Università degli studi di Bari Aldo Moro

SISTEMA DI MONITORAGGIO PER LA RILEVAZIONE DI UN ATTACCO CARDIACO

Gruppo di lavoro: Barile Rossana

Matricola: 738058

Email: r.barile@studenti.uniba.it

URL: https://github.com/Rbarile19/ICON/tree/main

INDICE

INTRODUZIONE

Il progetto si focalizza sulla diagnosi degli attacchi di cuore, un tema importante nella medicina moderna. Un attacco di cuore, o infarto miocardico, si verifica quando il flusso sanguigno verso una parte del cuore è bloccato, spesso a causa di un coagulo nelle arterie coronarie. Questa interruzione può portare alla morte delle cellule cardiache per mancanza di ossigeno, rendendo essenziale una diagnosi tempestiva e accurata.

Per affrontare questo problema, utilizzeremo una varietà di algoritmi di apprendimento automatico, tra cui K-Nearest Neighbors (KNN), Naive Bayes, Decision Tree, Random Forest, Neural Network, XGBoost e Support Vector Machine. Questi algoritmi permetteranno di analizzare i dati clinici e i fattori di rischio associati, contribuendo a identificare rapidamente i pazienti a rischio di attacco di cuore. Inoltre, implementeremo tecniche di visualizzazione per rappresentare graficamente i risultati delle diagnosi, facilitando una comprensione chiara e immediata delle informazioni ottenute.

Questo approccio integrato non solo esplorerà diverse metodologie di analisi, ma valuterà anche l'efficacia di ciascun algoritmo nel fornire diagnosi affidabili, contribuendo così a migliorare le pratiche cliniche e a salvare vite umane.

ELENCO ARGOMENTI DI INTERESSE

Creazione di una base di conoscenza:

- Sviluppo di una base di dati per analizzare le informazioni presenti nel dataset.
- Obiettivo di identificare il rischio di attacco cardiaco nei pazienti.

Utilizzo di query:

- Esecuzione di specifiche query per valutare i parametri clinici dei pazienti.

Algoritmi di apprendimento supervisionato impiegati:

- K-Nearest Neighbors (KNN): utilizzato per la classificazione basata sulla vicinanza dei dati.
- Naive Bayes: algoritmo probabilistico per la classificazione dei pazienti.
- Decision Tree: modello di classificazione che utilizza un approccio a struttura ad albero per le decisioni.

- Random Forest: combinazione di più alberi decisionali per migliorare l'accuratezza delle previsioni.
- XGBoost: algoritmo di boosting estremamente efficiente per la classificazione e la regressione.
- Neural Network: modello ispirato al cervello umano, utilizzato per riconoscere schemi complessi nei dati.
- Support Vector Machine (SVM): algoritmo di classificazione che cerca di trovare l'iperpiano ottimale per separare le classi.

ANALISI DEL PROBLEMA

l sistema di diagnosi di un attacco di cuore è un problema di ottimizzazione che si occupa di determinare il percorso migliore per identificare e trattare i sintomi di un attacco cardiaco. In questo contesto, si può considerare un insieme di indicatori di rischio e segni clinici, con lo scopo di trovare il metodo diagnostico più efficace per valutare la condizione del paziente.

DATI UTILIZZATI

Prima di iniziare a lavorare sul sistema di diagnosi per gli attacchi di cuore, è stato fondamentale condurre un'analisi approfondita dei dati presenti nel dataset, dal nome "cuore", (preso dal seguente sito https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset). È importante comprendere alcuni aspetti chiave del dataset prima di proseguire con l'analisi. Questo è composto dalle seguenti variabili:

- age: Età del paziente.
- sex: Genere del paziente (1 = maschio, 0 = femmina).
- cp: Tipo di dolore toracico (0 = angina tipica, 1 = angina atipica, 2 = dolore non correlato all'angina, 3 = asintomatico).
- TRTBPS: Pressione sanguigna a riposo (in mmHg).
- CHOL: Livello di colesterolo in mg/dl, misurato tramite un sensore BMI.
- fbs: Zucchero nel sangue a digiuno (> 120 mg/dl) (1 = vero, 0 = falso).

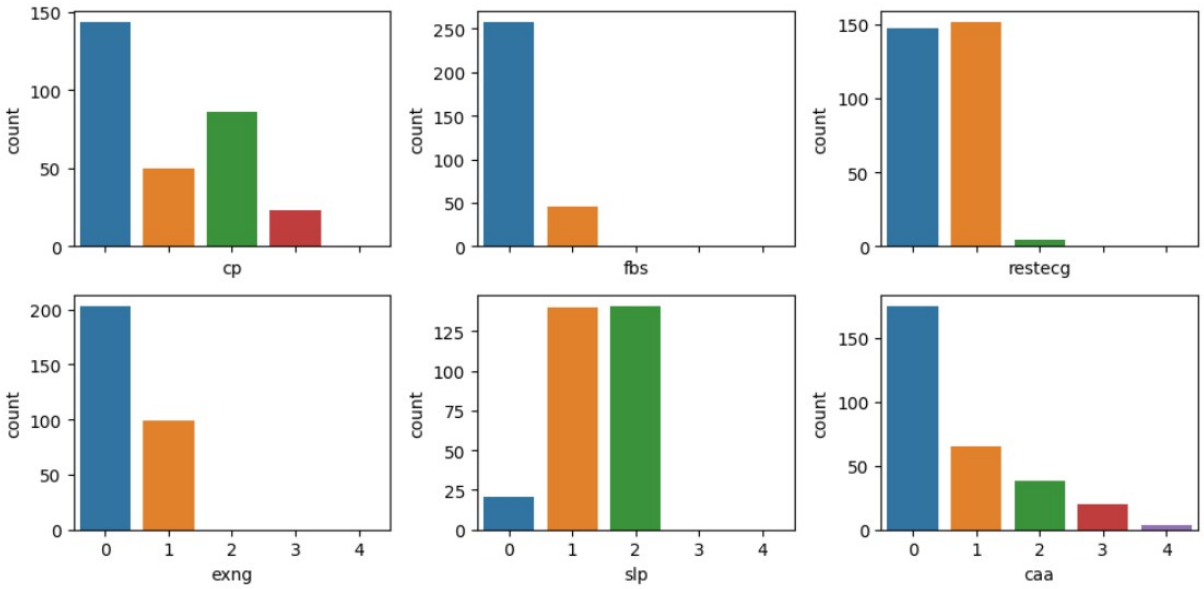
- restecg: Risultati dell'elettrocardiogramma a riposo (0 = normale, 1 = anomalia dell'onda ST-T, 2 = ipertrofia del ventricolo sinistro).
- thalachh: Frequenza cardiaca massima raggiunta.
- exng: Angina pectoris indotta dall'esercizio (1 = sì, 0 = no).
- old peak: Picco precedente.
- slp: Pendenza dell'elettrocardiogramma.
- caa: Numero di grandi vasi.
- thall: Risultato del test Thallium (0.3).
- output: Variabile target.

Queste informazioni sono essenziali per una corretta interpretazione e analisi dei dati al fine di sviluppare un sistema diagnostico efficace.

STRUMENTI AGGIUNTIVI

Questo codice permette di visualizzare la distribuzione di sei variabili categoriali relative a caratteristiche cliniche presenti nel dataset. Utilizzando i countplot, è possibile osservare rapidamente la frequenza di ciascuna categoria per ogni variabile, ad esempio il numero di persone con un certo tipo di dolore toracico o con glicemia alta.

L'inserimento di questi grafici risulta particolarmente utile nell'analisi esplorativa dei dati (EDA), poiché consente di comprendere meglio come sono distribuiti i valori delle variabili categoriali. Queste informazioni possono rivelarsi preziose per individuare eventuali squilibri nelle classi o tendenze nei dati, elementi che potrebbero influenzare sia l'accuratezza di un modello di machine learning, sia l'interpretazione dei risultati.



KNOWLEDGE BASE O BASE DI CONOSCENZA

Una base di conoscenza è un costrutto fondamentale per la rappresentazione della conoscenza in un formato formale e logico. Essa rappresenta un insieme di informazioni che possono essere interrogate, manipolate e inferite, costituendo la spina dorsale dei sistemi di intelligenza artificiale e dei programmi di ragionamento automatico. La sua struttura è composta principalmente da fatti e regole, che lavorano insieme per consentire il ragionamento e la deduzione di nuove informazioni.

COMPOSIZIONE DELLA BASE DI CONOSCENZA

I fatti sono le unità di informazione di base all'interno della base di conoscenza. Rappresentano affermazioni concrete o verità accettate riguardo il dominio in questione. I fatti non necessitano di ulteriori prove per essere considerati veri; la loro validità è assunta come predefinita. Queste affermazioni possono riguardare entità, relazioni o proprietà specifiche.

Le regole, d'altra parte, sono affermazioni condizionali che stabiliscono relazioni tra i fatti esistenti nella base di conoscenza. Esse definiscono come alcune conclusioni possano essere dedotte da altre affermazioni. Una regola è composta da una parte antecedente, nota come "corpo", che specifica le condizioni necessarie affinché la conclusione, o "testa", possa essere considerata vera. Le regole consentono di formulare ragionamenti

più complessi, trasformando la base di conoscenza in un sistema in grado di fornire nuove informazioni a partire da quelle già presenti.

FUNZIONAMENTO DELLA BASE DI CONOSCENZA

Il funzionamento di una base di conoscenza in Prolog è caratterizzato da diversi processi logici che permettono l'interazione con l'utente e la manipolazione delle informazioni. Gli utenti possono interagire con la base di conoscenza ponendo query o domande. Queste query possono essere formulate per verificare l'esistenza di un certo fatto, esplorare le relazioni tra diverse entità o inferire informazioni non esplicitamente presenti nella base di conoscenza. Prolog elabora queste query e restituisce risposte basate sulla conoscenza codificata.

In situazioni in cui non è disponibile una risposta diretta a una query, Prolog utilizza il processo di inferenza per dedurre nuove informazioni. Questo processo si basa sull'applicazione delle regole alla luce dei fatti esistenti. Se una regola è soddisfatta, Prolog può inferire conclusioni aggiuntive, estendendo così la propria base di conoscenza e rispondendo a query più complesse.

Un altro aspetto fondamentale del funzionamento di Prolog è il backtracking. Questo meccanismo consente al sistema di esplorare diverse possibilità di soluzione per una query. Se Prolog trova che una possibile via non porta a una conclusione valida, è in grado di tornare indietro e provare un'altra opzione, garantendo così una ricerca esaustiva delle risposte possibili. Il backtracking rende Prolog particolarmente potente nella gestione di problemi complessi e nella ricerca di soluzioni ottimali.

