcuni modelli di ML meglio una distribuzione normale

# Implementazione

## Funzionalità

---

1

### Tecniche ML supervisionato tradizionali

- Random Forest
- Linear Regression
- Support Vector Machines
- K-Nearest neighbors

2

### Tecniche ML supervisionato basate su reti

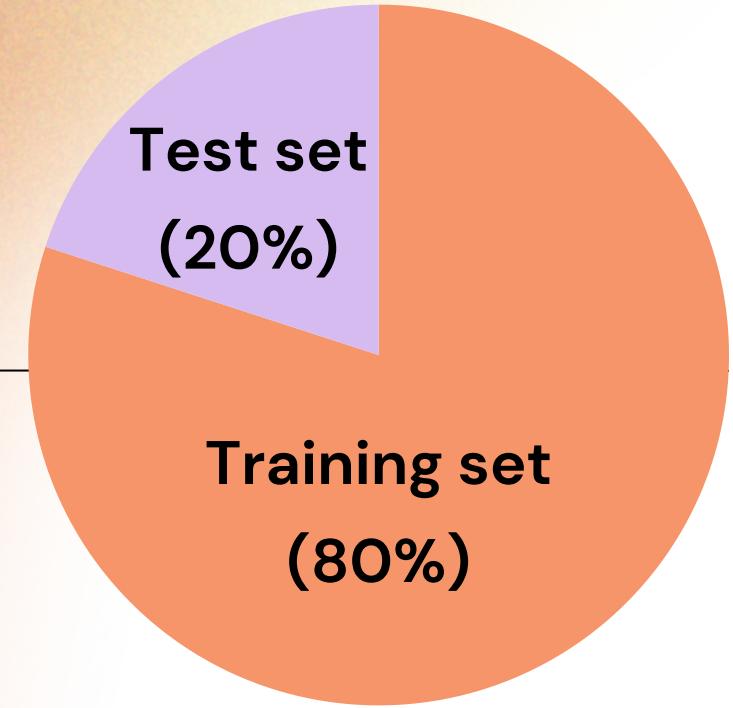
- Rete neurale con architettura Feed-Forward

3

### Tecniche ML supervisionato per dati tabulari

- TabNet
- TabTransformer

# Modelli Scikit-learn



- Suddivisione dataset:
- Ricerca iperparametri:
  - **Grid Search**: diverse combinazioni di iperparametri per ottimizzarli
  - **Cross-validation** (k-fold con k=5) sui vari sottinsiemi di dati
- Valutazione del modello

MSE Mean Squared Error	MAE Mean Absolute Error	MAPE Mean Absolute Percentage Error	R2 R-squared
---------------------------	----------------------------	--	-----------------

- Addestramento finale sull'intero dataset con migliore combinazione di iperparametri

# Random Forest

~~no applicazione Standard Scaler~~

applicato Standard Scaler

```
n_estimators=450, max_depth=170,  
min_samples_split=8,  
min_samples_leaf=5
```

## SVM

~~no applicazione Standard Scaler~~

applicato Standard Scaler

```
C = 0.4,  
kernel = 'rbf',  
gamma = 'scale',  
epsilon = 0.5
```

# Linear Regression

~~applicata PCA~~

applicato Standard Scaler

## KNN

~~no applicazione Standard Scaler~~

applicato Standard Scaler

```
n_neighbors = 17  
weights = 'distance'  
metric = 'manhattan'
```

# Rete neurale Feed-Forward

## Architettura

---

- **Input layer:** ogni neurone dell'input layer rappresenta una delle feature del dataset
- **2 Hidden layers** con 64 o 128 neuroni (ottimizzati tramite grid search)
  - **batch normalization:** migliora la stabilità e accelera l'addestramento
  - funzione di attivazione **ReLU:** introduce non linearità tra gli strati, quindi la rete apprende relazioni complesse
  - **dropout** per prevenire overfitting, disattivando casualmente un numero di neuroni durante l'addestramento (migliora la generalizzazione del modello)
- **Output layer:** un singolo neurone per la predizione continua (problema di regressione)

