Relazione finale

Introduzione al problema

Il dataset utilizzato in questo progetto, disponibile su Kaggle, contiene informazioni cliniche essenziali relative a pazienti con potenziale rischio di insufficienza cardiaca.

L'insufficienza cardiaca rappresenta la causa principale di mortalità a livello mondiale, richiedendo approcci efficaci per la diagnosi precoce e la prevenzione.

Per quanto riguarda le morti derivanti da problemi cardiovascolari, quattro su cinque di queste sono dovute ad attacchi di cuore o ictus, e un terzo di queste morti avviene prematuramente in persone di età inferiore a 70 anni. Identificare tempestivamente i pazienti a rischio è fondamentale per ridurre complicazioni e costi sanitari associati. In questo contesto, l'apprendimento automatico (machine learning) si presenta come uno strumento promettente per supportare i professionisti sanitari nell'individuare pattern e relazioni complesse tra le variabili cliniche che possono non essere immediatamente evidenti con le tecniche tradizionali.

Le feature a disposizione sono 11, ovvero:

1. **Age**: età in anni del paziente
2. **Sex**: sesso del paziente, nei valori M o F
3. **ChestPainType**: tipologia di dolore al petto
    a. *TA*: Typical Angina, dolore toracico classico causato da ischemia cardiaca
    b. *ATA*: Atypical Angina, dolore al petto che non segue tutte le caratteristiche della tipica angina
    c. *NAP*: Non-anginal Pain, dolore toracico non correlato a cause cardiache
    d. *ASY*: Asymptomatic, assenza di sintomi o dolore al petto
4. **RestingBP**: pressione arteriosa a riposo misurata in millimetri di mercurio (mm Hg)
5. **Cholesterol**: livello di colesterolo sierico (mg/dl)
6. **FastingBS**: glicemia a digiuno (valore booleano, 1: se > 120 mg/dl, 0: altrimenti)
7. **RestingECG**: risultati dell'elettrocardiogramma a riposo
    a. *Normal*: normale

    b. *ST*: anomalia dell'onda ST-T (inversione dell'onda T e/o sopraelevazione o depressione del tratto ST > 0.05 mV)
    c. *LVH*: possibile o certa ipertrofia ventricolare sinistra secondo i criteri di Estes
8. **MaxHR**: massima frequenza cardiaca raggiunta, in range tra 60 e 202 bpm
9. **ExerciseAngina**: angina indotta dall'esercizio (booleano)
10. **Oldpeak**: depressione del tratto ST sull'ECG in valore numerico
11. **ST_Slope**: categoria del segmento ST al picco dell'esercizio fisico:
    a. "Up" aumento;
    b. "Flat" (piatto) nessuna variazione;
    c. "Down" (discendente).

Il target è quello di individuare la presenza o meno di un Heart Disease; dunque possiamo categorizzare il problema come una classificazione binaria.

L'obiettivo principale di questo progetto è sviluppare un modello di machine learning in grado di prevedere la presenza o l'assenza di insufficienza cardiaca sulla base di queste caratteristiche cliniche.

Affrontare questo problema con tecniche di machine learning può portare a una diagnosi più rapida e accurata, migliorando gli esiti per i pazienti e ottimizzando le risorse mediche. Inoltre, l'analisi dei dati può contribuire a comprendere meglio i fattori di rischio più significativi, fornendo indicazioni utili per strategie preventive e interventi mirati.

Questo progetto si concentrerà sulle seguenti fasi principali:

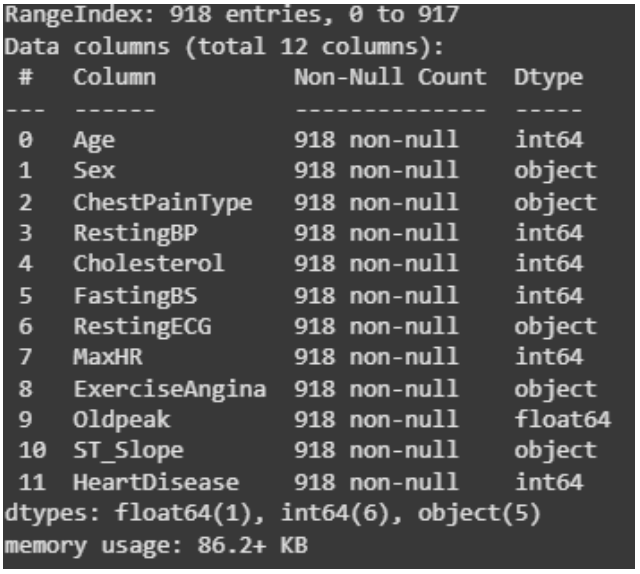
1. **Esplorazione e Preprocessing dei Dati:** Analisi statistica descrittiva e gestione dei dati mancanti o anomali.
2. **Costruzione dei Modelli:** Implementazione e confronto di diversi algoritmi di machine learning.
3. **Valutazione delle Prestazioni:** Utilizzo di metriche appropriate per misurare l'efficacia del modello.
4. **Interpretazione e Applicazione dei Risultati:** Discussione sui risultati ottenuti e sulle possibili applicazioni pratiche.

Attraverso questo progetto, si mira a dimostrare come l'integrazione di dati clinici e algoritmi predittivi possa supportare le decisioni mediche, contribuendo alla lotta contro l'insufficienza cardiaca e alla promozione di una medicina preventiva e personalizzata. In particolare sarà interessante osservare le differenze di prestazioni di modelli con complessità differente.