Prolog utilizza il concetto di unificazione, un processo che consente di confrontare e combinare termini all'interno delle query e delle regole. La unificazione è essenziale per stabilire corrispondenze tra variabili e valori specifici, permettendo a Prolog di adattare le regole e i fatti in base ai dati forniti dall'utente. Questo processo è alla base della capacità di Prolog di inferire nuove conoscenze e di rispondere in modo pertinente alle domande poste.

La base di conoscenza in è un costrutto essenziale che consente di rappresentare, organizzare e inferire informazioni in modo logico e coerente. Attraverso l'uso di fatti e regole, Prolog si configura come uno strumento potente per il ragionamento automatico,

in grado di gestire la complessità delle relazioni all'interno di un dominio specifico. La sua capacità di inferenza, backtracking e unificazione rende Prolog un linguaggio di programmazione particolarmente adatto per applicazioni in intelligenza artificiale, dove la rappresentazione della conoscenza e il ragionamento logico sono fondamentali.

DECISIONI DI PROGETTO

Nel progetto sono state formulate delle normative per la creazione della base di conoscenza. Grazie a queste normative, l'utente ha la possibilità di elaborare query per interrogare il sistema. Di seguito, sono presentate e illustrate in dettaglio le regole redatte in Prolog:

REGOLA 1

```
% Regola 1: Identificare se un paziente è a rischio di attacco cardiaco
paziente_a_rischio(Eta, Colesterolo, AnginaEsercizio, TipoDolore) :-
    Eta > 50,
    Colesterolo > 240,
    AnginaEsercizio = 1,
    TipoDolore > 1.
```

Questa regola Prolog ha lo scopo di determinare se un paziente presenta un rischio elevato di attacco cardiaco sulla base di specifiche variabili cliniche, incluse età, livelli di colesterolo, angina durante l'esercizio fisico e tipo di dolore. L'approccio logico consente un'elaborazione rapida e automatizzata delle informazioni cliniche.

La regola utilizza una forma di logica proposizionale per valutare se tutte le condizioni specificate sono vere. Se tutte le condizioni sono soddisfatte, il predicato “paziente_a_rischio” restituisce “true”, indicando che il paziente è considerato a rischio di attacco cardiaco. Questo approccio permette una facile estensione e modifica della logica, facilitando l'adattamento alle linee guida cliniche emergenti o alle nuove evidenze scientifiche.

- Predicato: paziente_a_rischio. Il predicato accetta quattro parametri di input:
    - Età: rappresenta l'età del paziente (numerico).
    - Colesterolo: rappresenta il livello di colesterolo nel sangue (numerico).

- AnginaEsercizio: rappresenta un flag binario (1 o 0) per indicare la presenza di angina durante l'esercizio.
- TipoDolore: un valore numerico che classifica l'intensità del dolore.

- Condizioni di Attivazione:
  - Eta > 50: Questa condizione verifica che l'età del paziente sia superiore a 50 anni.
  - Colesterolo > 240: Questa condizione verifica che il livello di colesterolo sia superiore a 240 mg/dL.
  - AnginaEsercizio = 1: Questa condizione verifica che il paziente abbia riportato angina durante l'esercizio fisico.
  - TipoDolore > 1: Questa condizione verifica che il tipo di dolore avvertito dal paziente sia di intensità moderata o elevata.

REGOLA 2

```
% Regola 2: Tipo di attacco basato su pressione e picco ST
tipo_attacco(Eta, Colesterolo, AnginaEsercizio, TipoDolore, Pressione, PiccoPrecedente, alto) :-
    paziente_a_rischio(Eta, Colesterolo, AnginaEsercizio, TipoDolore),
    Pressione > 160,
    PiccoPrecedente > 2.5.

tipo_attacco(Eta, Colesterolo, AnginaEsercizio, TipoDolore, Pressione, PiccoPrecedente, medio) :-
    paziente_a_rischio(Eta, Colesterolo, AnginaEsercizio, TipoDolore),
    Pressione > 140, Pressione =< 160,
    PiccoPrecedente =< 2.5.

tipo_attacco(Eta, Colesterolo, AnginaEsercizio, TipoDolore, Pressione, PiccoPrecedente, basso) :-
    paziente_a_rischio(Eta, Colesterolo, AnginaEsercizio, TipoDolore),
    Pressione =< 140.
```

La prima regola classifica un paziente come ad alto rischio di attacco cardiaco se soddisfa determinate condizioni. Innanzitutto, il paziente deve essere identificato come a rischio, attraverso la chiamata al predicato “paziente_a_rischio”. Inoltre, la pressione sanguigna deve essere superiore a 160 mmHg e il valore di picco ST precedente deve superare 2.5. L'inclusione di soglie specifiche per pressione sanguigna e picco ST consente di identificare pazienti con un rischio elevato in modo preciso, basandosi su dati clinici oggettivi.

- Condizioni: Questa regola classifica un paziente come ad alto rischio di attacco cardiaco se soddisfa le seguenti condizioni:
    1. Il paziente è identificato come a rischio (paziente_a_rischio).
    2. La pressione sanguigna è superiore a 160 mmHg.
    3. Il valore di picco ST precedente è superiore a 2.5.

L'inclusione di soglie specifiche per pressione sanguigna e picco ST consente di identificare pazienti con un rischio elevato in modo preciso, basandosi su dati clinici oggettivi.

La seconda regola classifica un paziente come a rischio medio, sempre previa identificazione come paziente a rischio. In questo caso, la pressione sanguigna deve essere compresa tra 140 mmHg e 160 mmHg (inclusi), e il valore di picco ST precedente deve essere minore o uguale a 2.5. La definizione di un range di pressione sanguigna specifico e un limite per il picco ST permette una categorizzazione più raffinata, garantendo una risposta adeguata per i pazienti non gravemente compromessi.

- Condizioni: questa regola classifica un paziente come a rischio medio se soddisfa le seguenti condizioni:
    1. Il paziente è identificato come a rischio (paziente_a_rischio).
    2. La pressione sanguigna è compresa tra 140 mmHg e 160 mmHg (inclusi).
    3. Il valore di picco ST precedente è minore o uguale a 2.5.

La definizione di un range di pressione sanguigna specifico e un limite per il picco ST permette una categorizzazione più raffinata, garantendo una risposta adeguata per i pazienti non gravemente compromessi.

La terza regola classifica un paziente come a rischio basso, richiedendo che il paziente sia identificato come a rischio. Inoltre, la pressione sanguigna deve essere minore o uguale a 140 mmHg. Stabilendo una soglia di pressione sanguigna ben definita, questa regola fornisce un chiaro segnale per identificare pazienti che, sebbene a rischio, presentano valori clinici che indicano una minore probabilità di attacco cardiaco imminente.

- Condizioni: Questa regola classifica un paziente come a rischio basso se soddisfa le seguenti condizioni:
    1. Il paziente è identificato come a rischio (paziente_a_rischio).
    2. La pressione sanguigna è minore o uguale a 140 mmHg.

Stabilendo una soglia di pressione sanguigna ben definita, questa regola fornisce un chiaro segnale per identificare pazienti che, sebbene a rischio, presentano valori clinici che indicano una minore probabilità di attacco cardiaco imminente.

REGOLA 3

```
% Regola 3: Età media dei pazienti
eta_media(ListaEta, EtaMedia) :-
    sumlist(ListaEta, Somma),
    length(ListaEta, NumeroPazienti),
    EtaMedia is Somma / NumeroPazienti.
```

La regola Prolog “eta_media” calcola l'età media dei pazienti a partire dalla lista dell’età presente nel dataset. Accetta due argomenti: “ListaEta”, che è una lista di numeri rappresentanti le età dei pazienti, e “EtaMedia”, un numero reale che rappresenta l'età media calcolata.

Utilizza il predicato built-in “sumList” per calcolare la somma di tutti gli elementi nella lista “ListaEta”, assegnando il risultato alla variabile “Somma”. Usa il predicato built-in “lenght” per determinare il numero di elementi nella lista “ListaEta”, assegnando il risultato alla variabile “NumeroPazienti”. Successivamente calcola l'età media dividendo la somma delle età per il numero di pazienti e assegna il risultato alla variabile “EtaMedia” utilizzando l'operatore “is”.

REGOLA 4

```
% Regola 4: Determinare se un paziente può avere un attacco cardiaco
puo_avere_attacco_cardiaco_prob(Eta, TipoDolore, AnginaEsercizio, Pendenza,
                                NumeroVasi, RisultatoThallium, FrequenzaCardiaca,
                                PiccoPrecedente, PuoAvereAttacco) :-
    (
        (TipoDolore == 0 -> ValoreCondizione1 = 0; ValoreCondizione1 = 1),
        (AnginaEsercizio == 1 -> ValoreCondizione2 = 0; ValoreCondizione2 = 1),
        ((Pendenza == 0; Pendenza == 1) -> ValoreCondizione3 = 0; ValoreCondizione3 = 1),
        (NumeroVasi =< 3 -> ValoreCondizione4 = 0; ValoreCondizione4 = 1),
        (RisultatoThallium == 3 -> ValoreCondizione5 = 0; ValoreCondizione5 = 1),
        ((Eta > 50; Eta < 30) -> ValoreCondizione6 = 0; ValoreCondizione6 = 1),
        (FrequenzaCardiaca < 130 -> ValoreCondizione7 = 0; ValoreCondizione7 = 1),
        (PiccoPrecedente > 2.0 -> ValoreCondizione8 = 0; ValoreCondizione8 = 1)
    ),
    SommaCondizioni is ValoreCondizione1 + ValoreCondizione2 + ValoreCondizione3 +
    ValoreCondizione4 + ValoreCondizione5 + ValoreCondizione6 + ValoreCondizione7 +
    ValoreCondizione8,
    (SommaCondizioni > 4 -> PuoAvereAttacco = si; PuoAvereAttacco = no).
```

Questa regola Prolog ha lo scopo di determinare se un paziente può essere a rischio di attacco cardiaco, basandosi su una serie di parametri clinici. La funzione riceve in ingresso diversi valori relativi alla condizione fisica del paziente, come l'età, il tipo di dolore toracico, la presenza di angina durante l'esercizio fisico, il numero di vasi sanguigni interessati, il risultato di un esame con tallio, la frequenza cardiaca massima e la pressione sanguigna in un determinato momento.

Il comportamento della funzione si basa sull'applicazione di otto condizioni che controllano ciascuna una specifica caratteristica del paziente. Per ogni condizione, viene effettuato un controllo logico che determina se il parametro corrispondente suggerisce un rischio maggiore o minore di attacco cardiaco. Se una determinata caratteristica si ritiene non influente o riduce il rischio, viene assegnato il valore 0 a una variabile temporanea, mentre se aumenta il rischio, viene assegnato il valore 1.