# Rete neurale Feed-Forward

## Processo di addestramento

- **Adam** (ottimizzatore) che utilizza *weight decay* per migliorare la generalizzazione e regolarizzazione del modello
- **Early stopping** per interrompere l'addestramento quando la performance sulla validazione non migliora per un certo numero di epochhe (*patience*)
- **Cross-validation** a 5 fold per trovare i migliori iperparametri
- Funzione di perdita: Mean Squared Error (**MSE**) per minimizzare l'errore nelle predizioni continue (Year)
- Numero di **epoche**: modello addestrato per X epoche

```
learning_rates = [0.1]
hidden_size1_values = [64, 128]
hidden_size2_values = [64, 128]
activation_functions = [nn.ReLU]
dropout_rate = 0.5
weight_decay = 0.01
num_epochs = 50
batch_size = 16
patience = 10
```

# Rete neurale Feed-Forward

## Addestramento finale

---

**Migliore combinazione di iperparametri**

```
Best Hyperparameters: {  
    'learning_rate': 0.1,  
    'hidden_size1': 64,  
    'hidden_size2': 128,  
    'activation_function': 'ReLU',  
    'dropout_rate': 0.5,  
    'weight_decay': 0.01,  
    'best_epoch': 19  
}
```

# Tabular

---

Modelli di deep learning progettati specificamente per i dati tabulari

- **TabNet**
  - Utilizza una rete neurale con **meccanismi di attenzione** per selezionare le feature rilevanti, migliorando così l'efficienza e l'interpretabilità
- **TabTransformer**
  - Sfrutta l'architettura dei **transformer** (originariamente sviluppata per il NLP) basata sui **meccanismi di attenzione multi-head** e consente di gestire relazioni più complesse tra le feature

# Tabular

## TabNet e TabTransformer

---

- **Configurazione** di TabNet e TabTransformer
  - Per TabNet:
    - **Dimensione del layer di predizione** ( $n_d$ ): numero di unità nel layer di predizione per apprendere pattern complessi
    - **Dimensione del layer di attenzione** ( $n_a$ ): capacità del modello di identificare quali features sono più rilevanti per ciascuna iterazione di addestramento, sfruttando un meccanismo di attenzione.
  - **Batch size**: numero di campioni usati in ogni aggiornamento dei pesi durante l'addestramento
  - **Early stopping**
  - Numero di **epoch**
- Imposta ricerca degli iperparametri con **grid-search** (confrontando miglior MSE)
- Addestramento su tutto il dataset con i miglior iperparametri

# Tabular

## TabNet e TabTransformer

---

### TabNet

Migliori Parametri:

```
'batch_size': 256, 'gamma': 1.0,  
'max_epochs': 70, 'n_a': 32,  
'n_d': 32, 'n_steps': 4
```

### TabTransformer

Migliori Parametri:

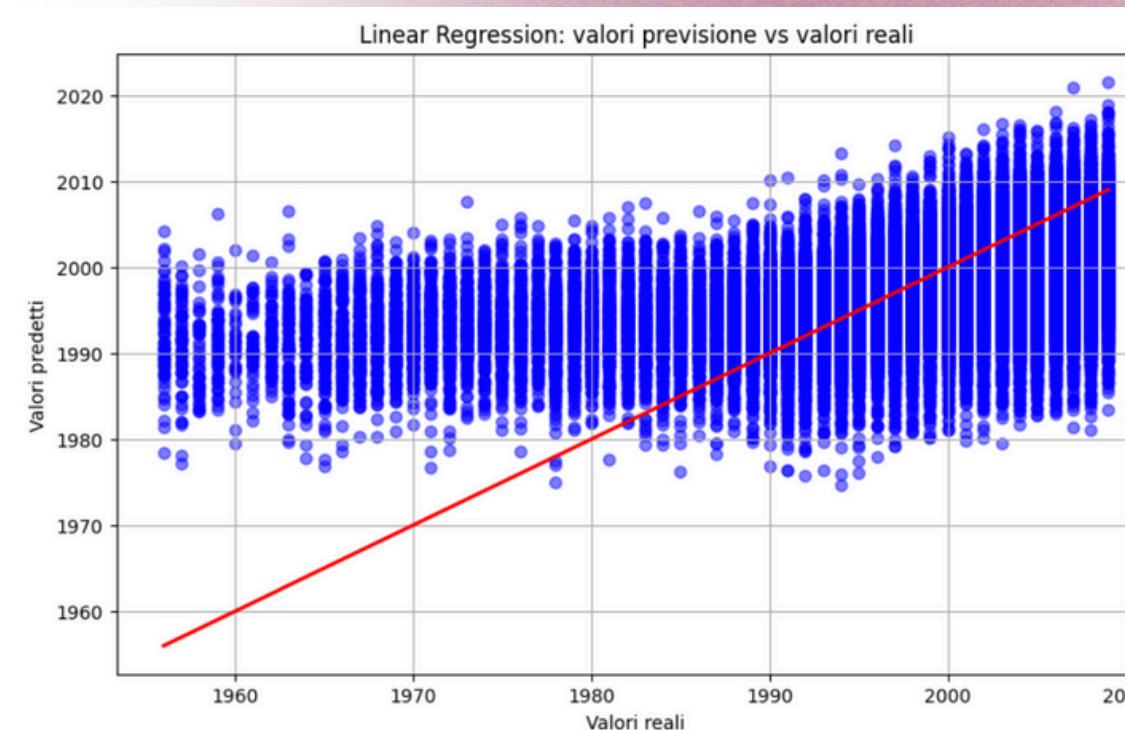
```
'batch_size': 128, 'max_epochs':  
100
```

