



**UNIVERSITÀ  
DI PARMA**

# **Classificazione di segnali mediante programmazione genetica modulare**

**Relatore**

Prof. Stefano Cagnoni

**Tesi di laurea di**

Arianna Cella

**Correlatore**

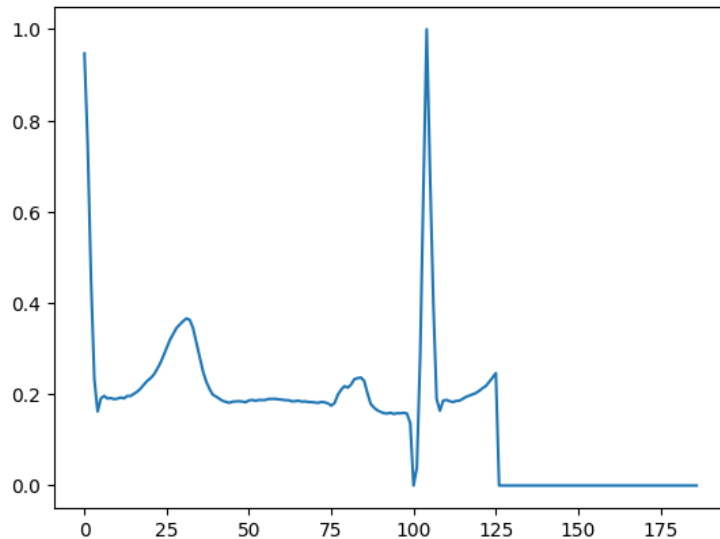
Dott.ssa Giulia Magnani

Anno accademico 2022-2023

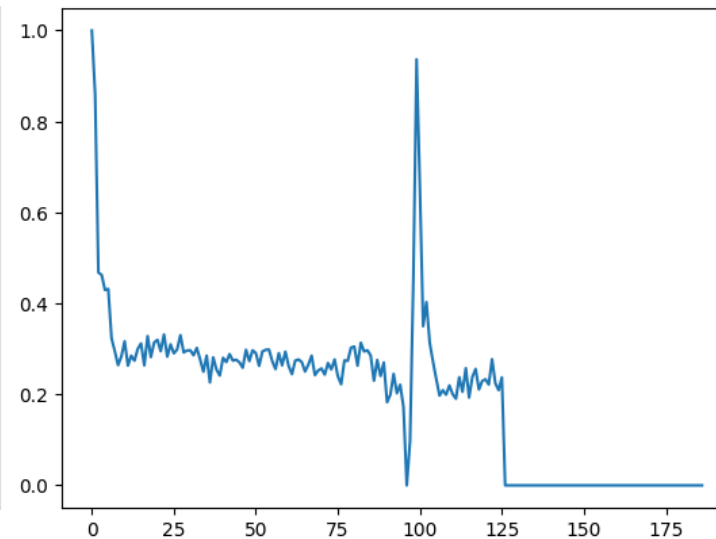
# Introduzione

La **classificazione di segnali** consiste nell'identificare e assegnare una categoria o una classe a ciascun segnale in base alle loro caratteristiche o a pattern distintivi.

## Dataset di segnali ECG

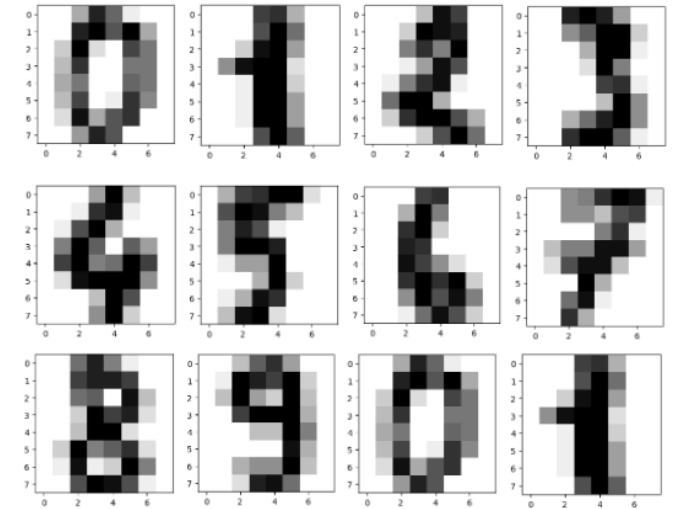


Battito normale (0)



Battito anormale (1)