Ogni controllo segue una struttura semplice di tipo "if-then-else". Per esempio, se il tipo di dolore al petto è uguale a 0 (il che potrebbe indicare assenza di dolore), la variabile associata a questa condizione assume il valore 0; altrimenti, assume il valore 1. Lo stesso vale per le altre condizioni: l'angina durante l'esercizio fisico, la pendenza del tratto ST nell'ECG, il numero di vasi sanguigni e così via. Ogni parametro è trattato individualmente, e alla fine tutti i valori temporanei ottenuti vengono sommati.

Il risultato di questa somma rappresenta un indice di rischio. Se il totale delle condizioni che indicano rischio è superiore a 4, allora la funzione conclude che il paziente è a rischio

di avere un attacco cardiaco, restituendo come risultato il valore "sì". Se, invece, la somma è 4 o inferiore, la funzione restituisce "no", indicando che il paziente ha un rischio basso.

REGOLA 5

```
% Regola 5: Condizione cardiovascolare grave
condizione_grave(Colesterolo, NumeroVasi, FrequenzaCardiaca) :-
    Colesterolo > 300,
    NumeroVasi >= 2,
    FrequenzaCardiaca < 100.
```

Questa regola Prolog che descrive una condizione cardiovascolare grave è progettata per identificare situazioni di rischio sulla base di tre parametri clinici: il livello di colesterolo, il numero di vasi sanguigni interessati e la frequenza cardiaca. Questa regola funziona come un filtro logico, verificando se i valori forniti per questi parametri superano o rientrano in soglie specifiche.

Nello specifico, la regola stabilisce che una condizione è considerata grave quando:

1. Il colesterolo supera i 300 mg/dL, indicando un livello particolarmente elevato di colesterolo nel sangue.
2. Sono coinvolti almeno due vasi sanguigni (indicati dal parametro NumeroVasi), il che suggerisce un'estensione significativa del problema cardiovascolare.
3. La frequenza cardiaca è inferiore a 100 battiti per minuto, che può riflettere una ridotta risposta del cuore sotto sforzo o in situazioni di rischio.

Quando questi tre criteri sono soddisfatti contemporaneamente, il sistema Prolog considera vera la regola e conclude che il paziente si trova in una condizione cardiovascolare grave.

Questa logica permette al sistema di classificare automaticamente le condizioni cardiovascolari, fornendo un meccanismo utile per evidenziare i casi di rischio elevato. In un contesto medico, potrebbe essere integrata in un sistema di supporto alle decisioni per agevolare la valutazione dei pazienti sulla base di dati clinici essenziali.

APPRENDIMENTO SUPERVISIONATO

L'apprendimento supervisionato è una tecnica di machine learning in cui un algortimo viene addestrato su dati etichettati, cioè su input associati a output noti. L'obiettivo è che il modello impari a mappare correttamente gli input agli output, così da poter fare previsioni su nuovi dati non visti. Il processo di divide in due fasi: addestramento, dove il modello impara dai dati, e test, dove viene valutata la sua capacità di generalizzare. Le sfide principali includono evitare l'verfitting e scegliere modelli e caratteristiche adeguate per ottenere buone prestazioni su dati nuovi.

SOMMARIO

Il progetto si concentra sulla diagnosi di possibili problemi cardiaci di un paziente, utilizzando come variabile di riferimento quella dell'output, che rappresenta il target da predire a partire dalle altre caratteristiche del paziente.

I modelli che sono stati scelti e valutati sono i seguenti: K-Nearest Neighbors (KNN), Random Forest, Decision Tree, Naive Bayes, Neural Netwok, XGBoost e Support Vector Machine utilizzando per tutti Scikit-learn.

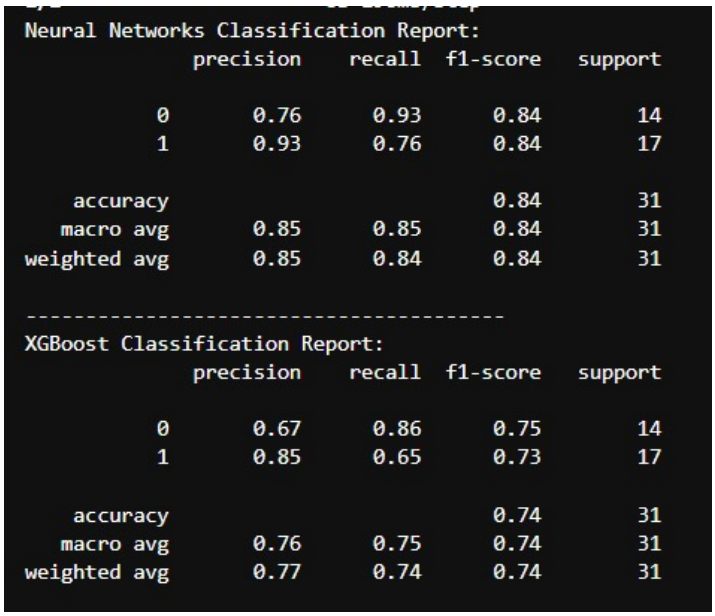
DECISIONI DI PROGETTO

Inizialmente, i diversi modelli sono stati testati senza applicazione di parametri specifici, con l'obiettivo di ottenere una panoramica preliminare dei loro comportamenti. Questa fase esplorativa ci ha permesso di valutare le prestazioni allo stato iniziale dei modelli, fornendo una base di confronto per eventuale miglioramenti futuri. I risultati ottenuti in questi primo step sono i seguenti.

```
=== Report Complessivo ===
Modello: KNN
  - Accuracy: 0.8710
  - F2 Score: 0.8434
----------------------------------------
Modello: Naive Bayes
  - Accuracy: 0.9032
  - F2 Score: 0.8929
----------------------------------------
Modello: Decision Tree
  - Accuracy: 0.6452
  - F2 Score: 0.5625
----------------------------------------
Modello: Random Forest
  - Accuracy: 0.8065
  - F2 Score: 0.7831
----------------------------------------
Modello: Support Vector Machine
  - Accuracy: 0.9355
  - F2 Score: 0.9036
----------------------------------------
Modello: Neural Networks
  - Accuracy: 0.8387
  - F2 Score: 0.7927
----------------------------------------
Modello: XGBoost
  - Accuracy: 0.7419
  - F2 Score: 0.6790
----------------------------------------
```

```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.86      0.80        14
           1       0.87      0.76      0.81        17

    accuracy                           0.81        31
   macro avg       0.81      0.81      0.81        31
weighted avg       0.81      0.81      0.81        31

----------------------------------------
Support Vector Machine Classification Report:
              precision    recall  f1-score   support

           0       0.88      1.00      0.93        14
           1       1.00      0.88      0.94        17

    accuracy                           0.94        31
   macro avg       0.94      0.94      0.94        31
weighted avg       0.94      0.94      0.94        31
```

```
KNN Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.93      0.87        14
           1       0.93      0.82      0.88        17

    accuracy                           0.87        31
   macro avg       0.87      0.88      0.87        31
weighted avg       0.88      0.87      0.87        31

----------------------------------------
Naive Bayes Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.93      0.90        14
           1       0.94      0.88      0.91        17

    accuracy                           0.90        31
   macro avg       0.90      0.91      0.90        31
weighted avg       0.91      0.90      0.90        31

----------------------------------------
Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.58      0.79      0.67        14
           1       0.75      0.53      0.62        17

    accuracy                           0.65        31
   macro avg       0.66      0.66      0.64        31
weighted avg       0.67      0.65      0.64        31
```



```
Neural Networks Classification Report:
              precision    recall  f1-score   support

           0       0.76      0.93      0.84        14
           1       0.93      0.76      0.84        17

    accuracy                           0.84        31
   macro avg       0.85      0.85      0.84        31
weighted avg       0.85      0.84      0.84        31

---------------------------------------
XGBoost Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.86      0.75        14
           1       0.85      0.65      0.73        17

    accuracy                           0.74        31
   macro avg       0.76      0.75      0.74        31
weighted avg       0.77      0.74      0.74        31
```

È stato creato un report in cui ho analizzato le metriche di valutazione per ciascun modello. In particolare, ho approfondito tre indicatori fondamentali: precision, recall e f1-score. Queste metriche sono essenziali per comprendere le prestazioni dei vari modelli. È stato realizzato un report complessivo sui diversi metodi, al fine di confrontare l'accuratezza iniziale.

- PRECISION: la precision è una metrica che valuta l'accuratezza delle predizioni positive di un modello. In altre parole, indica la proporzione di instaze correttamente classificate come positive rispetto al totale delle istante classificate positive. Risulta particolarmente utile quando il costo è particolarmente elevato. La precision si calcola con la seguente formula:

$$Precision = \frac{Vero\ Positivo\ (TP)}{Vero\ Positivo(TP) +\ Falso\ Positivo(FP)}$$

- RECALL: la recall misura la capacità del modello di identificare tutte le istanze positive nel dataset. Indica la porzione di istanze positive correttamente classificate rispetto al totale delle istanze realemente positive. Risulta molto utile quando il costo di un falso negativo è elevato. La recall si calcola con la seguente formula:

$$Recall = \frac{Vero\ Positivo(TP)}{Vero\ Positivo(TP) + Falso\ Negativo(FN)}$$

- F1-SCORE: l'F1 Score è una misura che combina sia la precision che la recall in un unico valore. È utile in situazioni in cui si desidera il bilanciamento tra le due metriche. L'F1 Score è particolarmente rilevante quando di ha una distribuzione

sbilanciata delle classi, poiché fornisce un'indicazione più compleya delle performance del modello rispetto alla sola precision o alla sola recall. La formula dell'F1 Score è la seguente:

$$F1 = 2 * \frac{Precisiom * Recall}{Precision + Recall}$$

Le decisioni cliniche sono fortemente influenzate dalle metriche di valutazione come la precision e la recall. Un modello che presenta valori elevati in entrambe le metriche risulta più affidabile e, di conseguenza, può offrire un valido supporto ai medici nella diagnosi degli attacchi cardiaci.

Queste metriche non solo riflettono l'affidabilità del modello, ma forniscono anche prezioni informazioni sulle caratteristiche del dataset utilizzato. Per esempio, se un modello mostra una precisione elevata ma una recall bassa, ciò potrebbe indicare la necessità di un'analisi più approfondita del dataset stesso. Potrebbe esserci, infatti, uno squilibrio tra le classi di dati oppure alcuni sintomi potrebbero non essere adeguatamente rappresentati, compromettendo la qualità della diagnosi.

Durante il processo di sviluppo del modello, il monitoraggio continuo di queste metriche è importante per ottimizzare gli algoritmi di apprendimento automatico. Questo approccio non solo migliora le prestazioni predittive del modello, ma contribuisce anche a garantire diagnosi più accurate e tempestive.