# Risultati

## Linear regression e random forest

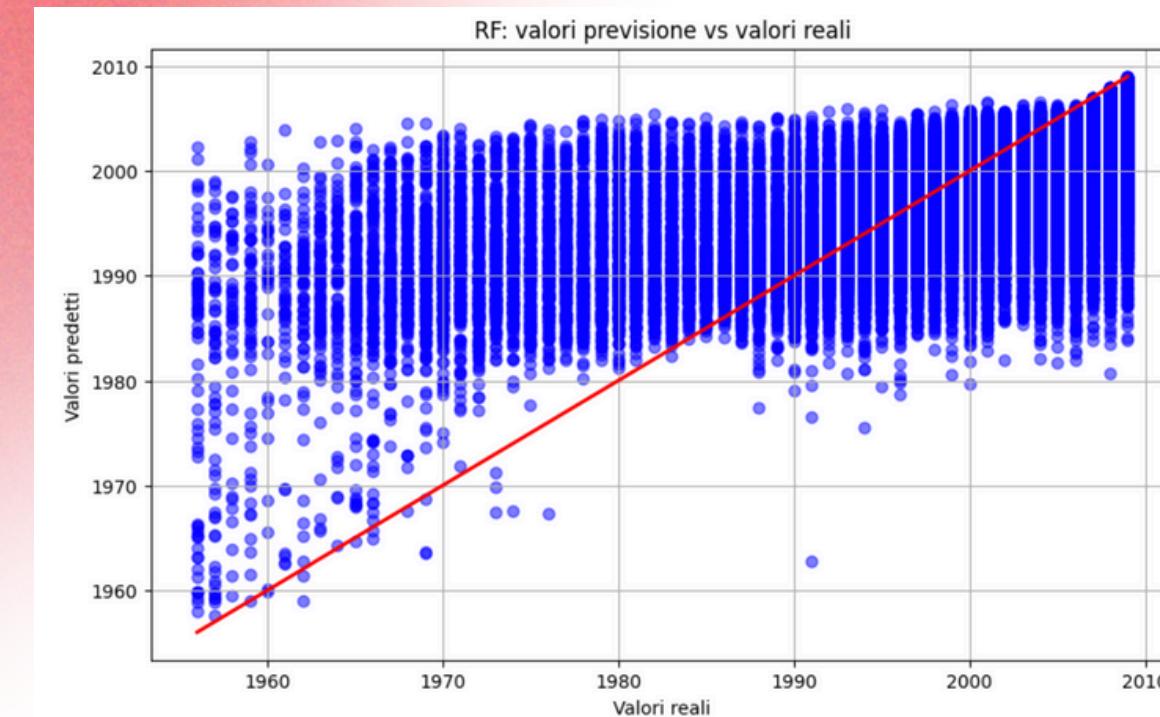
### Linear regression

Metrica	MSE	MAE	MAPE	R2 Score
Valore	79.06	6.51	0.00	0.27



### Random forest

Metrica	MSE	MAE	MAPE	R-squared
Valore	65.3022	5.6318	0.2826%	0.4008

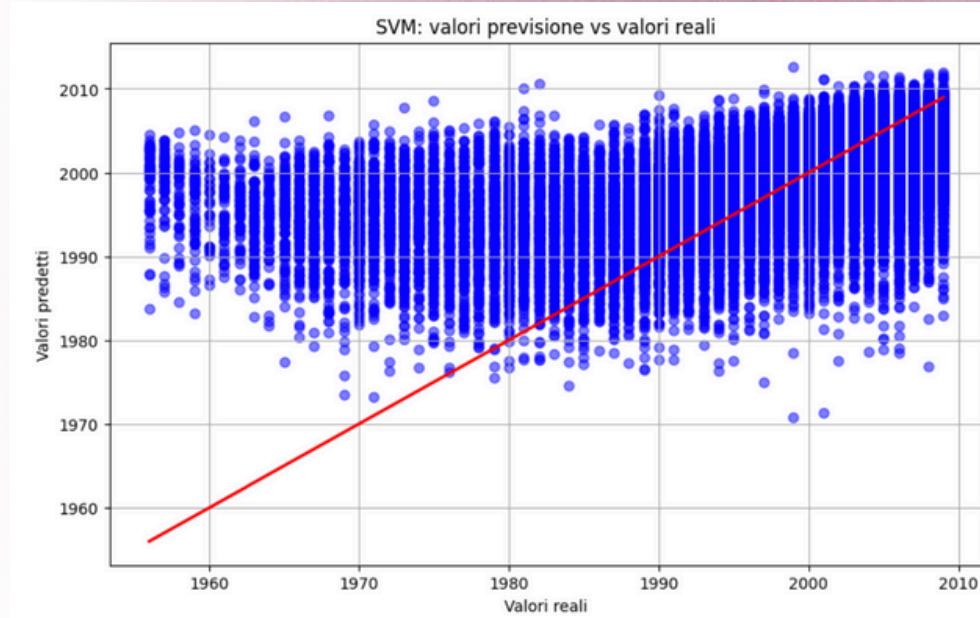