Analisi esplorativa

Statistiche descrittive

Analisi delle covariate e distribuzioni
Il dataset è costituito da 918 istanze, per ciascuna delle quali sono registrati i valori delle 11 covariate.



```
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    object
 2   ChestPainType   918 non-null    object
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    object
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    object
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object
 11  HeartDisease    918 non-null    int64
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

Come si può osservare dall'immagine, non sono presenti valori nulli per nessuna delle feature. E' bene cominciare con un processo di sistemazione del dataset e data cleaning per assicurare la qualità dei dati. Tipicamente questo processo si compone della gestione dei:

- duplicati
- valori mancanti
- outlier
- tipologia di dato

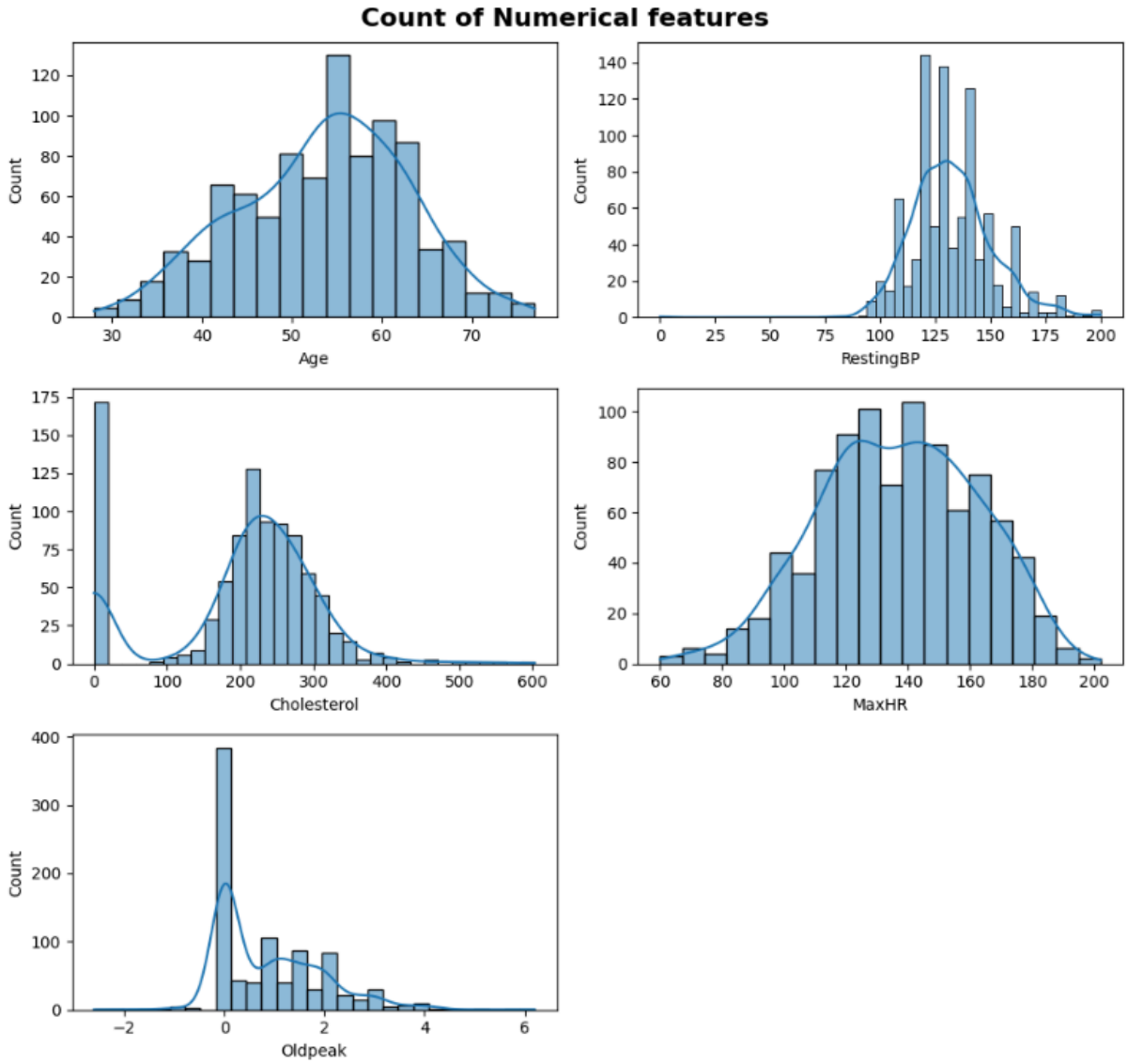
In merito ai valori mancanti, si osserva che non sono presenti; i duplicati sono stati eliminati e gli outlier non sono stati gestiti in quanto il numero di istanze è piuttosto ridotto e dalla analisi di distribuzione che si osserva successivamente non sono particolarmente significativi.
Per quanto riguarda la tipologia di dato si è pensato di effettuare un cast della variabile target "HeartDisease" a bool in quanto la rappresentazione dei valori 0 e 1 ha la presenza o meno del disturbo cardiaco come semantica.
Inoltre i dati di tipo 'object' sono variabili categoriche che per renderli adatti alle analisi e applicazione dei modelli successivamente, sono stati codificati con OneHotEncoder messo a disposizione da scikit-learn, dopo aver osservato che ciascuna di queste possedeva un

numero ridotto di valori unici. OneHotEncoder prende una feature (colonna del dataframe) e ne crea una nuova per ciascun valore unico che si osserva della feature in questione, assegnando il valore 1 se l'istanza aveva quel valore nel record originale e 0 altrimenti.

Ora che il dataset è stato "pulito", si osservi la distribuzione dei dati.

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

Notiamo che tutte le feature hanno media e deviazione standard differenti. HeartDisease come reso noto in precedenza è la variabile target e ha valori 0 ed 1.
I grafici delle distribuzioni possono rendere più immediata la comprensione delle feature numeriche.

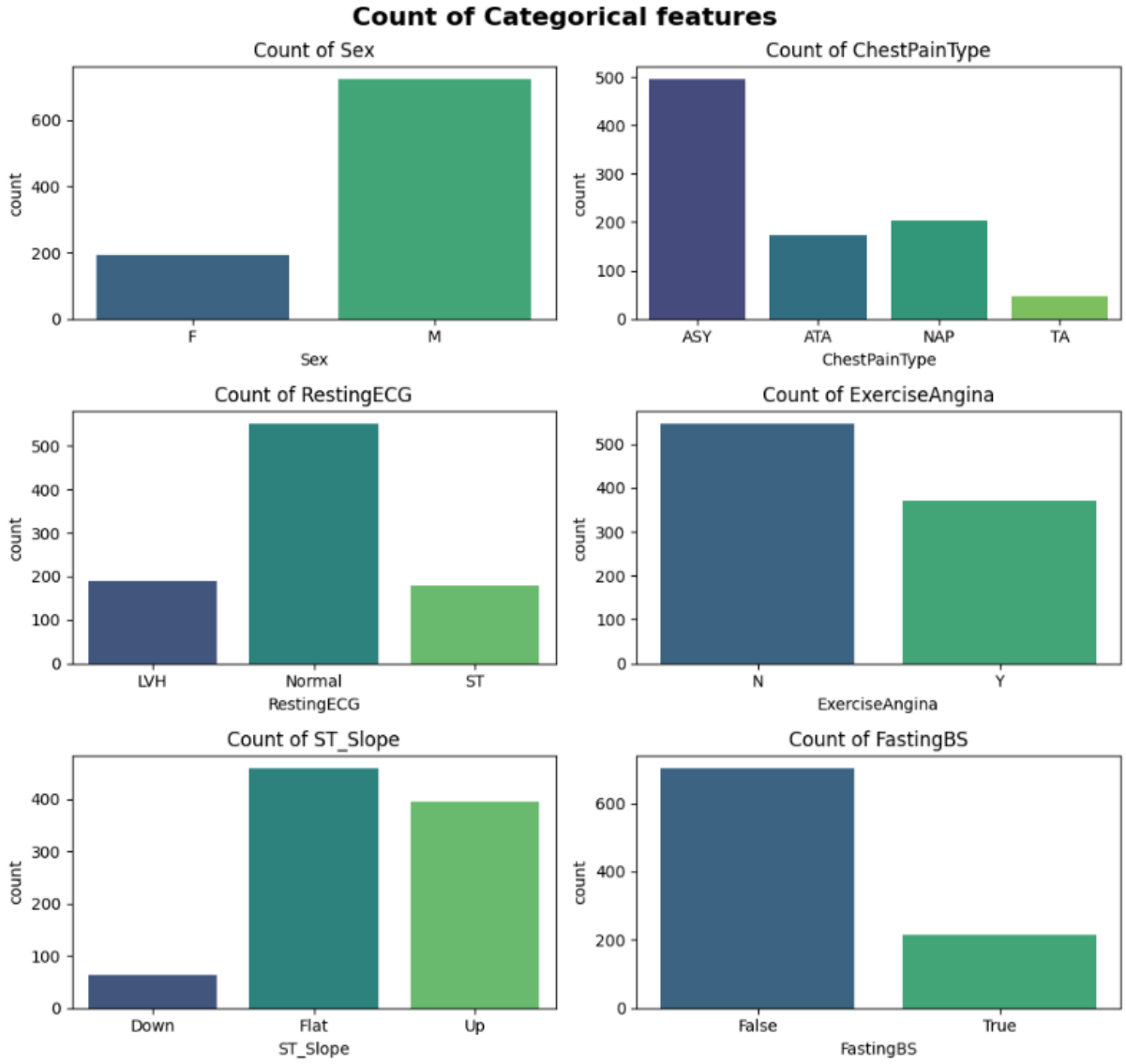


La distribuzione dell'età (Age) è approssimativamente normale, con la maggior parte dei pazienti tra i 47 e i 60 anni. La presenza di code leggere su entrambi i lati indica una buona rappresentatività anche delle fasce più giovani e più anziane.
La pressione sanguigna a riposo (RestingBP) mostra una distribuzione asimmetrica con un picco tra 120 e 140 mmHg.
Si noti che in Cholesterol molti valori sono a zero, visto che è pressoché impossibile che un paziente abbia valori a zero di colesterolo, quei valori rappresentano valori mancanti nonostante non siano risultati dall'esplorazione iniziale. Per risolvere questo problema di data quality si è pensato di inserire al posto dei valori nulli, la media dei valori non nulli del dataset. Dato che la curva rappresentata dai valori non nulli segue circa una distribuzione gaussiana, la media può essere una buona approssimazione del valore.
La feature “MaxHR” relativa alla frequenza cardiaca massima registrata mostra una distribuzione quasi normale con un picco intorno ai 125-140 bpm.
Distribuzione fortemente asimmetrica nella feature “Oldpeak” con la maggior parte dei valori vicini allo 0. Valori superiori a 2 potrebbero essere indicativi di un ischemia significativa.

	Age	RestingBP	Cholesterol	MaxHR	Oldpeak
kurtosis	-0.386140	3.271251	6.257803	-0.448248	1.203064
skew	-0.195933	0.179839	1.373396	-0.144359	1.022872

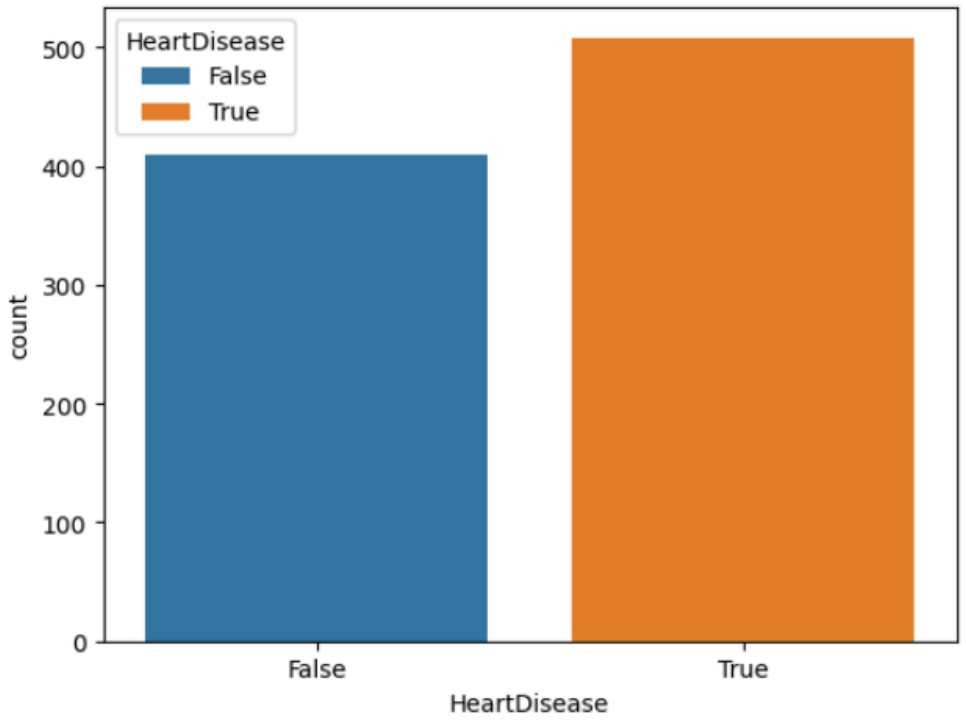
Dando un'occhiata ai valori di curtosi e asimmetria, le osservazioni precedenti sono confermate, una "campana" molto appuntita (leptocurtica) per le covariate RestingBP e Cholesterol e una asimmetria della curva accentuata in Cholesterol e Oldpeak positivamente che indica curve lunghe a destra delle medie rispettive.

Analizziamo invece le variabili categoriche e osserviamo come sono distribuite nel dataset



Nel dataset è evidente una predominanza maschile: circa 700 pazienti sono uomini, mentre le donne sono circa 200. Questo sbilanciamento potrebbe influenzare le previsioni del modello, soprattutto se il sesso risulta essere una variabile rilevante.
Analizzando la tipologia di dolore toracico, si nota che la categoria **ASY** (asintomatico) è la più frequente, seguita da **NAP** (dolore atipico) e **ATA** (angina tipica). La categoria **TA** (angina tipica) appare invece piuttosto rara.

Osservando la distribuzione dei risultati dell'elettrocardiogramma a riposo, circa la metà dei pazienti mostra un tracciato **normale**, mentre le categorie **ST** e **LVH** sono meno frequenti ma rappresentate in modo simile.