Inoltre, identificare correttamente i pazienti a rischio di attacco di cuore non è solo una questione di diagnosi, ma può anche avere un impatto significativo sulle strategie di trattamento. Un'identificazione precisa consente l'implementazione di misure preventive, migliorando così la prognosi e riducendo il numero di eventi avversi. In questo modo, le metriche di precisione e recall si dimostrano fondamentali non solo nella fase di analisi, ma anche nella pratica quotidiana.

RANDOM FOREST

Il modello Random Forest viene impiegato per la classificazione di un dataset, nel caso specifico relativo alla diagnosi di un attacco cardiaco. La Random Forest è un algoritmo basato su un insieme di alberi decisionali: ciascun albero è addestrato su un diverso sottoinsieme casuale dei dati, il che rende il modello particolarmente robusto nei confronti

dell'overfitting. Questo metodo è in grado di gestire dataset complessi, caratterizzati da numerose variabili e da interazioni non lineari tra di esse.

Una volta completata la fase di addestramento, il modello viene utilizzato per effettuare previsioni sui dati di test. Queste previsioni vengono successivamente confrontate con i valori reali della variabile target, al fine di calcolare le metriche di valutazione. Al termine del processo di cross-validation, vengono calcolati i valori medi e le deviazioni standard per le due metriche principali, l'accuracy *e* il F2-score, considerando i 10 fold.

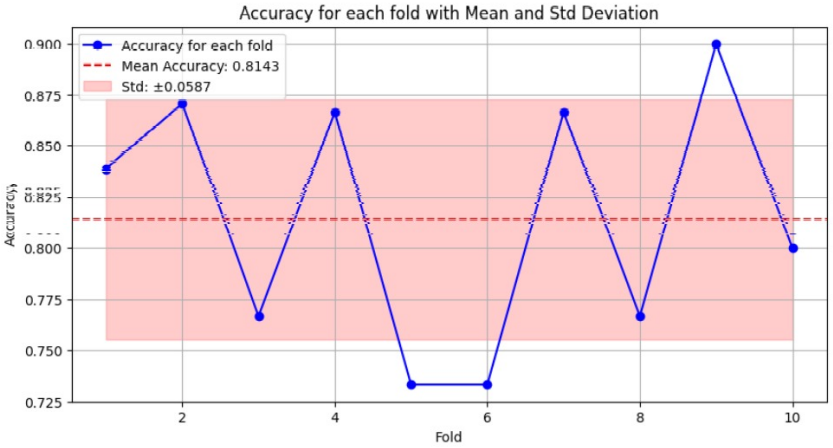
La media rappresenta una misura aggregata delle prestazioni del modello, riflettendo il livello di accuratezza medio ottenuto su tutti i fold, mentre la deviazione standard quantifica la variabilità dei risultati. Una deviazione standard bassa indica che il modello è stabile e fornisce risultati consistenti su tutti i fold, mentre una deviazione standard elevata suggerisce una maggiore variabilità nelle prestazioni, segnalando una possibile sensibilità del modello alla suddivisione dei dati.

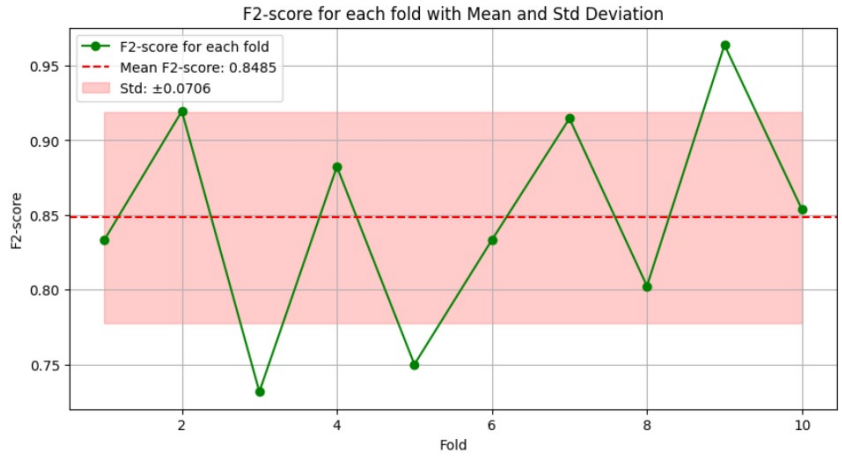
Inoltre, sono stati introdotti diversi iperparametri per migliorare le prestazioni del modello:

- n_estimators: il numero di alberi nella foresta, impostato su 100, che determina la dimensione dell'insieme di alberi decisionali.
- max_depth: la profondità massima degli alberi, limitata a 10 per prevenire l'overfitting e mantenere una struttura più semplice.
- max_features: il numero massimo di feature considerate per ciascuna suddivisione all'interno di un albero, impostato su sqrt, che permette di selezionare un sottoinsieme casuale di feature ad ogni nodo, favorendo la diversità tra gli alberi.
- n_jobs: impostato su -1, consente l'utilizzo di tutti i core del processore disponibili per velocizzare il processo di addestramento.

Questi iperparametri sono stati scelti per ottimizzare l'equilibrio tra prestazioni e capacità di generalizzazione del modello, riducendo al contempo i rischi di sovra-allenamento e migliorando la capacità predittiva.

```
Fold 1 completed
Fold 2 completed
Fold 3 completed
Fold 4 completed
Fold 5 completed
Fold 6 completed
Fold 7 completed
Fold 8 completed
Fold 9 completed
Fold 10 completed
Mean Accuracy over 10 folds: 0.8143
Standard Deviation of Accuracy: 0.0587
Mean F2-score over 10 folds: 0.8485
Standard Deviation of F2-score: 0.0706
```





Il codice genera due grafici:

- Grafico dell'accuratezza: ogni punto nel grafico rappresenta il valore di accuratezza ottenuto in uno dei fold durante la cross-validation. La linea tratteggiata indica la media dell'accuratezza su tutti i fold, mentre l'area colorata che circonda la linea rappresenta la variazione, espressa tramite la deviazione standard.

- Grafico del F2-score: simile al grafico dell'accuratezza, questo grafico mostra il valore dell'F2-score per ciascun fold, insieme alla media, con una banda che indica la deviazione standard.

Questi grafici offrono una rappresentazione visiva immediata della coerenza delle prestazioni del modello sui diversi fold, consentendo di valutare la stabilità del modello.

KNN (K-NEAREST NEIGHBORS)

L'algoritmo K-Nearest Neighbors (KNN) è un metodo di classificazione semplice e intuitivo. Il suo funzionamento si basa sull'assegnazione di una classe a un nuovo punto dati in base alla classe più frequente tra i suoi k vicini più prossimi. La distanza tra i punti dati, solitamente la distanza euclidea, viene utilizzata per identificare i vicini più prossimi. KNN non richiede un processo di addestramento esplicito, in quanto è un algoritmo memory-based, nel quale tutte le operazioni si svolgono al momento della previsione.

L'obiettivo del KNN è quello di addestrare e valutare il modello su un dataset relativo alla diagnosi di problemi cardiaci, attraverso una serie di esecuzioni ripetute. L'intento è analizzare le prestazioni del modello su diverse suddivisioni del dataset e misurare metriche come l'accuratezza e l'F2-score, con lo scopo di valutare la robustezza e la stabilità del modello in differenti contesti di test. Nel codice, grazie all'impiego della cross-validation, vengono calcolate la media e la deviazione standard delle metriche su più fold (10 in questo caso):

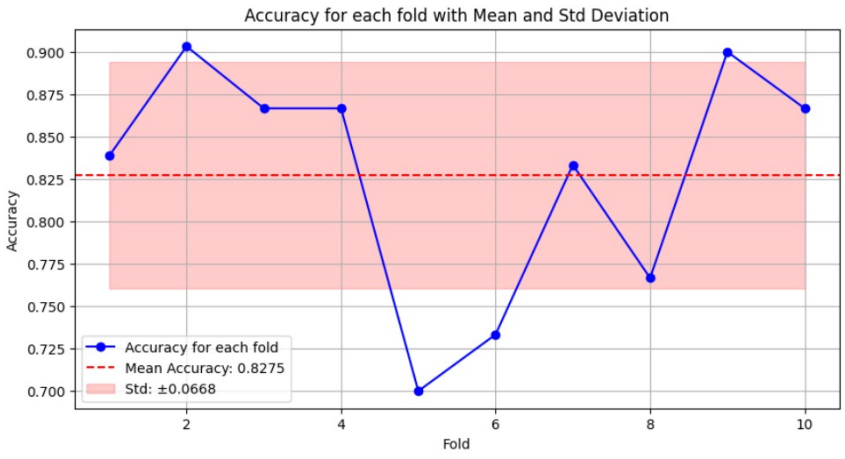
- La media rappresenta il comportamento medio del modello su tutti i fold.
- La deviazione standard quantifica la variabilità dei risultati rispetto alla media, fornendo un'indicazione sulla stabilità complessiva del modello.

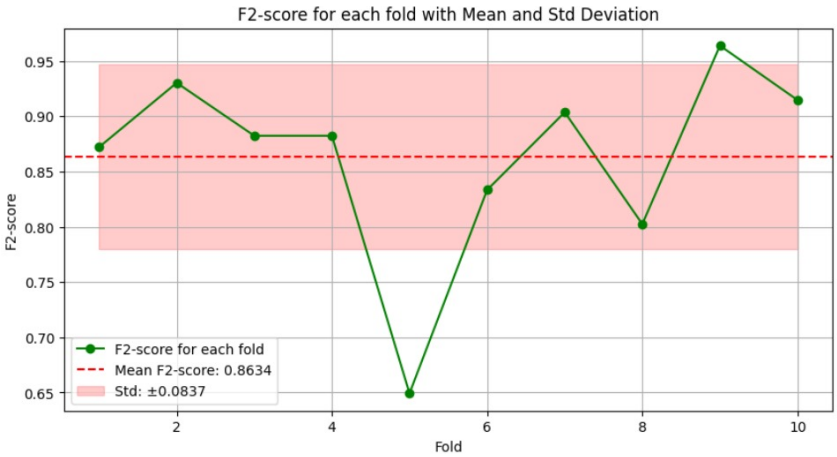
Nel modello K-Nearest Neighbors (KNN), la scelta degli iperparametri gioca un ruolo importante nel determinare le prestazioni del modello. Ecco una descrizione degli iperparametri aggiunti:

- n_neighbors: specifica il numero di vicini da considerare durante la classificazione. Un valore più alto può rendere il modello meno sensibile al rumore nei dati, ma potrebbe anche portare a una maggiore complessità computazionale.