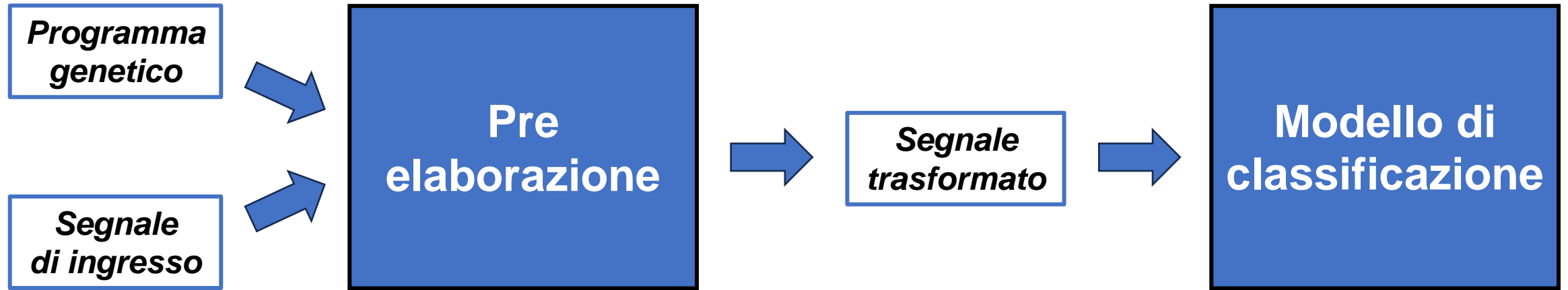
## Dataset Digit



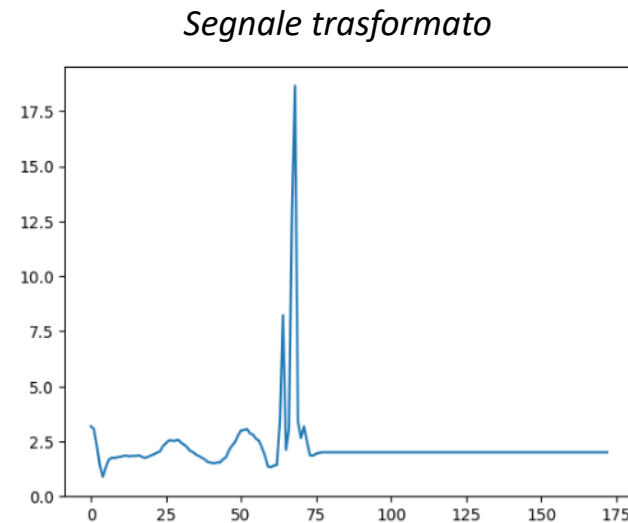
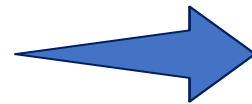
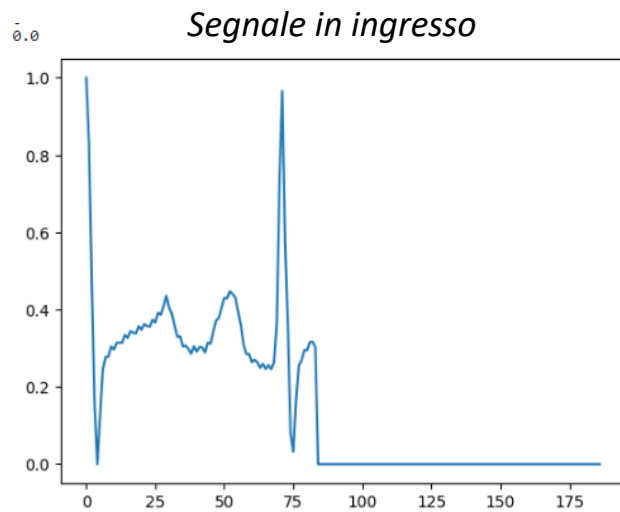
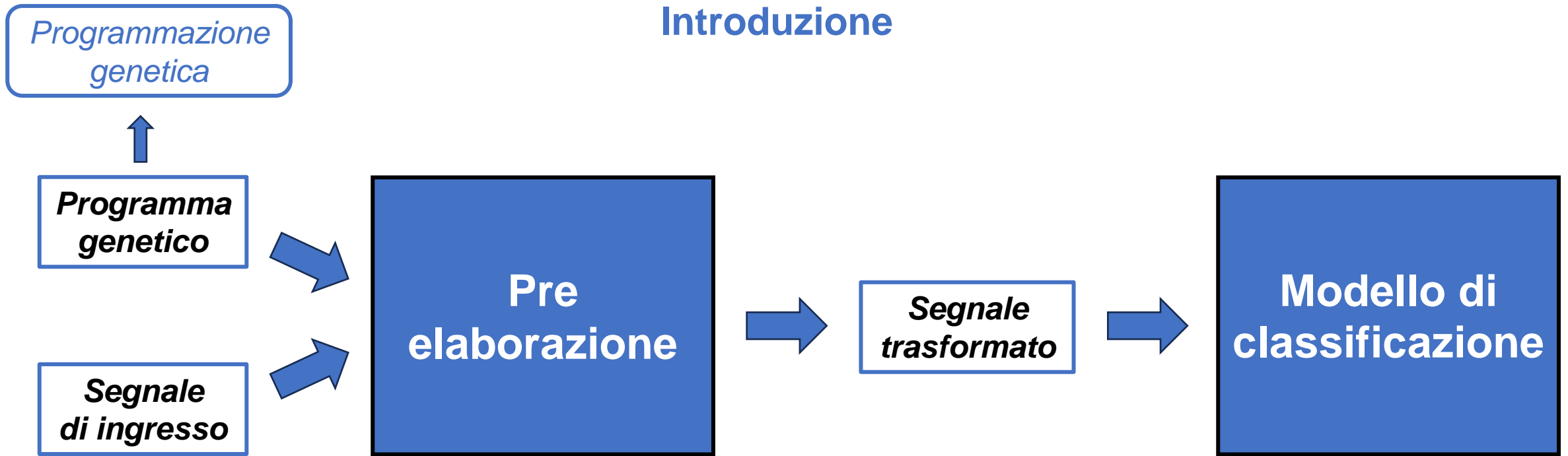
## Obiettivi:

1. Utilizzare la **programmazione genetica (GP)** per effettuare una **preelaborazione dei dataset** al fine di migliorare le prestazioni di un classificatore applicato direttamente al dataset originale.
2. Utilizzo di un **approccio modulare della GP** per effettuare la preelaborazione, in modo da semplificare l'implementazione del sistema su dispositivi FPGA grazie all'utilizzo di moduli che implementano funzioni di alto livello.

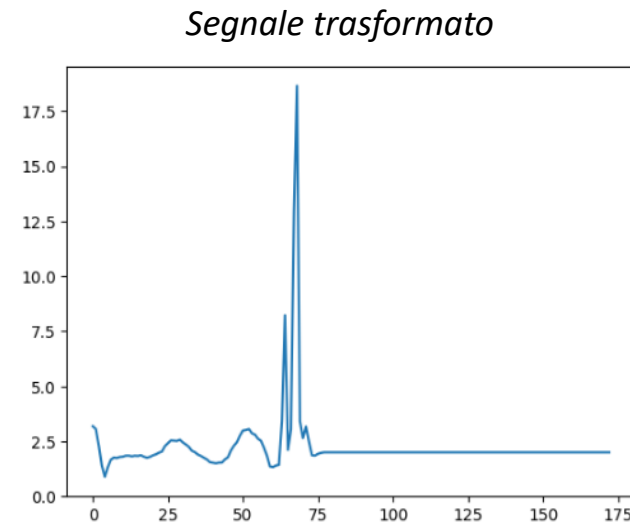
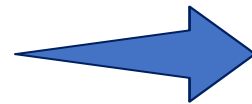
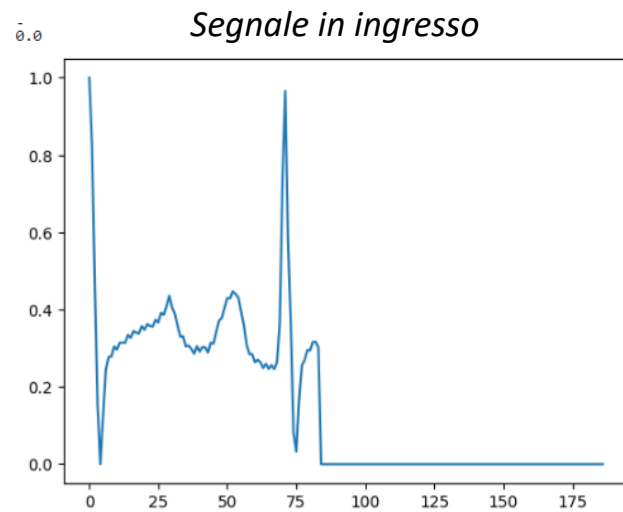
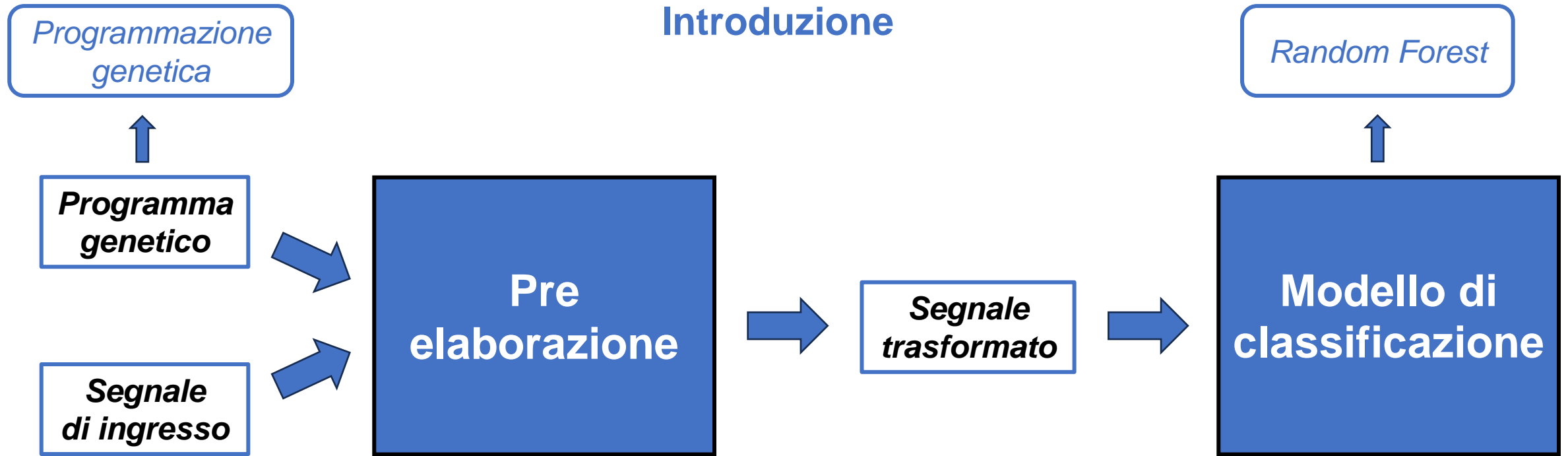
# Introduzione