La variabile che indica la presenza di angina da sforzo mostra che la maggior parte dei pazienti non ne soffre (**N**), mentre circa il 40% presenta angina indotta dall'esercizio (**Y**).
Infine, analizzando la pendenza del segmento ST, si nota una leggera prevalenza della categoria **Flat**, seguita da **Up**, mentre **Down** risulta meno comune.
Variabile binaria "FastingBS" presenta la maggior parte dei valori a 0 (glicemia normale) e una minoranza a 1 (glicemia elevata).

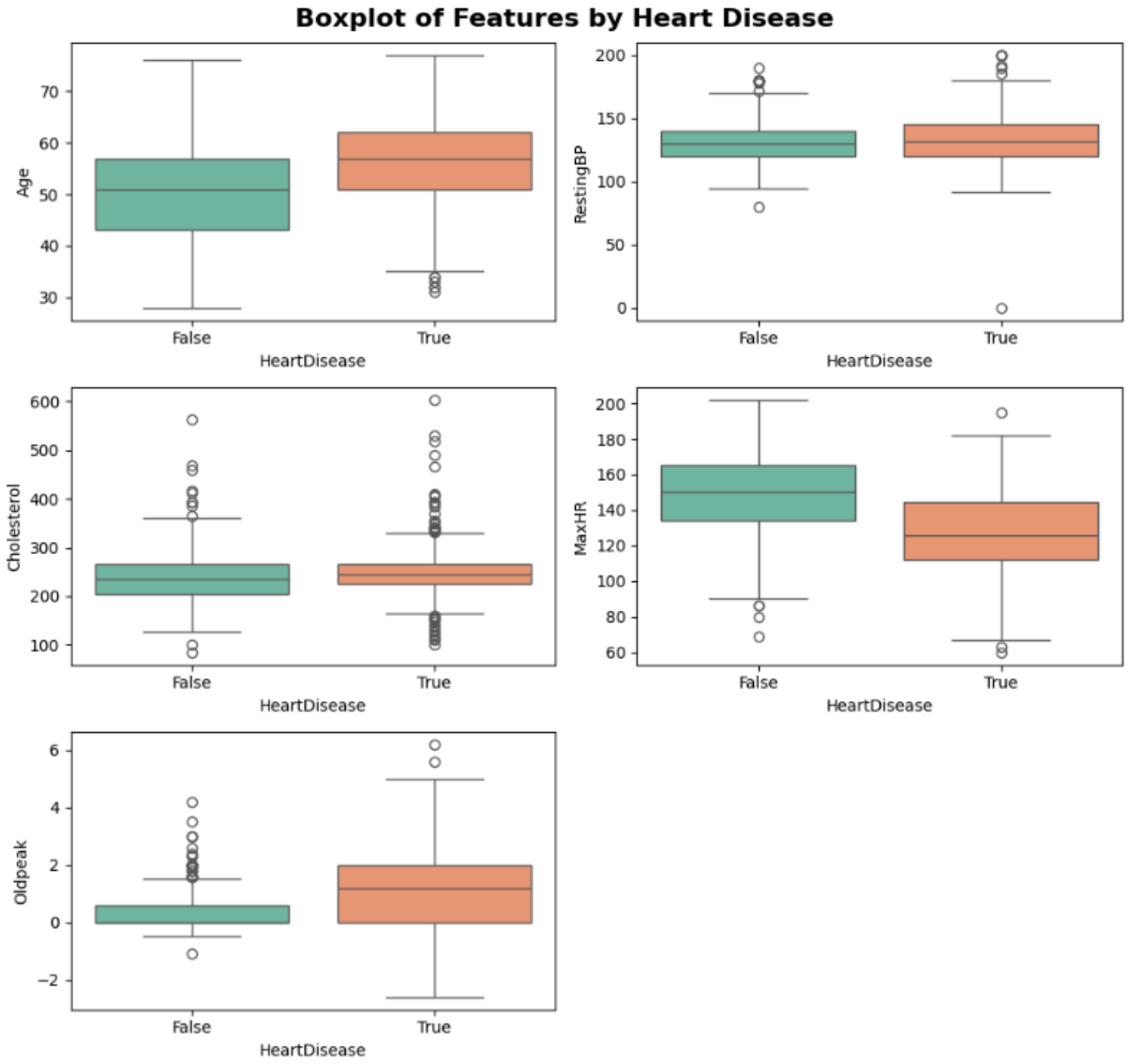


Il target presenta una distribuzione equilibrata tra i casi positivi e negativi di malattie cardiache, con le seguenti percentuali:

- **True:** 55,34%
- **False:** 44,66%

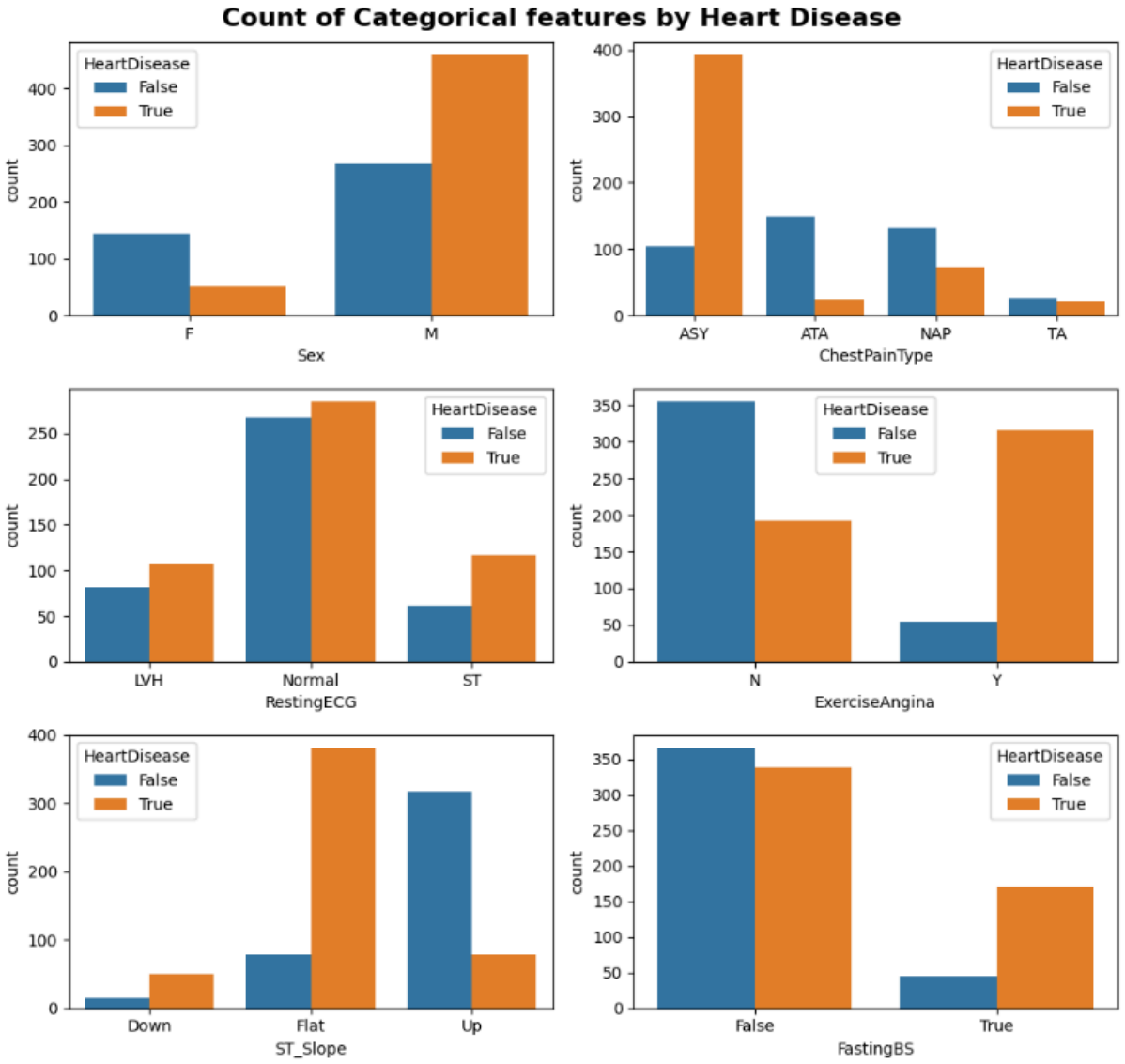
In relazione ai modelli di machine learning descritti nei capitoli successivi, un modello triviale utilizzato come baseline — che assegna sempre il valore *True* a ogni istanza — otterrebbe un'accuratezza pari al **55,34%**

Analisi rispetto al target
Invece che osservare la distribuzione che assumono le singole variabili, proviamo ad osservarne la distribuzione rispetto al valore target "HeartDisease". Visualizziamo le variabili continue attraverso dei boxplot.



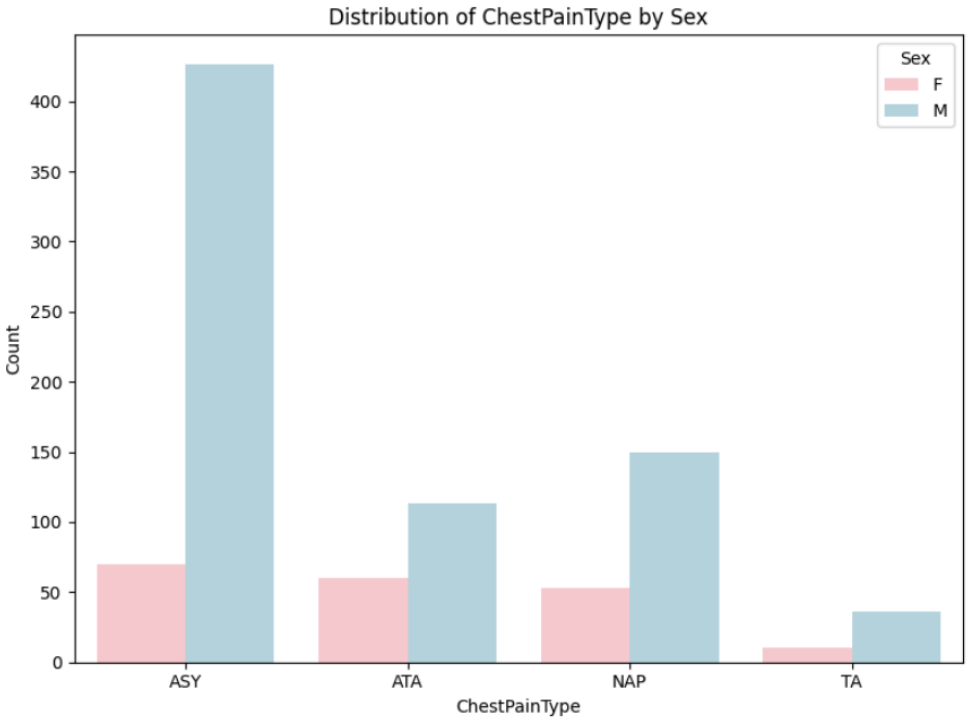
I grafici evidenziano fatti significativi.
L'età dei pazienti che presentano un HeartDisease è leggermente maggiore di quella dei pazienti sani, con qualche caso particolare tra i 30 e i 40 anni. I soggetti positivi mostrano una massima frequenza cardiaca raggiunta leggermente inferiore a quella dei soggetti sani e un "Oldpeak" (dunque Depressione ST nell'ECG) con una curtosi inferiore dei soggetti sani e dunque una distribuzione più ampia. Le variabili coinvolte in queste osservazioni potrebbero essere significative ai fini delle predizioni.

Si osservi ora se emergono nuove informazioni rilevanti dalle feature categoriche.



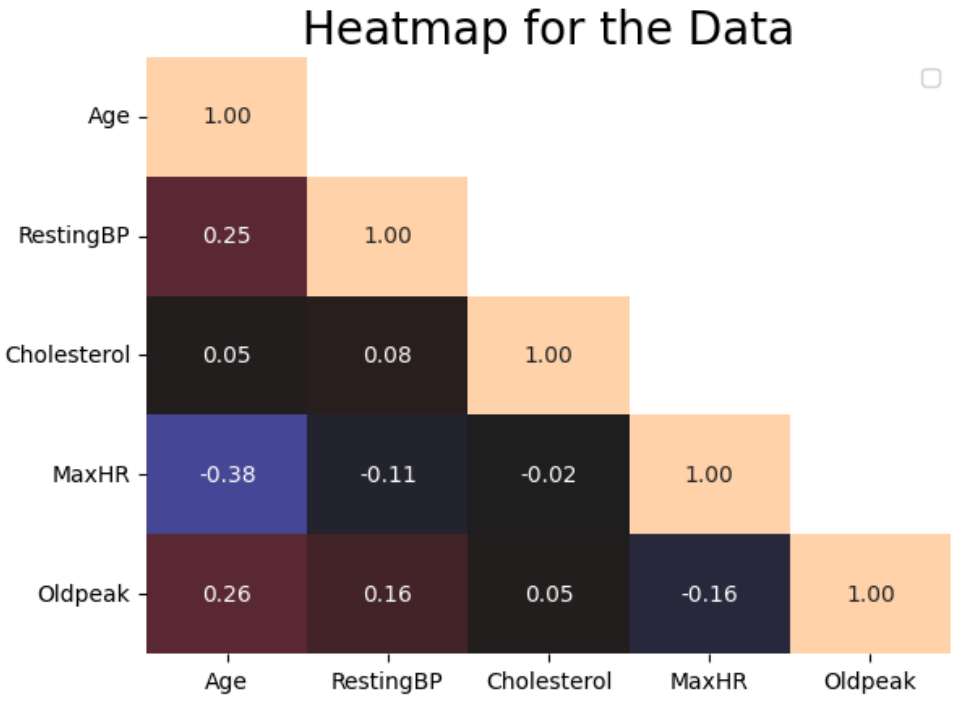
“ChestPain” evidenzia come sia molto probabile che chi ha “HeartDisease” non abbia dolore toracico e che sia appunto ASYmptomatic. Gli ECG a riposo possono essere significativi per fare predizioni in merito a un disturbo cardiovascolare se presentano anomalie di picco (ST). E’ evidente un forte sbilanciamento nella composizione del dataset in cui le istanze “malate” sono in forte prevalenza maschili, questo potrebbe indurre la presenza di bias nei modelli e una rappresentazione inesatta del dominio. Molto significativa invece la feature di ExerciseAngina che è molto probabile che sia Yes se il soggetto in questione ha HeartDisease (ma non è altrettanto probabile che sia a No se Il soggetto è sano). I valori Flat e Down per la feature ST_Slope sono piuttosto rilevanti per identificare il disturbo nonostante occorrano con frequenza significativamente diversa.

Viene spontaneo chiedersi se donne e uomini riscontrino dolori al petto con una distribuzione differente.



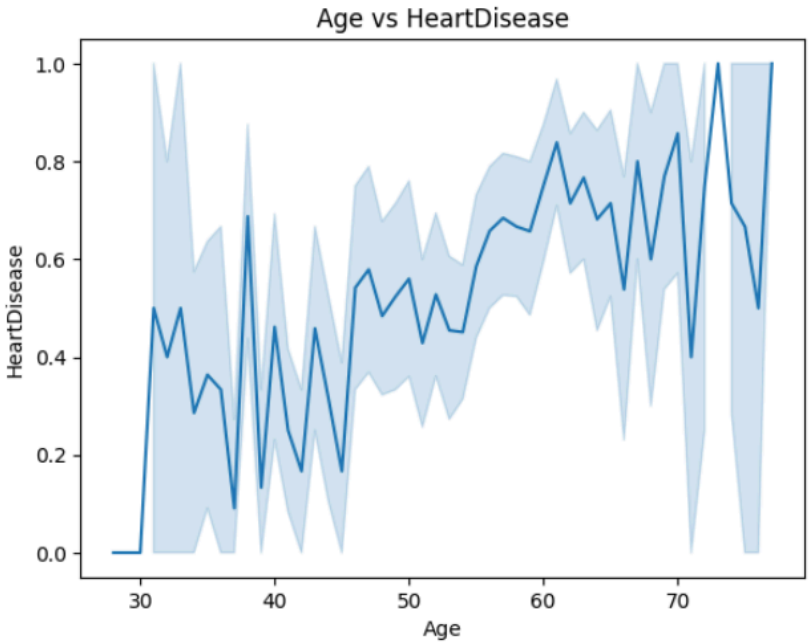
Dal grafico emerge che i soggetti di sesso maschile siano prevalente asintomatici per quanto riguarda il dolore al petto, mentre per le donne anche probabilmente per natura anatomica presentino una distribuzione delle tipologie di dolore al petto più omogenea, nonostante la tipologia leggermente dominante sia sempre la categoria Asyntomatic.

Analisi di correlazione
Ricerchiamo ora delle possibili correlazioni tra le variabili numeriche.



Non sono presenti correlazioni significative. Un accenno di correlazione negativa si individua tra le feature MaxHR e Age. E’ intuitivo pensare che ad un'età più elevata la frequenza massima raggiungibile dal cuore si riduca leggermente.

Inoltre il grafico successivo che può rappresentare il legame che unisce Age e HeartDisease può evidenziare che il rischio di contrarre un disturbo cardiaco può aumentare con l'avanzare dell'età.



E' probabile che la varianza del grafico si vada ad appiattire in un dataset più ampio.

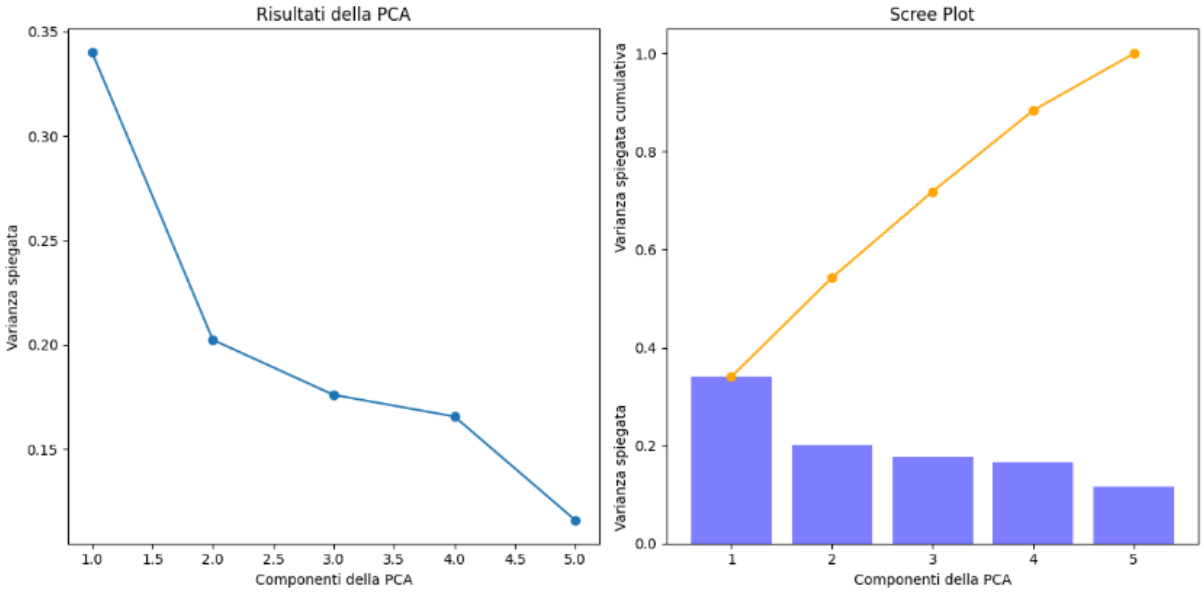
Principal Component Analysis

Il Dataset ottenuto finora contiene 5 feature continue, Si può provare ad effettuare una analisi esplorativa con Principal Component Analysis.

Nella sezione dedicata alla PCA, l'obiettivo era ridurre la dimensionalità dei dati mantenendo la maggior parte dell'informazione presente. Il processo è stato eseguito nei seguenti passaggi:

1. **Preprocessing dei dati:**
   È stata effettuata una standardizzazione delle feature numeriche mediante lo StandarScaler, in modo che tutte le variabili avessero media zero e varianza unitaria. Questo passaggio è cruciale per evitare che variabili con scale differenti influiscono in modo sproporzionato sulla PCA.
2. **Applicazione della PCA:**
   Dopo la standardizzazione, la PCA è stata calcolata sul set di dati normalizzato. È stata considerata l'intera gamma di componenti principali per valutare la varianza spiegata da ciascuna.
3. **Analisi della varianza spiegata:**