- metric: indica la metrica di distanza utilizzata per misurare la distanza tra i punti nel nostro spazio delle feature. Il valore di default è 'minkowski', che include la distanza euclidea (p=2), ma è possibile specificare altre metriche come “manhattan”, “chebyshev”, o altre metriche personalizzate.
- p: parametro per la metrica Minkowski. Quando metric=”minkowski”, p=2 corrisponde alla distanza euclidea. p=1 corrisponde alla distanza di Manhattan. Questo parametro è rilevante solo se metric=”minkowski”.
- weights: specifica il metodo per pesare i punti vicini durante la previsione. 'uniform' assegna lo stesso peso a tutti i vicini, mentre 'distance' pesa i vicini in modo inversamente proporzionale alla loro distanza dal punto da classificare.

```
Fold 1 completed
Fold 2 completed
Fold 3 completed
Fold 4 completed
Fold 5 completed
Fold 6 completed
Fold 7 completed
Fold 8 completed
Fold 9 completed
Fold 10 completed
Mean Accuracy over 10 folds: 0.8275
Standard Deviation of Accuracy: 0.0668
Mean F2-score over 10 folds: 0.8634
Standard Deviation of F2-score: 0.0837
```





Nei grafici è possibile osservare:

- Accuratezza per ogni fold: vengono visualizzati i valori di accuratezza per ciascun fold tramite punti blu. La linea tratteggiata rossa rappresenta la media dell'accuratezza, mentre l'area ombreggiata mostra la deviazione standard. Questo grafico consente di valutare la consistenza dell'accuratezza tra i vari fold.
- F2-score per ogni fold: analogamente al grafico dell'accuratezza, questo grafico riporta il valore dell'F2-score per ciascun fold, con la media e la deviazione standard rappresentate nello stesso modo. Anche qui, è possibile osservare se le performance del modello sono stabili o se emergono variazioni significative tra i fold.

In entrambi i grafici, è possibile analizzare la costanza delle prestazioni del modello o identificare eventuali variazioni sostanziali tra i fold, che potrebbero indicare problemi di instabilità nel comportamento del modello.

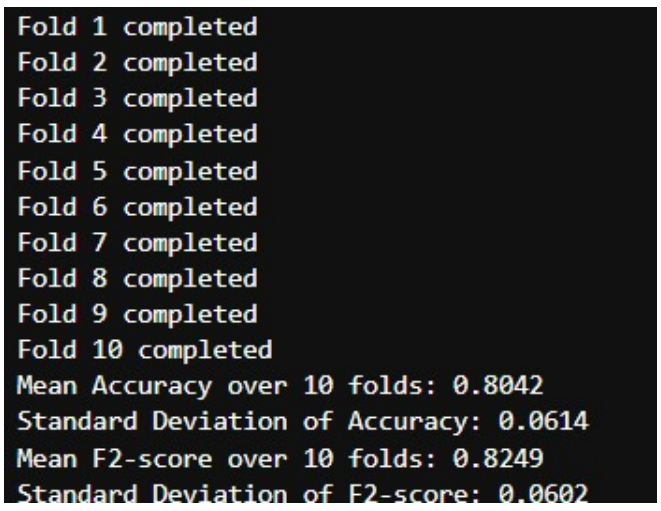
NAIVE BAYES GAUSSIANO

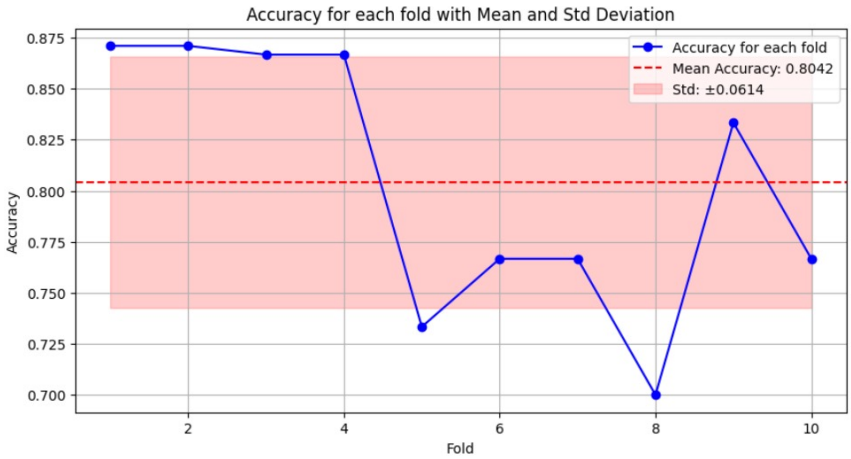
L'obiettivo del modello Naive Bayes Gaussiano (GaussianNB) è valutare le sue prestazioni su un dataset relativo a problemi cardiaci attraverso un processo iterativo basato su 10 esecuzioni (run). Questo approccio permette di analizzare come variano le metriche di prestazione, quali l'accuratezza e l'F2-score, su diverse suddivisioni dei dati di training e testing, garantendo una valutazione affidabile delle capacità predittive del modello.

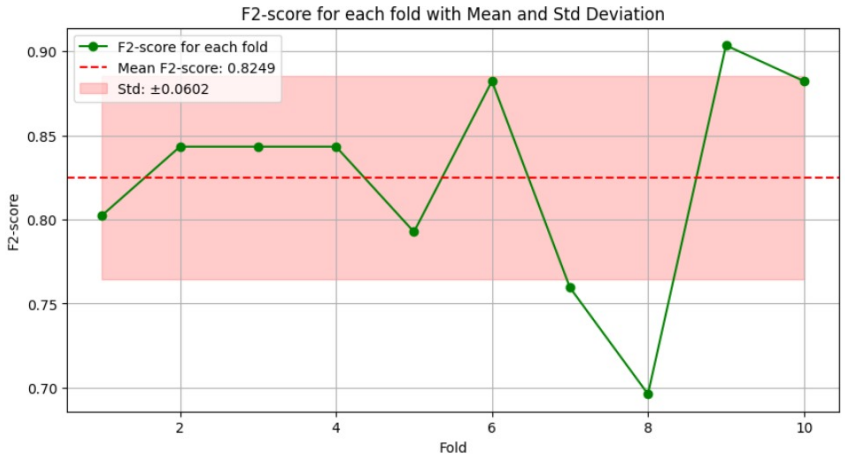
Il Naive Bayes Gaussiano è un classificatore probabilistico che si basa sul teorema di Bayes e sull'assunzione che le variabili siano indipendenti tra loro e seguano una distribuzione gaussiana (normale). È particolarmente adatto per dataset con variabili numeriche e può essere applicato sia a problemi di classificazione binaria che multiclasse. La sua semplicità computazionale lo rende rapido da addestrare, sebbene l'ipotesi di indipendenza delle variabili sia spesso una semplificazione eccessiva.

Nel codice vengono calcolate sia l'accuratezza, che rappresenta la percentuale di predizioni corrette sui dati di test, sia l'F2-score, che combina precisione e recall. Queste metriche vengono calcolate per ogni fold durante la cross-validation. Successivamente, vengono determinate la media e la deviazione standard di entrambe le metriche su tutti i fold, fornendo così una valutazione complessiva delle prestazioni del modello e della sua stabilità.

Inoltre, è stato inserito var_smoothing è un iperparametro del modello GaussianNB che aggiunge una piccola quantità alla varianza di ogni feature durante il calcolo delle probabilità. Questo serve a stabilizzare i calcoli, specialmente quando alcune feature hanno una varianza molto bassa o vicina a zero, evitando così problemi numerici come divisioni per zero. In pratica, aiuta a rendere il modello più robusto quando le feature presentano valori simili o costanti. Il valore predefinito di var_smoothing è 1e-9, ma può essere regolato per ottimizzare le prestazioni del modello in base al dataset.







Nei grafici è possibile osservare e analizzare:

- Accuratezza per ogni fold: questo grafico visualizza l'accuratezza ottenuta su ciascun fold durante la cross-validation. I punti blu rappresentano i valori di accuratezza per i singoli fold, mentre la linea tratteggiata rossa indica la media dell'accuratezza su tutti i fold. L'area ombreggiata rossa rappresenta la deviazione standard rispetto alla media, consentendo di valutare la consistenza del modello nelle diverse suddivisioni del dataset.
- F2-score per ogni fold: in maniera analoga al grafico dell'accuratezza, questo grafico mostra l'F2-score per ciascun fold tramite punti verdi. La linea tratteggiata rossa rappresenta la media dell'F2-score, e l'area ombreggiata evidenzia la deviazione standard. Questo permette di valutare se vi siano variazioni significative nelle prestazioni tra i diversi fold.

Questi grafici sono strumenti utili per valutare la stabilità del modello, mostrando se le metriche rimangono costanti o se variano tra i diversi fold. Ciò fornisce indicazioni sulla robustezza complessiva del modello.

DECISION TREE

L'albero decisionale (Decision Tree) viene impiegato per valutare la stabilità e l'affidabilità del modello attraverso una serie di esecuzioni (run), durante le quali vengono calcolate l'accuratezza e l'F2-score. Questo approccio consente di ottenere una visione complessiva delle capacità predittive del modello, riducendo l'incertezza che potrebbe derivare da una singola suddivisione del dataset.

L'albero decisionale è un modello di classificazione non parametrico che suddivide i dati in modo iterativo, basandosi su condizioni stabilite dai valori delle feature. Ogni nodo dell'albero rappresenta una condizione di separazione dei dati, mentre le foglie finali indicano la classe predetta. Questo modello è particolarmente apprezzato per la sua elevata interpretabilità, in quanto le decisioni prese dall'albero possono essere facilmente espresse sotto forma di regole comprensibili.

Nel codice vengono utilizzate due metriche principali per la valutazione delle prestazioni del modello:

- Accuratezza: misura la proporzione di predizioni corrette rispetto al numero totale di osservazioni. Si calcola come la somma di veri positivi e veri negativi divisa per il totale degli esempi.
- F2-score: combina precisione e recall, attribuendo maggiore peso al recall (con $\beta=2$). È particolarmente utile in contesti in cui è cruciale identificare correttamente le istanze positive, come nei problemi legati alla salute.