# Risultati

## SVM e KNN

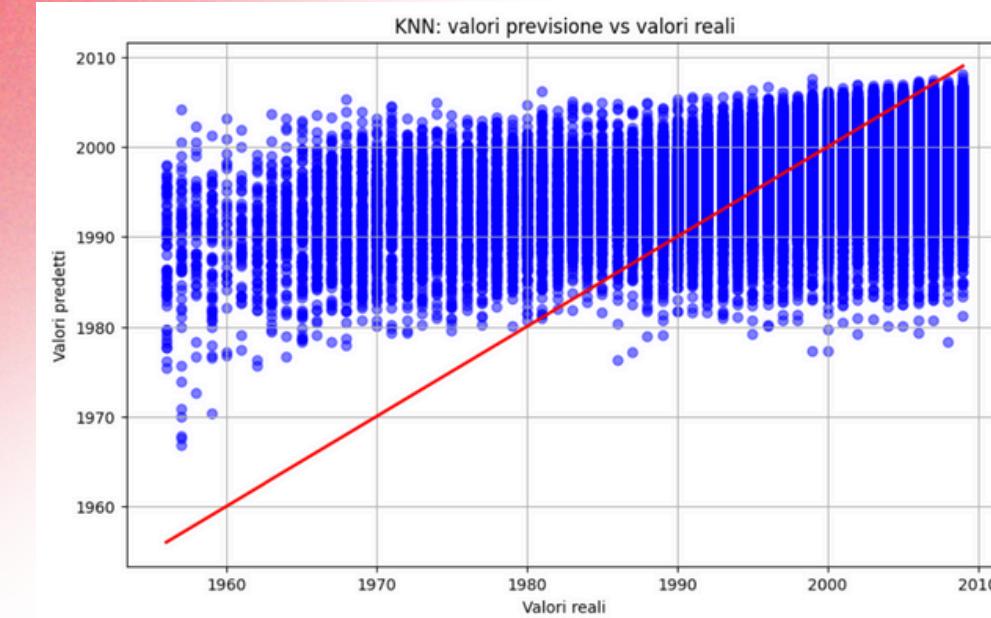
### SVM

Metrica	MSE	MAE	MAPE	R2 Score
Valore	75.1262	5.6833	0.2856	0.3107



### KNN

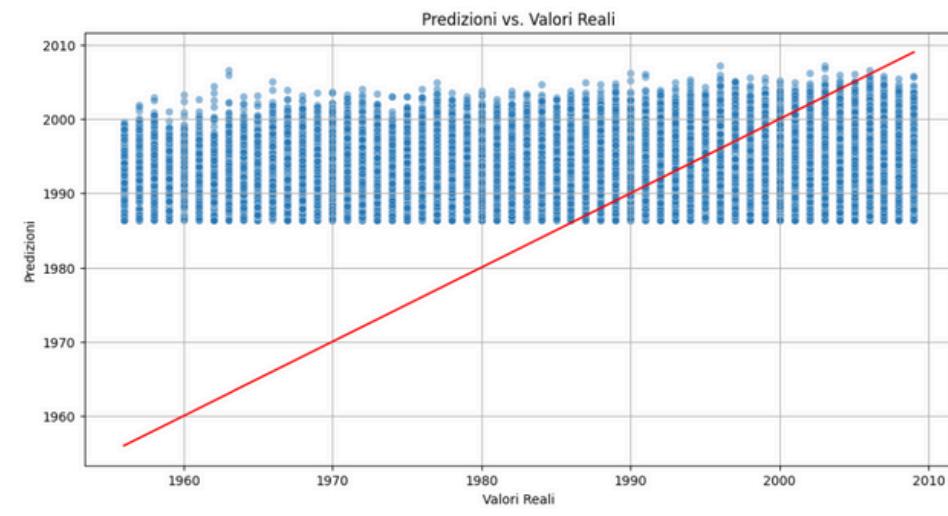
Metrica	MSE	MAE	MAPE	R2 Score
Valore	80.6958	6.7374	0.3379	0.2596