   - È stato generato un grafico che mostra la **varianza spiegata** da ciascuna componente principale. Questo aiuta a identificare quante componenti sono necessarie per catturare una percentuale significativa dell'informazione originale.

- È stato creato anche lo **scree plot**, che mostra sia la varianza spiegata singola sia quella cumulativa. Questo grafico è utile per individuare il cosiddetto "gomito" (elbow), che suggerisce il numero ottimale di componenti da mantenere.

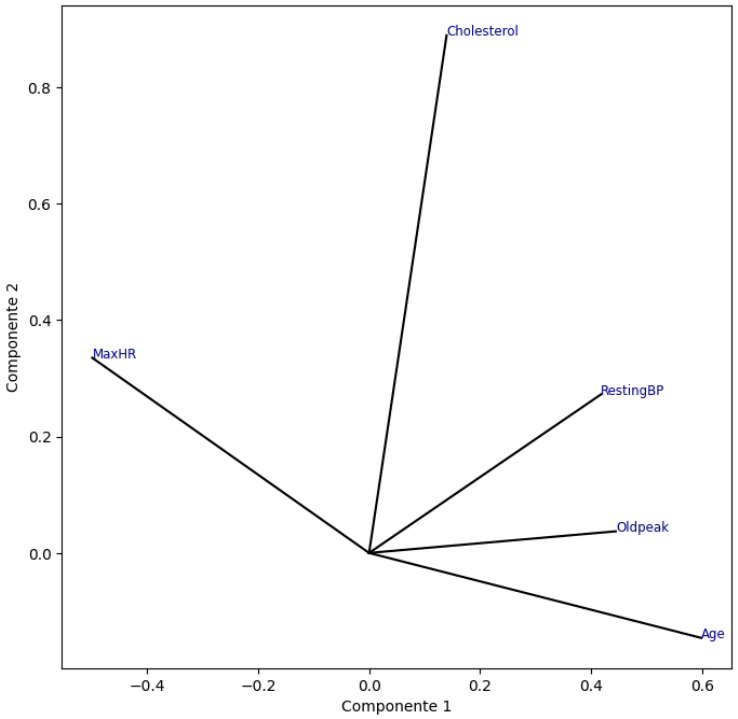


	Eigenvalue	Variance Percent	Cumulative Variance Percent
Component 1	1.702267	34.008252	34.008252
Component 2	1.012787	20.233682	54.241934
Component 3	0.880805	17.596920	71.838853
Component 4	0.829078	16.563507	88.402360
Component 5	0.580514	11.597640	100.000000

Risultati principali

- La varianza spiegata dalle prime componenti principali ha mostrato che una piccola parte di esse è sufficiente a mantenere la maggior parte dell'informazione.
- Dal grafico della varianza cumulativa, si è osservato che con 4 componenti si raggiungeva una soglia di varianza spiegata superiore al **88%**, suggerendo una riduzione della dimensionalità senza perdita rilevante di informazione.

E' stato creato inoltre un grafico 2D che mostra le correlazioni tra le variabili iniziali e le prime due componenti principali. Ciascun vettore consente di ricavare diverse informazioni: due vettori correlati positivamente sono raggruppati insieme, mentre in caso di correlazione negativa si trovano in quadranti opposti, e infine è possibile quantificare la qualità di una singola feature in base alla distanza dall'origine.



Cholesterol, per esempio, è molto correlata con la componente principale 2 (asse x) e per nulla correlata con la componente principale 2, il chè può significare che PC1 e Cholesterol rappresentano all'incirca una stessa caratteristica delle istanze.

Modelli applicati

Per l'applicazione dei modelli di Machine Learning si è utilizzato il dataset originale (non quello ridotto con l'applicazione di PCA), con l'opportuna pulizia e trasformazione, descritta in precedenza.

Per la fase di cross validation, è stato utilizzato il training set a causa del numero limitato di istanze disponibili. Questa scelta è stata fatta per sfruttare al massimo i dati disponibili e ottenere comunque una stima affidabile delle prestazioni del modello. Tuttavia, in presenza di un dataset più ampio, sarebbe stato preferibile utilizzare un validation set separato per evitare possibili bias e garantire una valutazione più robusta e generalizzabile del modello.

Sono stati quindi definiti i set di dati da utilizzare per l'addestramento (e validation) e test separando opportunamente insieme di feature e la variabile target "HeartDisease", con una proporzione di 80% training e 20% test.

I modelli applicati sono alberi di decisione, support vector machines e reti neurali. Prima di costruire gli ultimi due modelli è stata applicata la standardizzazione con StandardScaler, in quanto questi risultano essere scale-sensitive.

Alberi di decisione

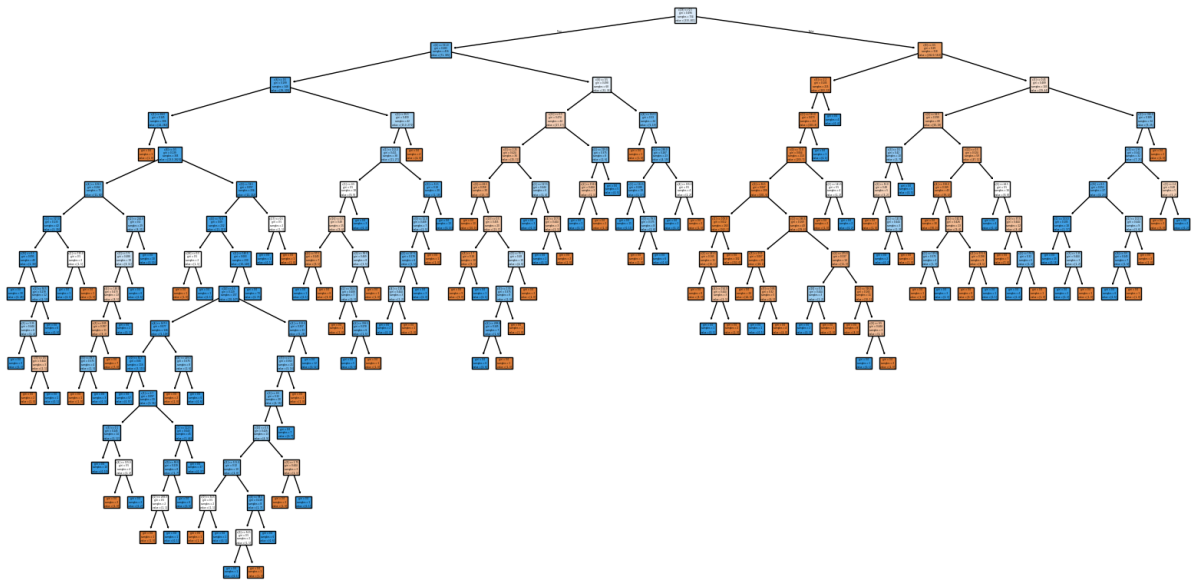
Come primo modello è stato scelto l'albero di decisione, poiché si presta particolarmente bene alla classificazione binaria. Tra i principali vantaggi di questo modello vi sono la semplicità, l'efficienza e l'interpretabilità dei risultati. Quest'ultimo aspetto è particolarmente rilevante nel contesto medico, in quanto permette di comprendere le motivazioni che portano a classificare un paziente in una determinata categoria, facilitando così l'analisi e l'interpretazione dei risultati.

In particolare sono stati costruiti 2 alberi di decisione:

- **Un albero naive**: con un approccio semplice ed immediato, utilizzando iperparametri di default per la creazione dell'albero, dunque non ponendo alcun limite rispetto alle dimensioni dell'albero in termini di nodi e profondità
- **Un albero ottimizzato**: Per ottimizzare la costruzione dell'albero di decisione, è stata utilizzata la Grid Search per selezionare i migliori iperparametri. In particolare, sono stati valutati diversi valori di ccp_alpha, splitter, criterion, max_depth e max_features, con l'obiettivo di massimizzare l'accuracy sul test. Questo approccio consente di identificare la combinazione ottimale di parametri, migliorando le prestazioni del modello rispetto a una configurazione predefinita.

Costruzione e Training con Approccio Naive

Approfondendo l'approccio naive, l'albero è stato costruito utilizzando la libreria sklearn nel modo più semplice possibile, ovvero lasciando gli iperparametri al valore di default.



L'addestramento del modello è stato eseguito utilizzando la funzione fit() di scikit-learn, applicata ai dati precedentemente suddivisi in training e test. A fine esecuzione possiamo osservare la struttura finale dell'albero che ha una profondità massima di 17. Inoltre è stata calcolata anche l'accuracy di training:

```
Accuracy di training: 1.0000
```

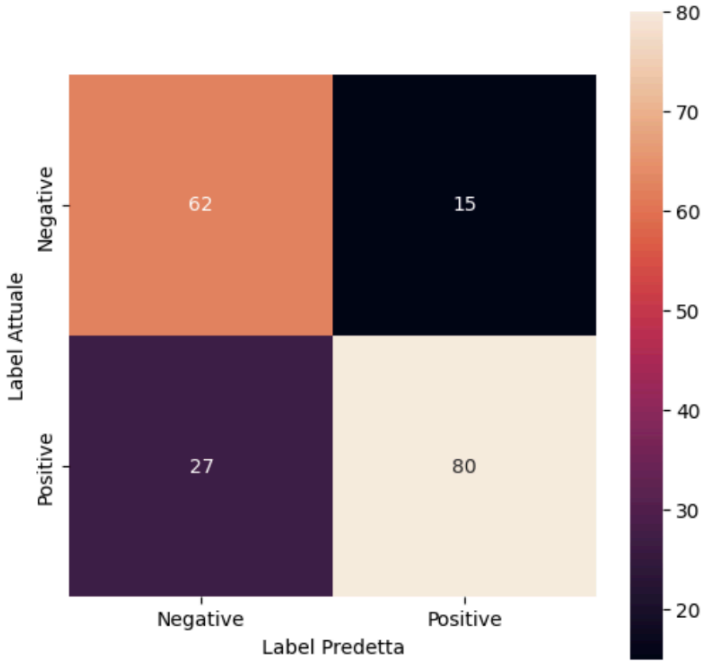
```
tempo training albero di decisione naive in secondi: 0.0049
```

Valutazione delle Performance Naive

Dopo l'addestramento si è andato a valutare il modello, effettuando predizioni e visualizzando misure e grafici delle performance.

La matrice di confusione è una tabella utilizzata per valutare le prestazioni di un modello di classificazione. Essa confronta le predizioni del modello con i valori reali, mostrando quanti campioni sono stati classificati correttamente e quanti sono stati erroneamente assegnati ad altre classi. Per un problema di classificazione binaria, la matrice di confusione è strutturata come segue:

	Predetto Positivo	**Predetto Negativo**
Reale Positivo	Vero Positivo (TP)	Falso Negativo (FN)
Reale Negativo	Falso Positivo (FP)	Vero Negativo (TN)



Misure di performance

Sono state calcolate ulteriori misure di performance per una valutazione più dettagliata:

Misure di performance per ogni classe target: misure per il target che forniscono informazioni specifiche sulle prestazioni del modello per ciascuna classe di destinazione.

- **Precision**: misura la proporzione di istanze identificate correttamente come appartenenti a una determinata classe rispetto a tutte le istanze identificate come appartenenti a quella classe. È calcolata come il rapporto tra i veri positivi (TP) e la somma dei veri positivi e dei falsi positivi (FP).
- **Recall**: misura la proporzione di istanze di una determinata classe che sono state identificate correttamente dal modello rispetto a tutte le istanze effettivamente appartenenti a quella classe. È calcolato come il rapporto tra i veri positivi (TP) e la somma dei veri positivi e dei falsi negativi (FN).
- **F1-score**: è la media armonica di precision e recall. L'F1-score è calcolato come il rapporto tra due volte il prodotto di precision e recall e la loro somma
- **Support**: rappresenta il numero di campioni di test che appartengono a ciascuna classe target.