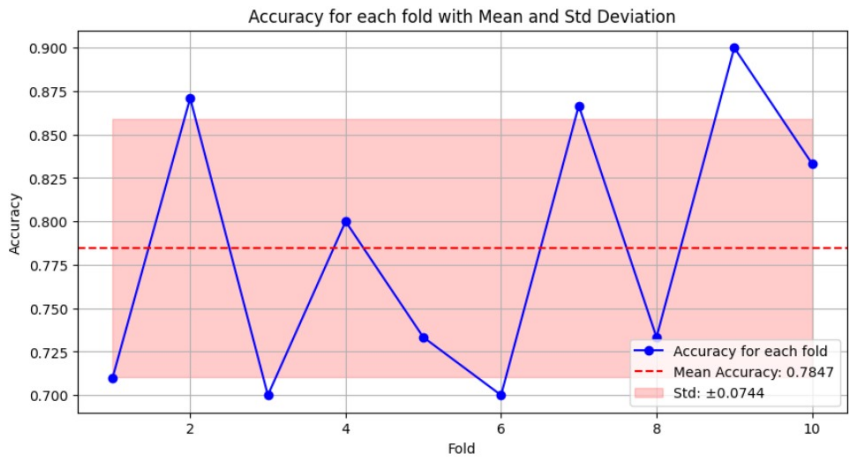
Dopo aver eseguito la cross-validation, il codice calcola la media e la deviazione standard delle metriche. La media rappresenta una valutazione aggregata delle prestazioni del modello, mentre la deviazione standard fornisce un'indicazione della variabilità delle performance su diversi fold.

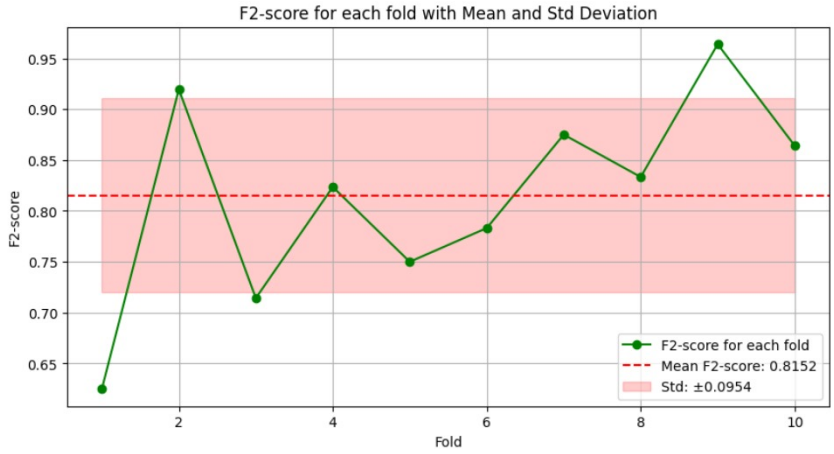
Inoltre, sono stati introdotti degli iperparametri per migliorare le prestazioni del modello:

- "criterion=gini": definisce il criterio per valutare la qualità delle suddivisioni. L'opzione "gini" utilizza l'indice di Gini, che misura l'impurità di un nodo. L'obiettivo è minimizzare l'impurità in ogni suddivisione.

- “max_depth=5”: limita la profondità massima dell'albero, evitando che diventi troppo profondo. Un valore basso come 5 riduce il rischio di overfitting, limitando la complessità del modello.
- “min_samples_split=10”: specifica il numero minimo di campioni necessari per effettuare una suddivisione in un nodo. Con “min_samples_split=10”, l'albero non suddividerà un nodo a meno che non ci siano almeno 10 campioni.
- “min_samples_leaf=5`: imposta il numero minimo di campioni richiesti in un nodo foglia. Con “min_samples_leaf=5”, si garantisce che ogni foglia contenga almeno 5 campioni, riducendo il rischio di avere foglie troppo specifiche che potrebbero indicare overfitting.
- “random_state=42”: questo parametro assicura la riproducibilità dei risultati. Usando un valore fisso, come 42, si garantisce che l'algoritmo produca sempre gli stessi risultati se eseguito con le stesse condizioni e dati.

```
Fold 1 completed
Fold 2 completed
Fold 3 completed
Fold 4 completed
Fold 5 completed
Fold 6 completed
Fold 7 completed
Fold 8 completed
Fold 9 completed
Fold 10 completed
Mean Accuracy over 10 folds: 0.7847
Standard Deviation of Accuracy: 0.0744
Mean F2-score over 10 folds: 0.8152
Standard Deviation of F2-score: 0.0954
```





Dall'output è possibile analizzare:

- Grafico dell'accuratezza: visualizza i valori di accuratezza per ciascun fold della cross-validation. I punti blu rappresentano l'accuratezza ottenuta in ogni fold, mentre la linea tratteggiata rossa indica la media dell'accuratezza su tutti i fold. L'area ombreggiata attorno alla media rappresenta la deviazione standard, evidenziando la variabilità delle prestazioni. Un'accuratezza costante e vicina alla media indica che il modello è robusto e affidabile, mantenendo prestazioni stabili attraverso le diverse suddivisioni del dataset.
- Grafico dell'F2-score: in maniera simile al grafico dell'accuratezza, questo grafico mostra i valori dell'F2-score per ciascun fold. I punti verdi rappresentano i singoli valori dell'F2-score, mentre la linea tratteggiata rossa rappresenta la media su tutti i fold. L'area ombreggiata intorno alla linea rossa evidenzia la deviazione standard. Questo grafico permette di identificare eventuali fluttuazioni significative nelle performance del modello, soprattutto per quanto riguarda il rilevamento delle istanze positive, il che è particolarmente rilevante in applicazioni dove il recall è cruciale.

Questi grafici forniscono una chiara indicazione sulla stabilità e affidabilità del modello, evidenziando se le prestazioni sono consistenti o se variano significativamente tra i diversi fold.

XGBOOST

XGBoost (Extreme Gradient Boosting) è un algoritmo di apprendimento automatico basato sul metodo del gradient boosting, noto per la sua efficienza e per le elevate

prestazioni nei problemi di classificazione e regressione. Utilizza alberi decisionali in modo sequenziale, in cui ogni albero si propone di correggere gli errori del precedente, risultando in un modello robusto e preciso.

Il codice impiega la cross-validation utilizzando il metodo StratifiedKFold per garantire che la distribuzione delle classi rimanga bilanciata tra i vari fold. Questo aspetto è particolarmente rilevante nei problemi di classificazione in cui le classi possono presentare uno sbilanciamento significativo.

Le metriche utilizzate per valutare le prestazioni del modello comprendono: accuratezza e f2-score

Dopo l'esecuzione della cross-validation, il codice calcola:

- Media: fornisce un'indicazione complessiva delle prestazioni del modello su tutti i fold.
- Deviazione standard: misura quanto variano le prestazioni del modello tra i diversi fold, evidenziando la stabilità del modello nel tempo.

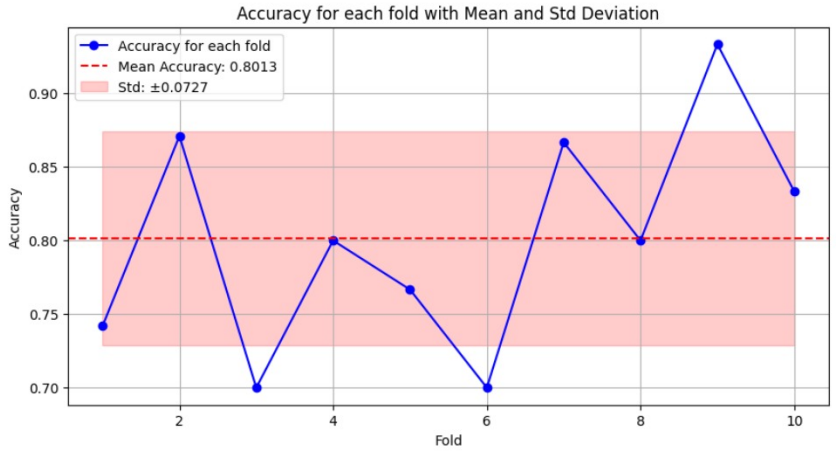
Queste analisi consentono di ottenere una valutazione completa dell'affidabilità e della robustezza del modello.

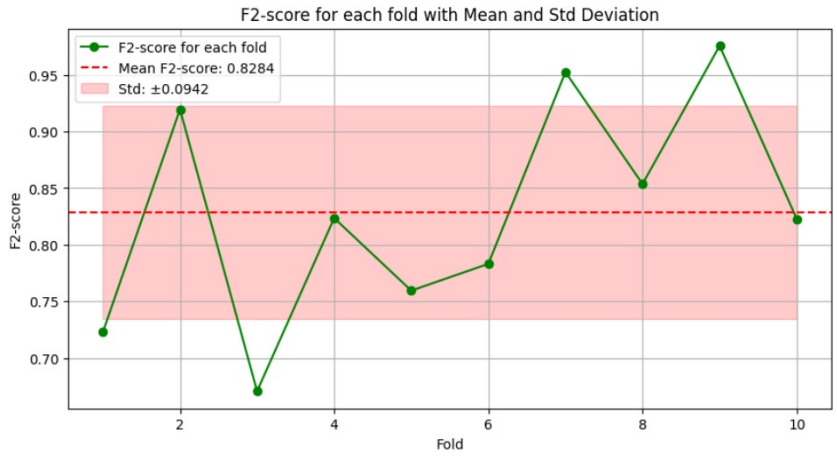
Sono stati, anche, inseriti i seguenti iperparametri, utili per regolare il comportamento di XGBoost.

- “n_estimators”: il numero di alberi che verranno costruiti dal modello. Un valore più alto può aumentare l'accuratezza ma potrebbe causare overfitting.
- “max_depth”: la profondità massima di ciascun albero. Controlla quanto profondamente ogni albero può ramificarsi. Valori più alti possono permettere al modello di catturare più dettagli, ma possono anche causare overfitting.
- “learning_rate”: il tasso di apprendimento per il boosting. Riduce l'impatto di ogni albero per rendere il modello più stabile e meno incline a sovra-adattarsi. Valori più bassi richiedono più alberi (n_estimators) per ottenere buone prestazioni.
- “scale_pos_weight”: bilancia il peso delle classi nel caso di dataset sbilanciati. Utile quando ci sono molte più osservazioni in una classe rispetto all'altra.

- “random_state”: imposta un seme casuale per garantire che i risultati siano riproducibili.

```
Fold 1 completed
Fold 2 completed
Fold 3 completed
Fold 4 completed
Fold 5 completed
Fold 6 completed
Fold 7 completed
Fold 8 completed
Fold 9 completed
Fold 10 completed
Mean Accuracy over 10 folds: 0.8013
Standard Deviation of Accuracy: 0.0727
Mean F2-score over 10 folds: 0.8284
Standard Deviation of F2-score: 0.0942
```





Il primo grafico illustra l'accuratezza per ogni fold della cross-validation. I punti blu rappresentano le accuratezze ottenute in ciascun fold, mentre la linea rossa tratteggiata indica la media dell'accuratezza. L'area ombreggiata attorno alla media rappresenta la deviazione standard. Un modello caratterizzato da un'accuratezza media elevata e una

bassa deviazione standard suggerisce che il modello è consistente e robusto attraverso le diverse suddivisioni del dataset.