# Introduzione

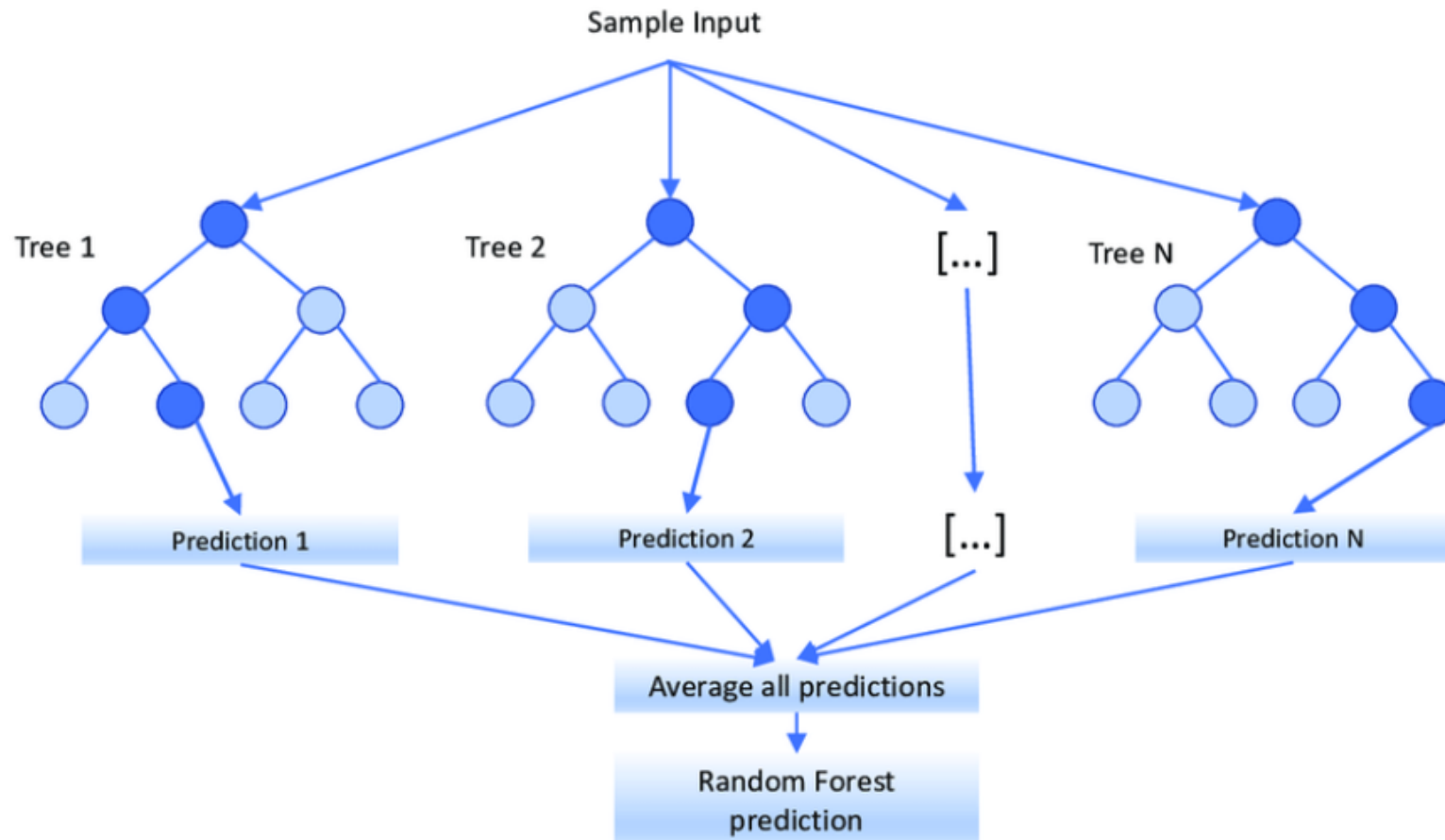


# Introduzione



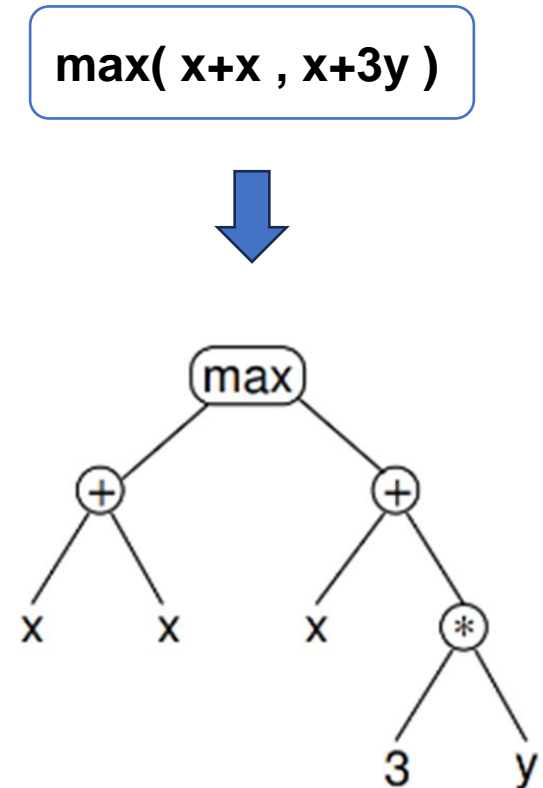
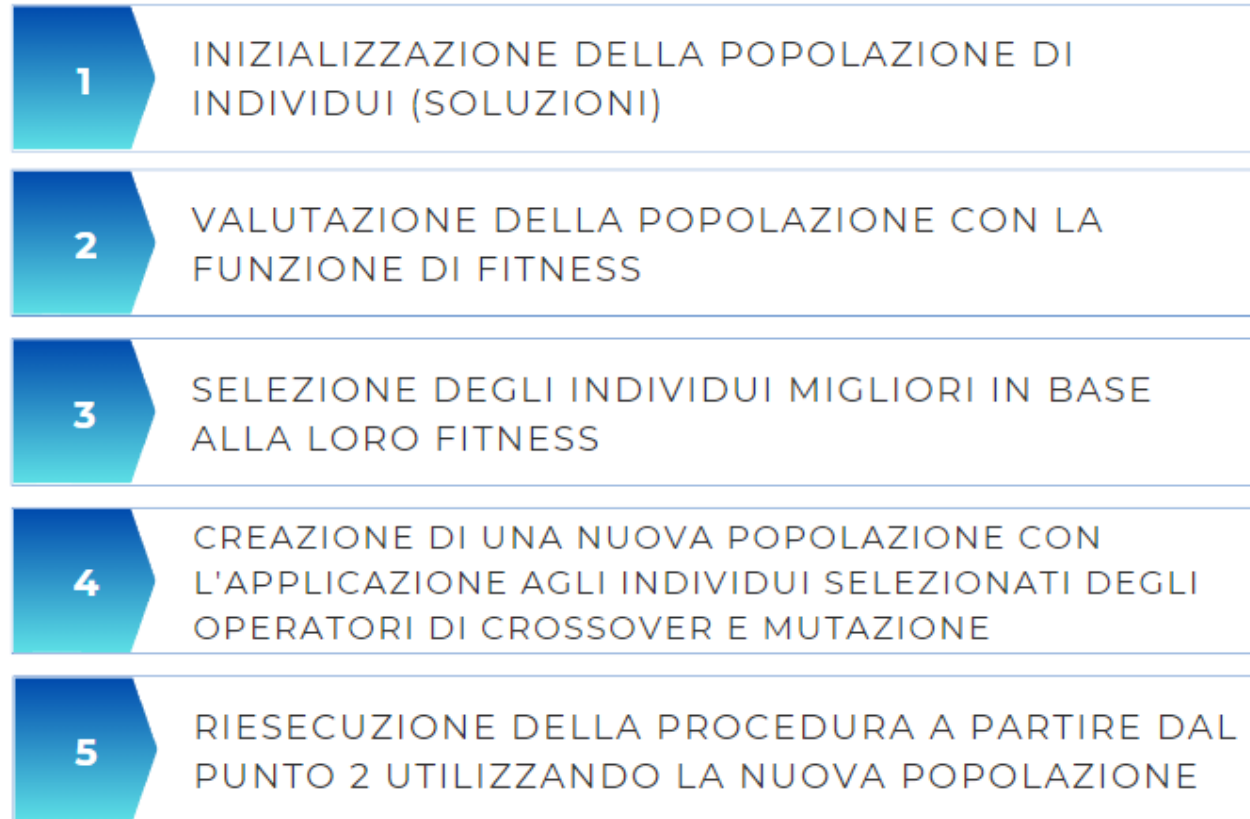
## Cenni teorici

Il **Random Forest** combina diversi alberi decisionali, ognuno dei quali effettua una previsione sulla classe di appartenenza di un'istanza da classificare. La classe che ottiene il maggior numero di voti viene scelta come classe di appartenenza finale.