```
misure di performance albero di decisione a livello di classe:

              precision    recall  f1-score   support

       False       0.70      0.81      0.75        77
        True       0.84      0.76      0.80       107

    accuracy                           0.78       184
   macro avg       0.77      0.78      0.77       184
weighted avg       0.79      0.78      0.78       184
```

Misure di performance globali: misure che forniscono una valutazione complessiva delle prestazioni del modello sull'intero insieme di dati di test.

```
Misure di performance globali albero di decisione

Accuracy: 0.7771739130434783
Precision: 0.84375
Recall: 0.7570093457943925
F1-score: 0.7980295566502463
```

Curva ROC e AUC

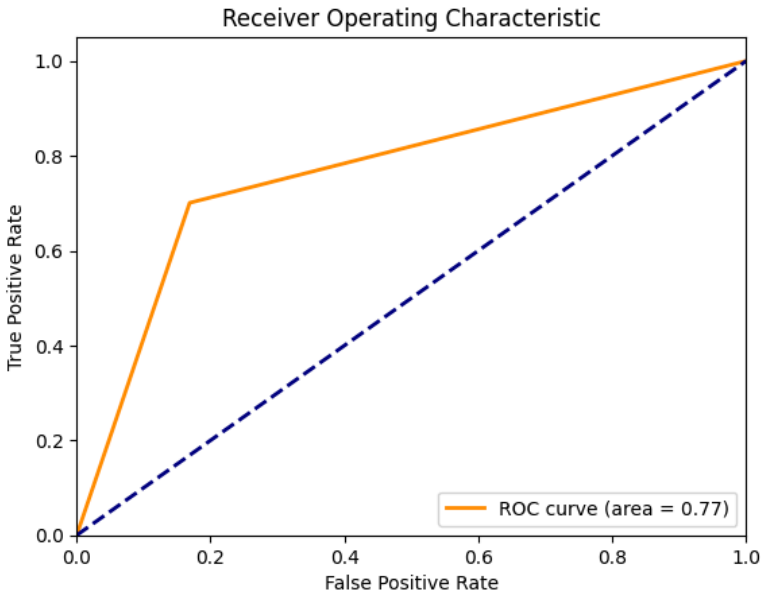
La curva ROC è un grafico che rappresenta le prestazioni di un classificatore binario al variare della soglia di decisione. Sull'asse x si trova il tasso di falsi positivi (FPR), mentre sull'asse y si trova il tasso di veri positivi (TPR), noto anche come recall o sensibilità.

Una curva ROC più vicina all'angolo in alto a sinistra indica un modello più performante, poiché implica un alto tasso di veri positivi e un basso tasso di falsi positivi.

AUC

L'AUC (Area Under the Curve) è un valore numerico che rappresenta l'area sotto la curva ROC. Indica la capacità del modello di distinguere tra le classi:

- AUC = 1 → il modello distingue perfettamente tra le due classi.
- AUC = 0.5 → il modello è equivalente a una classificazione casuale.
- AUC < 0.5 → il modello è peggiore di una scelta casuale.



La curva ROC ottenuta dal modello naive ha prodotto un AUC di 0.77, un valore che indica una buona capacità discriminante. Sebbene non sia ottimale, è comunque significativamente superiore al valore casuale (0.5) e discretamente vicino al massimo teorico (1.0), suggerendo che il modello è in grado di distinguere le due classi in maniera discreta.

Costruzione con Approccio ottimale

Per ottimizzare l'albero di decisione tramite la scelta dei migliori iperparametri abbiamo utilizzato grid search, in particolare abbiamo scelto la funzione GridSearchCV della libreria sklearn. Per prima cosa abbiamo definito un nuovo modello di albero di decisione in maniera analoga a quello definito nell'approccio naive. Successivamente abbiamo identificato gli iperparametri da ottimizzare:

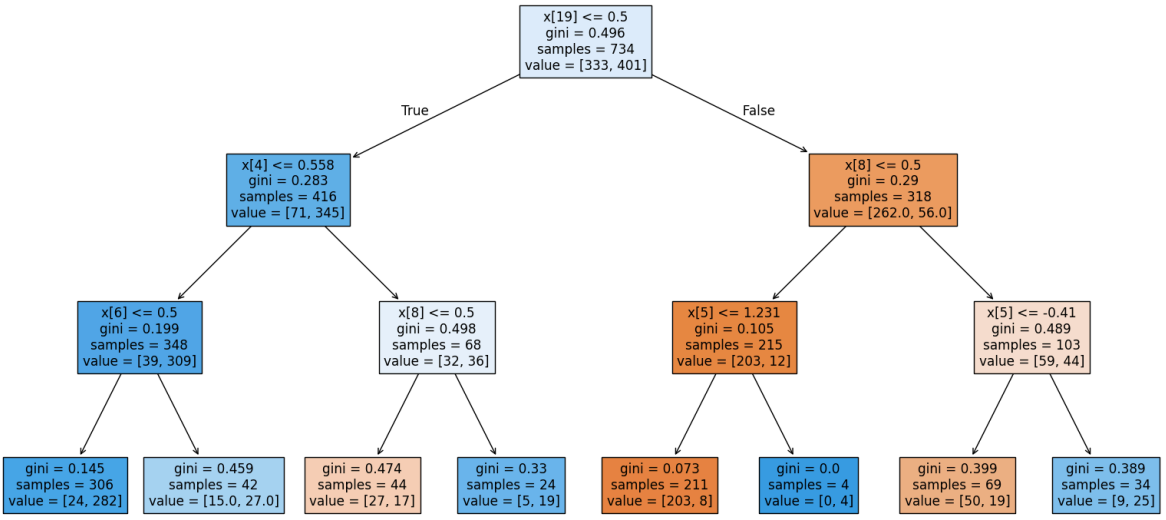
- ccp_alpha: parametro di complessità per la potatura dell'albero. Un valore elevato tende a ridurre l'albero alla sola radice, mentre un valore pari a 0 consente la crescita senza restrizioni.
- criterion: funzione per misurare la qualità dello split (gini, entropy)
- max_depth: la profondità massima che può raggiungere l'albero
- max_features: numero di feature utilizzate per trovare il migliore split
- splitter: la politica utilizzata per scegliere il miglior split ad ogni nodo

risultati ottenuti:

```
Migliori parametri: {'ccp_alpha': 0.0, 'criterion': 'gini', 'max_depth': 3, 'max_features': None, 'splitter': 'best'}
```

```
tempo calcolo iperparametri ottimali con grid search in secondi: 7.0188
```

Una volta trovato gli iperparametri ottimali procedo con la costruzione e il training dell'albero di decisione




```
Accuracy di training: 0.8856
```

```
tempo training albero di decisione ottimale in secondi: 0.0034
```

Notiamo che l'accuracy sul training set è diminuita rispetto al modello naive, ciò significa che probabilmente l'overfitting si è ridotto. Dall'albero si conferma quanto ottenuto utilizzando gli iperparametri restituiti dalla Grid Search (per esempio la profondità massima di 5)

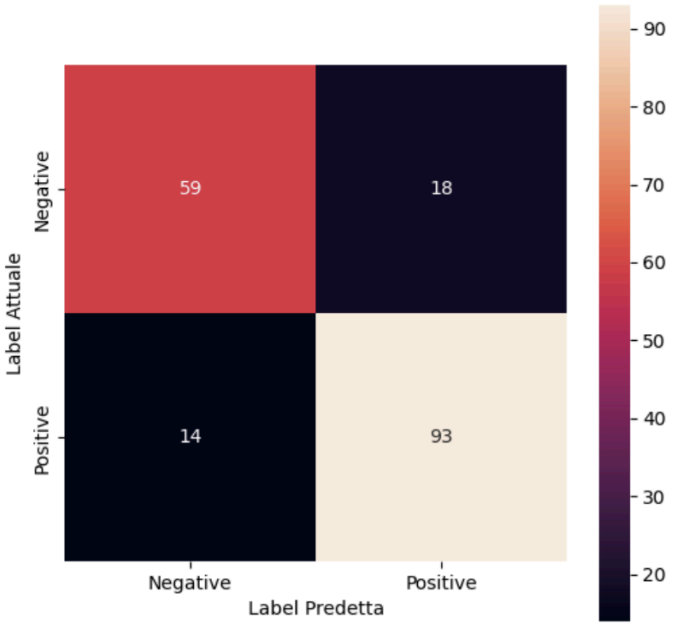
Valutazione performance con albero con approccio ottimale

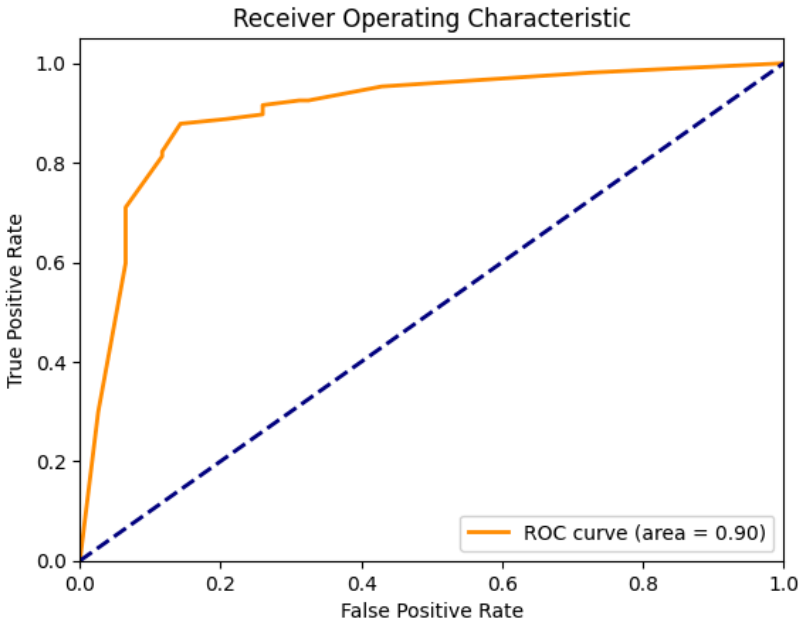
Dopo l'addestramento si è andato a valutare il modello, effettuando predizioni e visualizzando misure e grafici delle performance.

Misure di performance per ogni classe target:

```
Misure di performance albero di decisione ottimizzato a livello di classe:
              precision    recall  f1-score   support

       False       0.81      0.77      0.79        77
        True       0.84      0.87      0.85       107

    accuracy                           0.83       184
   macro avg       0.82      0.82      0.82       184
weighted avg       0.83      0.83      0.83       184
```

```
Misure di performance globali albero di decisione ottimizzato

Accuracy: 0.8260869565217391
Precision: 0.8378378378378378
Recall: 0.8691588785046729
F1-score: 0.8532110091743119
```





Dall'analisi dei risultati, si osserva che l'albero di decisione ottimizzato ha ottenuto prestazioni complessivamente migliori rispetto al modello naive.

L'albero ottimizzato risulta preferibile in quanto abbiamo riscontrato:

1. Miglioramento di tutte le performance globali misurate e anche di quelle relative alle classi target
2. Migliore area ROC-AUC: passa da 0.77 a 0.90, indicando una capacità superiore nel distinguere tra le classi.
3. Meno overfitting: pur avendo un accuracy di training inferiore rispetto all'approccio naive, il miglioramento sulle metriche di test suggerisce un modello più generalizzabile.

Support Vector Machines

Il secondo modello presentato sono le Support Vector machine. Il modello sembra adatto a questo tipo di problema in quanto può gestire sia relazioni lineari che non lineari grazie all'uso di kernel. SVM massimizza il margine tra le classi e, con gli opportuni parametri, generalizza bene e ha rischio ridotto di overfitting, caratteristica utile con dataset medici complessi. Inoltre, è robusta anche con un numero limitato di campioni, situazione comune nelle diagnosi cardiache.
Sono state sviluppate due diverse architetture di support vector machines:

- **SVM naive**: Un modello di base costruito con un approccio immediato, senza particolari ottimizzazioni, in seguito i dettagli.
- **SVM ottimizzato**: Un modello più sofisticato, in cui gli iperparametri sono stati selezionati per migliorare la capacità di classificazione. Per ottimizzarli, è stata utilizzata la tecnica del grid search, al fine di individuare la combinazione di parametri che migliori le prestazioni complessive del modello.

Costruzione e Training con Approccio Naive

Ecco la descrizione dei parametri scelti manualmente per la costruzione del modello naive:

- **kernel='rbf'**: Utilizza la funzione kernel radiale di base (Radial Basis Function), che permette di catturare relazioni non lineari tra le variabili. È adatto per dati complessi dove le classi non sono linearmente separabili.
- **C=1000000**: Un valore di **C molto elevato** indica che il modello cerca di classificare correttamente ogni punto di training, riducendo gli errori di classificazione sul training set. Questo porta a un margine molto stretto e aumenta il rischio di **overfitting**.

In sintesi, questo modello punta ad **azzerare gli errori di training**, ma potrebbe generalizzare male su dati non visti a causa dell'elevata penalizzazione sugli errori.

Ecco che proprio come si può immaginare il modello addestrato è in overfitting.

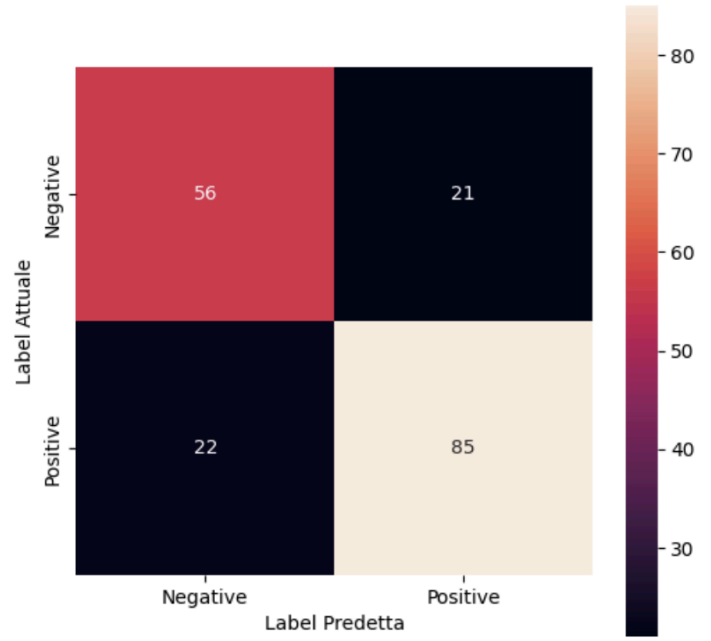
```
Accuracy di training: 1.0000
```