Il secondo grafico rappresenta l'F2-score per ogni fold. I punti verdi indicano i valori dell'F2-score, con la linea rossa che mostra la media. La zona ombreggiata intorno alla media evidenzia la deviazione standard. Un F2-score elevato e consistente tra i fold è indicativo di un modello efficace nel rilevare le istanze positive, aspetto particolarmente rilevante in situazioni in cui è fondamentale minimizzare i falsi negativi.

L'analisi condotta utilizzando il classificatore XGBoost fornisce una valutazione dettagliata delle prestazioni del modello attraverso metriche chiave come l'accuratezza e l'F2-score. I grafici generati offrono una visualizzazione chiara della stabilità e dell'affidabilità del modello. Un buon equilibrio tra le metriche indica che il modello è stato adeguatamente addestrato e ha la capacità di generalizzare efficacemente le sue previsioni. Ciò è particolarmente rilevante in contesti applicativi critici, in cui decisioni accurate possono avere un impatto significativo sulla salute e sul benessere delle persone.

NEURAL NETWORK

Il codice implementa una rete neurale semplice utilizzando Keras, una rinomata libreria di alto livello integrata in TensorFlow, progettata per semplificare lo sviluppo e l'addestramento di modelli di deep learning. Le reti neurali artificiali, ispirate al funzionamento del cervello umano, sono costituite da strati di neuroni artificiali (nodi) collegati tra loro e hanno la capacità di apprendere relazioni complesse tra input e output.

Queste reti si dimostrano particolarmente efficaci nel risolvere problemi di classificazione, in cui l'obiettivo è assegnare le osservazioni a una o più categorie predefinite, soprattutto quando i dati presentano complessità e non linearità. Un modello di rete neurale, ad esempio, può essere impiegato per riconoscere oggetti in immagini, classificare sentimenti in testi o persino fare previsioni su serie temporali.

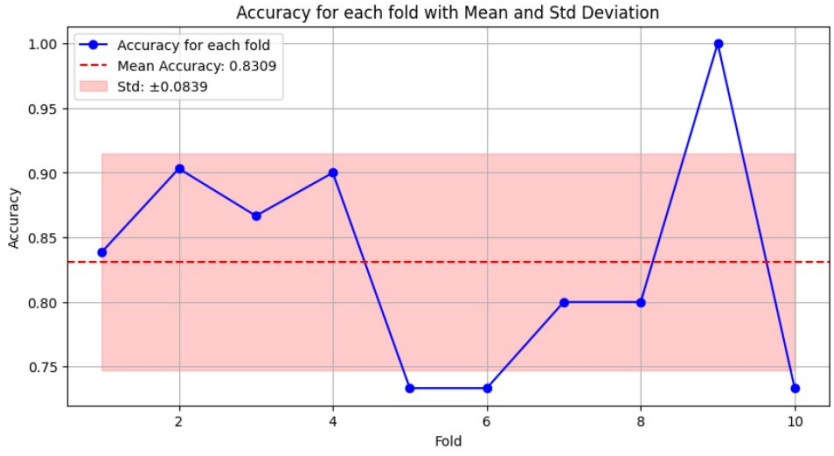
Il codice utilizza la cross-validation per garantire una distribuzione bilanciata delle classi, un aspetto essenziale per evitare che il modello venga influenzato da un campione non rappresentativo. Sono state adottate metriche di valutazione, tra cui l'accuratezza e l'F2-score. Queste due metriche sono state utilizzate per calcolare sia la media che la deviazione standard.

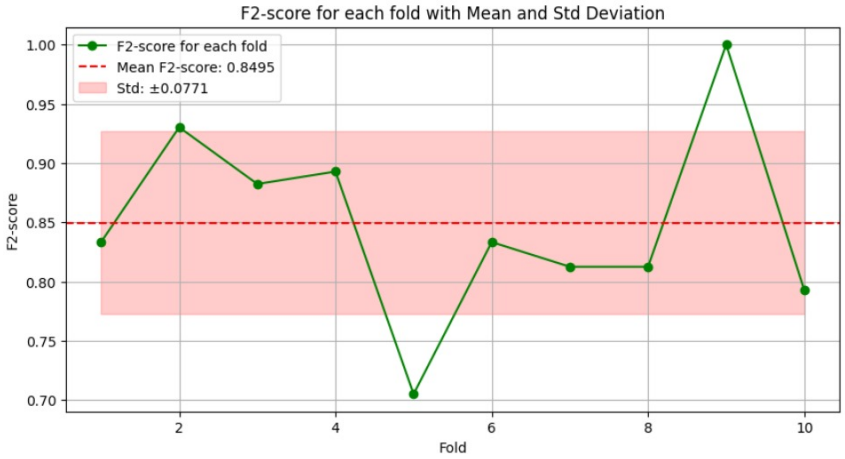
Questa metodologia consente di ottenere una valutazione dettagliata delle prestazioni del modello, contribuendo a garantire la sua robustezza e affidabilità nelle applicazioni pratiche.

Inoltre, sono stati aggiunti degli iperparametri, utili per migliorare le prestazioni del modello

- hidden_layers: numero di strati nascosti nella rete neurale.
- hidden_units: lista del numero di neuroni in ciascun strato nascosto.
- activation: funzione di attivazione utilizzata nei neuroni (es. 'relu', 'sigmoid').
- earning_rate: tasso di apprendimento per l'ottimizzatore.
- epochs: numero di epoche durante le quali il modello viene addestrato.
- batch_size: numero di campioni utilizzati per aggiornare i pesi del modello in un singolo passo di addestramento.

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step
Fold 1 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 204ms/step
Fold 2 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 164ms/step
Fold 3 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 162ms/step
Fold 4 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 157ms/step
Fold 5 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step
Fold 6 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 173ms/step
Fold 7 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 176ms/step
Fold 8 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 173ms/step
Fold 9 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 184ms/step
Fold 10 completed
Mean Accuracy over 10 folds: 0.8309
Standard Deviation of Accuracy: 0.0839
Mean F2-score over 10 folds: 0.8495
Standard Deviation of F2-score: 0.0771
```





I grafici ci permettono di visualizzare sia l'accuratezza per ogni fold. I punti blu rappresentano l'accuratezza per ciascun fold, mentre la linea rossa tratteggiata indica l'accuratezza media. L'area ombreggiata intorno alla media mostra la deviazione standard. Un modello robusto avrà un'accuratezza media alta e una deviazione standard bassa, suggerendo coerenza nelle prestazioni. Sia l'F2-score per ciascun fold. I punti verdi indicano i valori di F2-score, con la linea rossa che rappresenta la media. L'area ombreggiata indica la deviazione standard. Un F2-score elevato e consistente suggerisce che il modello è efficace nel rilevare le istanze positive, minimizzando i falsi negativi.

L'implementazione di una rete neurale per la classificazione mostra l'efficacia delle tecniche di deep learning nel gestire problemi complessi. La combinazione di cross-validation e metriche di valutazione come l'accuratezza e l'F2-score fornisce una valutazione completa delle prestazioni del modello. I grafici prodotti offrono una visualizzazione chiara della stabilità e dell'affidabilità del modello, fondamentale in applicazioni dove le decisioni possono avere un impatto significativo, come nel campo medico.

SUPPORT VECTOR MACHINE

Il codice impiega un Support Vector Machine (SVM) con kernel lineare per classificare un dataset relativo alla salute cardiaca, il quale include variabili (features) che descrivono lo stato di salute del paziente e una variabile target che indica la presenza o assenza di malattia cardiaca.

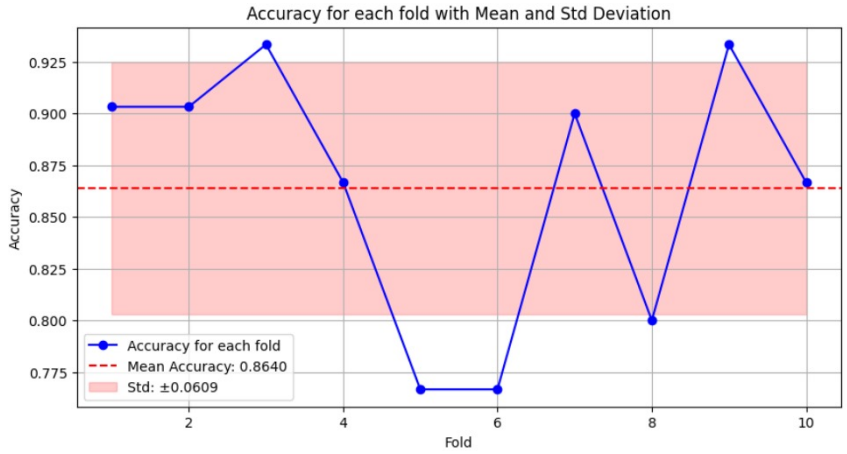
Per ciascun fold, il modello viene addestrato sui dati di training e successivamente valutato sui dati di test. Durante questo processo, vengono calcolate due metriche di

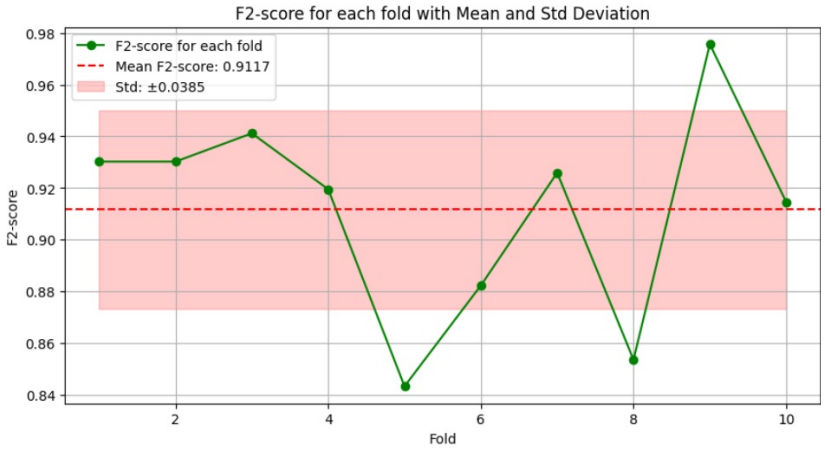
prestazione: l'accuratezza e l'F2-score. Queste metriche consentono di analizzare l'efficacia del modello nella previsione della condizione cardiaca del paziente, fornendo una valutazione complessiva delle sue capacità predittive.

Sono stati aggiunti anche degli iperparametri, per migliorare le prestazioni del modello:

- C: regolarizzazione e trade-off tra complessità del modello e accuratezza.
- Kernel: metodo per trasformare i dati; può essere lineare o non lineare (es. RBF).
- Gamma: influenza di un singolo esempio di addestramento, specifico per kernel RBF.