# Risultati

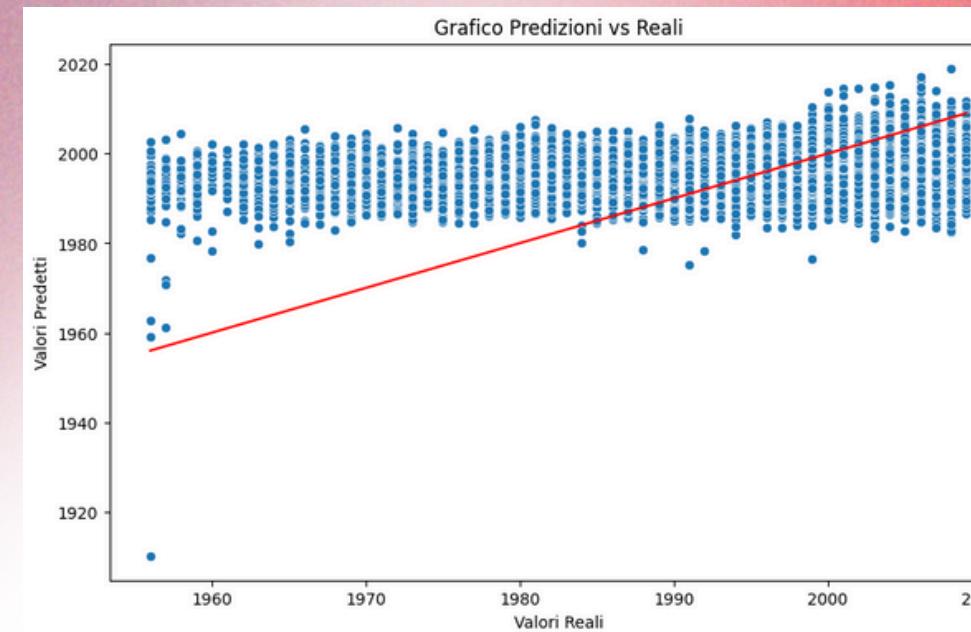
## Rete neurale, TabNet e TabTransformer

Rete Neurale FF



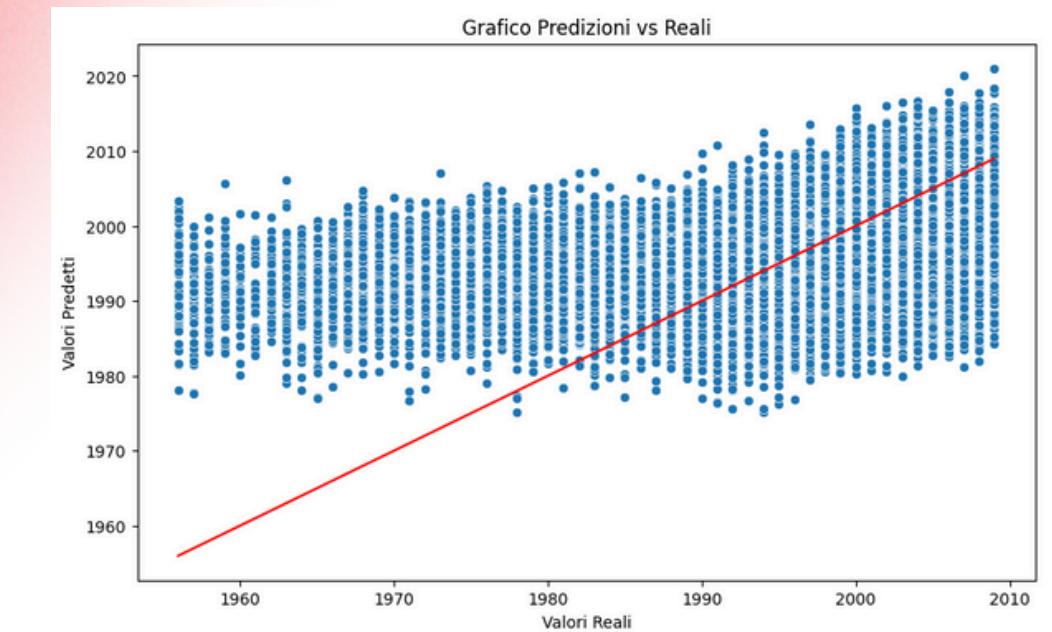
TabNet

Test metric	DataLoader 0
test_loss	97.62861633300781
test_mean_absolute_error	7.508788108825684
test_mean_absolute_percentage_error	0.0037667506840080023
test_mean_squared_error	97.62861633300781
test_r2_score	0.10504426807165146



TabTransformer

Test metric	DataLoader 0
test_loss	79.19596099853516
test_mean_absolute_error	6.5241169929504395
test_mean_absolute_percentage_error	0.0032740216702222824
test_mean_squared_error	79.19596099853516
test_r2_score	0.2634124457836151



**Grazie per l'attenzione**

---