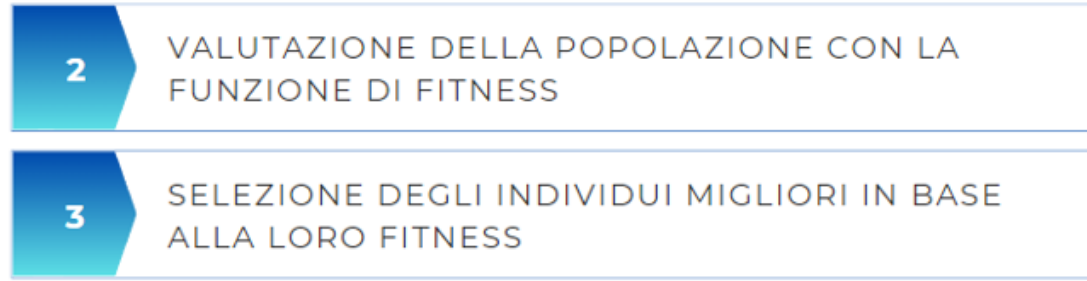
## Cenni teorici

La **programmazione genetica** è un algoritmo evolutivo finalizzato all'evoluzione di funzioni sottoforma di alberi sintattici, che rappresentano la soluzione al problema da ottimizzare.





## Cenni teorici



La selezione degli individui migliori viene effettuata dalla **funzione di fitness** che assegna un punteggio a ciascun individuo in base alla capacità di soddisfare gli obiettivi desiderati.

Nel contesto specifico:

- l'**obiettivo** è trovare una trasformazione del dataset che permetta di ottimizzare le prestazioni del classificatore applicato ai dati trasformati
- la **funzione di fitness** valuta le prestazioni del classificatore con la metrica di valutazione F1 score (fitness dell'individuo).