```
Fold 1 completed
Fold 2 completed
Fold 3 completed
Fold 4 completed
Fold 5 completed
Fold 6 completed
Fold 7 completed
Fold 8 completed
Fold 9 completed
Fold 10 completed
Mean Accuracy over 10 folds: 0.8273
Standard Deviation of Accuracy: 0.0654
Mean F2-score over 10 folds: 0.8643
Standard Deviation of F2-score: 0.0728
```





Il grafico delle accuratezze illustra l'accuratezza per ciascun fold, con una linea media rossa che rappresenta l'accuratezza media complessiva del modello. La banda attorno alla linea media indica la deviazione standard, fornendo un'indicazione della variabilità dell'accuratezza tra i fold. Se la banda è stretta, ciò suggerisce che il modello è coerente; al contrario, se è ampia, vi è una maggiore variabilità tra i fold. Questo tipo di grafico è utile per valutare l'eventuale presenza di differenze significative tra i fold e per determinare se l'accuratezza del modello è costante.

Analogamente, il grafico degli F2-score presenta l'F2-score per ciascun fold rappresentato individualmente, con una linea rossa che indica il valore medio dell'F2-score. La banda di deviazione standard attorno alla media consente di visualizzare la variabilità delle prestazioni del modello in termini di recall e precisione. La stabilità di questo punteggio è particolarmente rilevante in contesti medici, dove minimizzare i falsi negativi, ovvero malati non diagnosticati, è cruciale.

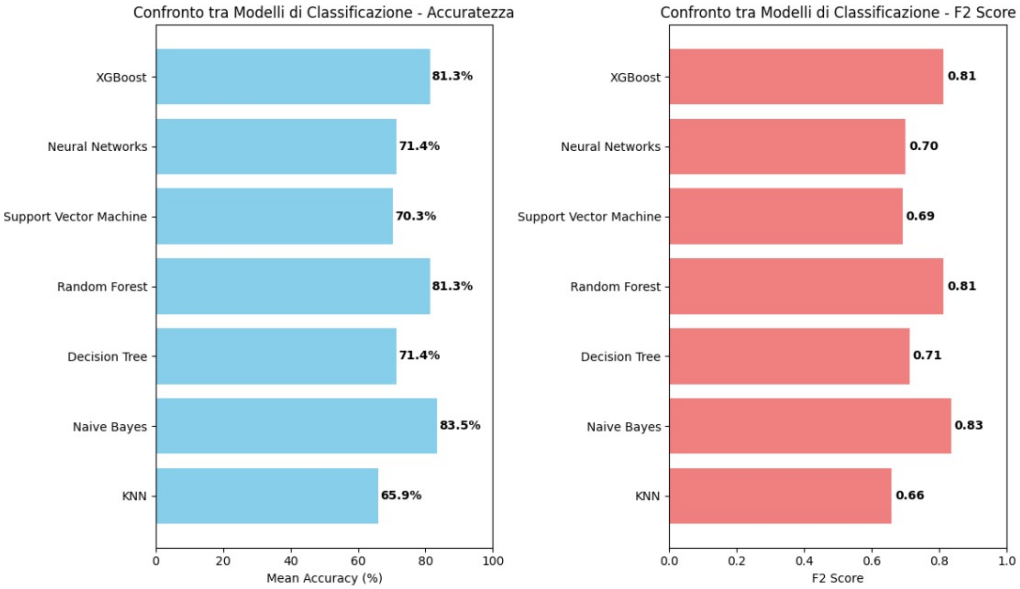
VALUTAZIONI FINALI

Il grafico che confronta l’accuratezza e l'F2-score di diversi modelli di classificazione offre una chiara panoramica delle loro performance relative. Partendo dall'accuratezza, si può osservare come il modello Naive Bayes risulti il più efficace, con un’accuratezza dell’83,5%. Questo risultato evidenzia la capacità del modello di effettuare predizioni corrette nella maggior parte dei casi. Subito dopo, i modelli XGBoost e Random Forest si posizionano con un’accuratezza molto simile, pari all'81,3%. Entrambi questi modelli, noti per le loro strutture basate su alberi e tecniche di ensemble, confermano la loro robustezza in termini di predizioni accurate.

Le reti neurali e l’albero decisionale (Decision Tree), invece, mostrano un’accuratezza inferiore, attestandosi al 71,4%. Questi modelli, pur essendo ampiamente utilizzati in vari contesti, qui sembrano non raggiungere la precisione dei modelli basati su ensemble. Il Support Vector Machine (SVM) si colloca appena sotto, con un’accuratezza del 70,3%, confermando una prestazione leggermente inferiore rispetto ai modelli precedentemente menzionati. Infine, K-Nearest Neighbors (KNN) si posiziona all'ultimo posto in termini di accuratezza, con un risultato del 65,9%, dimostrando di essere il modello meno performante in questo confronto specifico.

Passando all’analisi dell’F2-score, il modello Naive Bayes si conferma ancora una volta il più performante, con un punteggio di 0,83. Questo indica che non solo il modello è accurato, ma è particolarmente efficace nel bilanciare la precisione con il recall, ponendo maggiore attenzione su quest’ultimo. Anche i modelli XGBoost e Random Forest ottengono un ottimo F2-score, con un valore di 0,81, confermando la loro affidabilità nel gestire sia la precisione che il recall. L’albero decisionale, con un F2-score di 0,71, segue la stessa tendenza già vista nell’accuratezza, con una performance moderata, simile a quella delle reti neurali, che ottengono un F2-score di 0,70. Anche il Support Vector Machine, con un valore di 0,69, si conferma un modello che, pur essendo efficace, non riesce a raggiungere le prestazioni dei migliori algoritmi in termini di bilanciamento tra precisione e recall. KNN, infine, registra un F2-score di 0,66, confermando di essere il modello meno efficiente in questo contesto.

In sintesi, dall'analisi congiunta di accuratezza e F2-score emerge che Naive Bayes si distingue come il modello complessivamente più performante, seguito a breve distanza da XGBoost e Random Forest. I modelli come le reti neurali, l’albero decisionale e il Support Vector Machine si piazzano a metà classifica, con prestazioni medie ma non eccezionali. KNN, invece, risulta essere il modello meno efficace tra quelli analizzati. La scelta del miglior modello dipende quindi dalle esigenze specifiche: se si privilegia l’accuratezza complessiva o un bilanciamento ottimale tra precisione e recall, i modelli Naive Bayes, XGBoost e Random Forest rappresentano le opzioni più solide.



Nel confronto tra i risultati iniziali e quelli finali dei modelli di classificazione, emerge un quadro chiaro delle prestazioni, con alcuni modelli che hanno beneficiato di un netto miglioramento, mentre altri hanno visto una leggera flessione. Analizzando l'accuratezza e l'F2-score di ciascun modello, si nota come le ottimizzazioni abbiano inciso sulle loro capacità predittive.

Il modello K-Nearest Neighbors (KNN) ha registrato un progresso significativo. Nella fase iniziale l'accuratezza era del 65,9%, mentre nei risultati finali ha raggiunto l'87,1%, mostrando un incremento di oltre venti punti percentuali. Anche l'F2-score è migliorato, passando da 0,66 a 0,8434, segno di una maggiore capacità del modello di bilanciare recall e precisione, rendendolo quindi più efficace nel gestire le classificazioni corrette.

Naive Bayes, già un modello performante nella fase iniziale, ha visto un ulteriore affinamento. L'accuratezza iniziale, che era dell'83,5%, è salita al 90,32% nei risultati finali, consolidando ulteriormente la solidità del modello. Anche l'F2-score ha seguito un percorso di crescita, passando da 0,83 a 0,8929, a indicare una maggiore capacità nel riconoscere correttamente i casi positivi. Questo miglioramento rende Naive Bayes uno dei modelli più stabili e affidabili tra quelli analizzati.

Di contro, il Decision Tree ha mostrato un peggioramento evidente. Nella fase iniziale, l'accuratezza era del 71,4%, ma nei risultati finali è scesa al 64,52%, suggerendo una perdita di precisione nel processo di classificazione. Lo stesso andamento si riflette nell'F2-score, che è calato da 0,71 a 0,5625, segnalando una diminuzione nella capacità

del modello di bilanciare in modo efficace precisione e recall. Questo declino indica una minore efficienza rispetto ai risultati iniziali.

Random Forest, invece, ha mostrato una leggera flessione nelle prestazioni. L'accuratezza è passata dall'81,3% nella fase iniziale all'80,65% nei risultati finali, una riduzione modesta che non ha inciso pesantemente sulla performance complessiva del modello. Anche l'F2-score ha subito un lieve calo, da 0,81 a 0,7831, ma questo non ha comportato una perdita significativa della sua efficacia, mantenendo comunque una stabilità nelle prestazioni.

Il modello Support Vector Machine (SVM) ha rappresentato uno dei miglioramenti più notevoli. Partendo da un'accuratezza iniziale del 70,3%, ha raggiunto un impressionante 93,55% nei risultati finali, evidenziando un miglioramento di oltre venti punti percentuali. Anche l'F2-score ha registrato un netto incremento, passando da 0,69 a 0,9036, rendendo questo modello uno dei più efficienti nel bilanciare recall e precisione, soprattutto nelle situazioni dove l'identificazione dei casi positivi è cruciale.

Le reti neurali hanno anch'esse mostrato un buon margine di miglioramento. Inizialmente l'accuratezza era del 71,4%, ma nei risultati finali è aumentata all'83,87%, dimostrando un affinamento significativo del modello. L'F2-score ha seguito lo stesso trend positivo, salendo da 0,70 a 0,7927, a conferma di una maggiore efficacia nel gestire la classificazione dei dati.

Infine, XGBoost ha evidenziato un netto peggioramento rispetto alla fase iniziale. L'accuratezza, che era dell'81,3%, è scesa al 74,19% nei risultati finali. Anche l'F2-score ha subito un calo significativo, passando da 0,81 a 0,6790, suggerendo che il modello ha perso parte della sua capacità di bilanciare precisione e recall in modo ottimale.

In conclusione, dal confronto tra i risultati iniziali e finali si nota un miglioramento considerevole per modelli come KNN, Naive Bayes, Support Vector Machine e reti neurali, che hanno beneficiato di ottimizzazioni che ne hanno migliorato le performance complessive. Al contrario, modelli come Decision Tree e XGBoost hanno visto una riduzione delle loro capacità, dimostrando di essere meno efficienti nella seconda fase. Random Forest, pur subendo un lieve calo, ha mantenuto una buona stabilità nelle sue prestazioni. Questi risultati evidenziano l'importanza delle ottimizzazioni nella scelta del modello giusto, a seconda delle esigenze specifiche di accuratezza e bilanciamento tra precisione e recall.