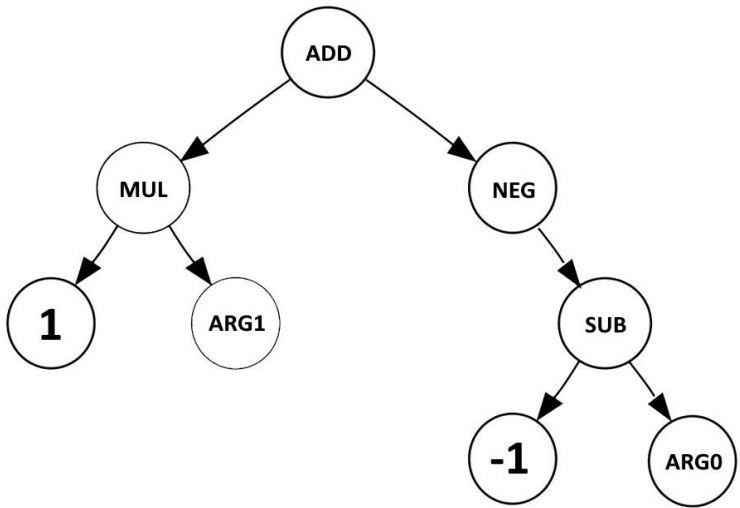
$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

$$Precision = \frac{TP}{TP + FP}$$

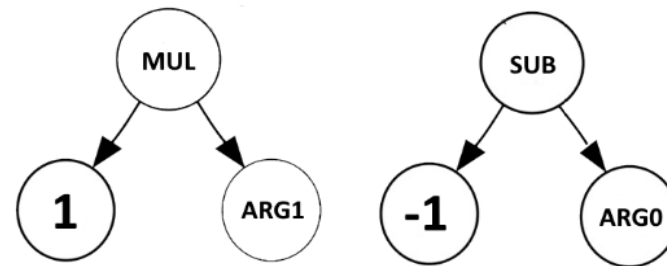
$$Recall = \frac{TP}{TP + FN}$$

## Approccio modulare

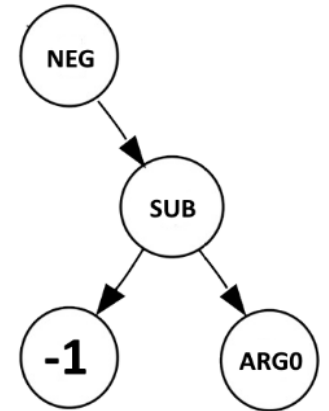
1. **Estrazione dei sottomoduli di profondità 1 e 2** dagli individui della popolazione valutata, assegnando a ciascuno un valore di frequenza e di fitness.
2. **Selezione dei sottomoduli da mantenere nelle iterazioni successive** in base alla loro frequenza e al valore di fitness associato.
3. **Inserimento dei sottomoduli selezionati nel set delle primitive** per utilizzarli nella creazione di nuovi individui all'interno della popolazione.



*Individuo  $\text{add}(\text{mul}(1, \text{ARG1}), \text{neg}(\text{sub}(-1, \text{ARG0})))$*



*Sottomoduli di profondità 1 e 2 che vengono estratti*



## Interfaccia grafica

L'utente può settare i parametri principali dell'algorithm evolutivo e inserire il dataset su cui vuole lavorare.

Inserisci parametri per la run

Numero di run

1

Individui da mantenere

3

Max depth

4

Numero di iterazioni per run

3

Numero di generazioni per run

20

Dimensione kernel

15

Dimensione della popolazione

150

Inserisci dataset già formattato e label target per classificazione binaria:

Carica dataset CSV

Esegui

Algoritmo in esecuzione...

# Risultati

Approcci da confrontare:

1

PRE-ELABORAZIONE CON PROGRAMMAZIONE  
GENETICA MODULARE

2

PRE-ELABORAZIONE CON PROGRAMMAZIONE  
GENETICA SENZA MODULI


3


RANDOM FOREST APPLICATO DIRETTAMENTE  
AL DATASET ORIGINALE


## Risultati su dataset ECG

Confronto dei risultati sul dataset di segnali ECG tra programmazione genetica senza moduli (**GP**), programmazione genetica modulare (**GPM**) e random forest (**RF**) applicato ai dati originali.

Kernel size	Pop size	N° generazioni	N° sottomoduli	F1 test GP	F1 val GP	F1 test GPM	F1 val GPM	F1 test RF	F1 val RF
3	50	20	3	<b>0.8744</b>	0.9411	0.8744	0.941	0.8707	0.895
5	150	20	5	<b>0.892</b>	0.95	0.892	0.95	0.889	0.905
10	50	20	10	<b>0.9142</b>	0.9464	0.9142	0.946	0.885	0.888
15	50	20	3	0.9247	0.924	<b>0.9284</b>	0.937	0.8889	0.888
15	150	20	3	<b>0.903</b>	0.928	0.903	0.928	0.86	0.88
15	50	30	5	<b>0.899</b>	0.919	0.899	0.919	0.889	0.864
20	50	20	3	<b>0.9249</b>	0.9374	0.9249	0.9374	0.899	0.91
25	50	20	5	0.8785	0.9553	<b>0.8892</b>	0.977	0.878	0.906
25	50	20	10	<b>0.896</b>	0.937	0.896	0.937	0.894	0.892
35	50	20	3	0.8677	0.9104	<b>0.8785</b>	0.923	0.8499	0.901
50	50	20	3	0.87	0.885	<b>0.888</b>	0.889	0.882	0.856
55	50	20	10	0.863	0.932	0.863	0.932	<b>0.888</b>	0.883
60	50	20	5	0.877	0.9193	0.877	0.919	<b>0.90</b>	0.87
85	50	20	5	0.907	0.914	<b>0.914</b>	0.923	0.89	0.87


 Esecuzioni in cui la GP modulare migliora i risultati rispetto alla GP standard.


 Esecuzioni in cui GP e GPM ottengono risultati uguali.


 Esecuzioni in cui una preelaborazione con GP o GPM non porta a miglioramenti


## Risultati su dataset Digit

Confronto dei risultati ottenuti sul dataset Digit con programmazione genetica senza moduli (**GP**), programmazione genetica modulare (**GPM**) e random forest (**RF**) applicato ai dati originali.

 Esecuzioni in cui la GP modulare migliora i risultati rispetto alla GP standard.

 Esecuzioni in cui GP e GPM ottengono risultati uguali.

 Esecuzioni in cui una preelaborazione con GP o GPM non porta a miglioramenti

 Esecuzioni in cui una preelaborazione non porta a miglioramenti, ma GPM ha risultati migliori rispetto a GP.

Kernel size	Pop size	N° generazioni	N° sottomoduli	F1 test GP	F1 val GP	F1 test GPM	F1 val GPM	F1 test RF	F1 val RF
5	50	20	10	0.95821	0.98829	<b>0.96026</b>	0.98944	0.95788	0.98074
5	150	40	10	0.95724	0.9930	0.95724	0.9930	<b>0.9701</b>	0.9713
10	50	20	3	0.9578	0.98379	0.9578	0.98379	<b>0.9633</b>	0.9802
10	150	30	5	<b>0.98356</b>	0.99676	0.98356	0.99676	0.9722	0.9779
15	150	20	3	<b>0.95718</b>	0.98621	0.95718	0.98621	0.9521	0.9633
15	50	20	5	<b>0.9736</b>	0.98212	0.9736	0.98212	0.97	0.9773
15	50	20	10	0.9551	0.97638	<b>0.9602</b>	0.97966	0.9655	0.9755
15	50	20	3	0.96587	0.97095	<b>0.97735</b>	0.97115	0.9718	0.96576
20	50	20	5	0.96895	0.96968	0.96895	0.96968	<b>0.973</b>	0.9719
25	300	50	5	<b>0.97206</b>	0.9851	0.97206	0.9851	0.968	0.972

# Conclusioni

Possiamo concludere che:

```
graph TD; A[Possiamo concludere che:] --> B[Una pre-elaborazione del dataset usando la GP migliora i risultati rispetto ad un'applicazione del classificatore Random Forest sul dataset originale.]; A --> C[La GP modulare produce spesso miglioramenti rispetto alla GP standard, anche se non è stata sviluppata con tale obiettivo.];
```

Una pre-elaborazione del dataset usando la GP migliora i risultati rispetto ad un'applicazione del classificatore Random Forest sul dataset originale.

La GP modulare produce spesso miglioramenti rispetto alla GP standard, anche se non è stata sviluppata con tale obiettivo.



**UNIVERSITÀ  
DI PARMA**

**Grazie  
per l'attenzione!**