```
tempo training SVM naive in secondi: 0.0325
```

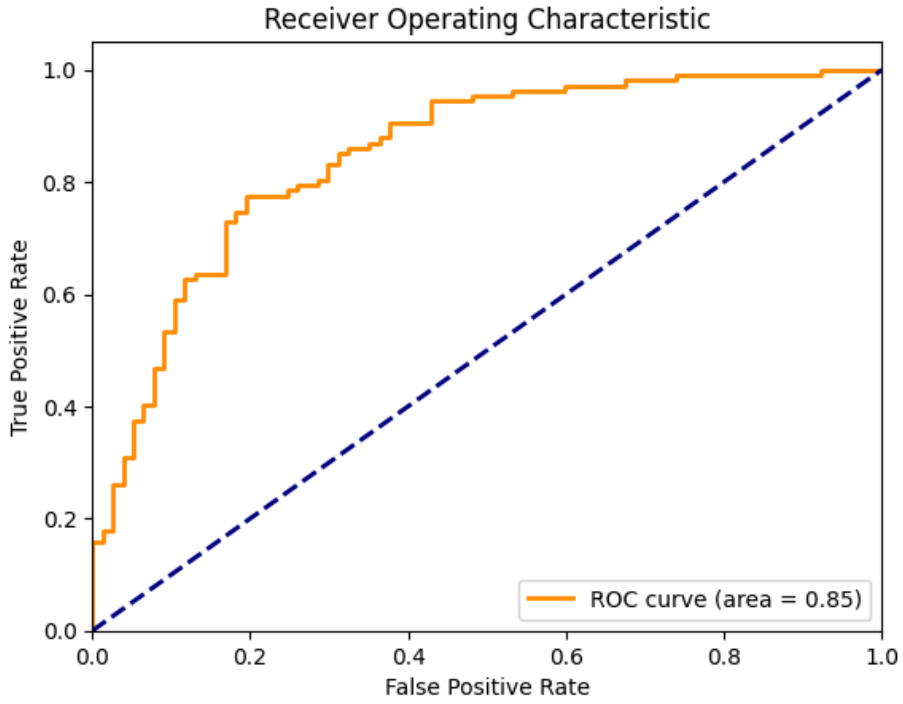
Mentre i risultati in fase di test non sono altrettanto buoni.

```
Misure di performance SVM globali

Accuracy: 0.7663043478260869
Precision: 0.8018867924528302
Recall: 0.794392523364486
F1-score: 0.7981220657276995
```

```
Misure di performance support vector machines a livello di classe:

              precision    recall  f1-score   support

       False       0.72      0.73      0.72        77
        True       0.80      0.79      0.80       107

    accuracy                           0.77       184
   macro avg       0.76      0.76      0.76       184
weighted avg       0.77      0.77      0.77       184
```



Questi risultati indicano prestazioni equilibrate, con precisione e recall abbastanza elevate. Il modello gestisce bene sia i falsi positivi che i falsi negativi. Anche la curva Roc è già molto buona con un AUC pari a 0.85. Possiamo ottenere di meglio da una ottimizzazione opportuna.



Costruzione e Training del modello ottimizzato

Il tuning degli iperparametri è un passaggio cruciale nell'ottimizzazione delle performance di un modello di machine learning, in particolare per modelli complessi come le macchine a vettori di supporto (SVM). In questo contesto, il processo di tuning si concentra sull'ottimizzazione dei principali iperparametri di un SVM, ovvero **C**, **kernel** e **gamma**. Il parametro **gamma** è coefficiente di scala, determina la forma della funzione di kernel (per le funzioni non lineari), influenzando la capacità del modello di adattarsi ai dati. I possibili valori per **kernel** includono 'rbf', 'poly' e 'linear', ciascuno dei quali ha un impatto significativo sulle performance del modello. Utilizzando una **GridSearchCV**, si esegue una ricerca su una griglia di combinazioni di valori per questi iperparametri, al fine di trovare quella che ottimizza l'accuratezza del modello sui dati di addestramento. In questo caso, la funzione di scoring utilizzata è l'accuratezza.

Il modello migliore trovato ha i seguenti iperparametri:

```
Iperparametri ottimali: {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
```

```
tempo calcolo iperparametri ottimali con grid search in secondi: 32.4274
```

Valutazione performance con SVM con approccio ottimale

```
Accuracy di training: 0.9128
```

```
tempo calcolo SVM ottimale in secondi: 0.0184
```

misure di performance di classe:

```
Misure di performance support vector machines ottimizzato a livello di classe:

              precision    recall  f1-score   support

       False       0.83      0.77      0.80        77
        True       0.84      0.89      0.86       107

    accuracy                           0.84       184
   macro avg       0.84      0.83      0.83       184
weighted avg       0.84      0.84      0.84       184
```

Con un'accuratezza di 0.84, il modello si comporta in modo molto buono, classificando correttamente l'84% dei casi.

La media ponderata (weighted average) per precisione, recall e f1-score è 0.84, il che conferma che il modello è equilibrato nel trattare entrambe le classi.

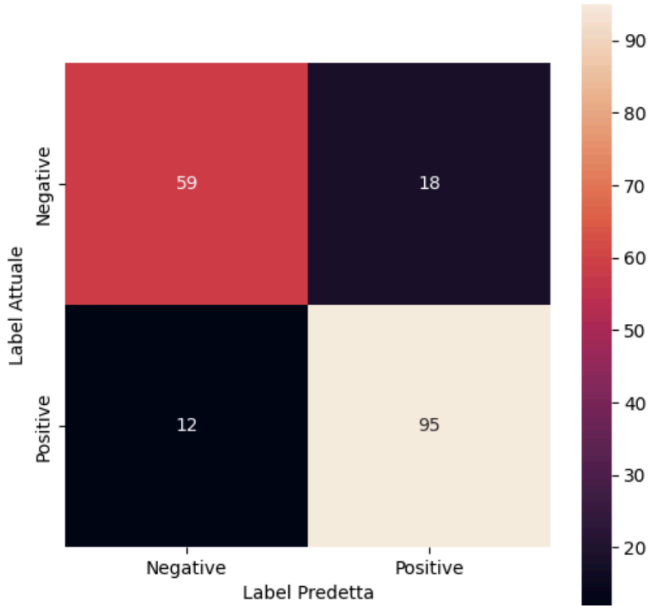
```
Misure di performance SVM globali

Accuracy: 0.8369565217391305
Precision: 0.8407079646017699
Recall: 0.8878504672897196
F1-score: 0.8636363636363636
```

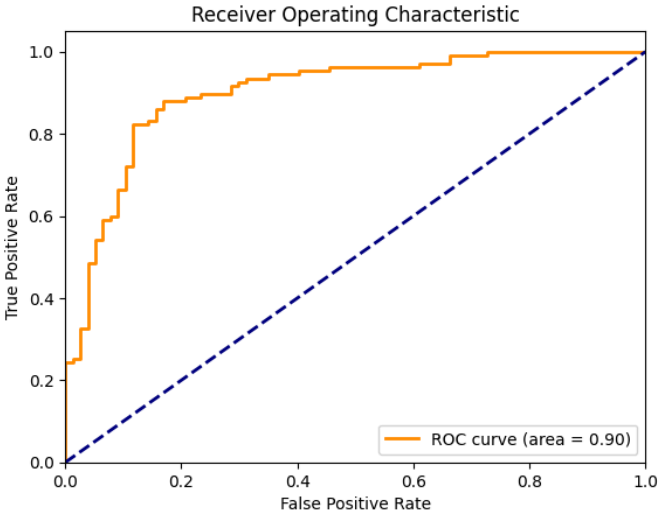
Accuracy: L'accuratezza del modello è **0.837** (circa 84%), il che significa che il modello ha classificato correttamente circa l'84% delle istanze totali. Questo è un buon risultato, indicando che la maggior parte delle previsioni fatte dal modello sono corrette.

Precision: La precisione è **0.841**, il che implica che, tra tutte le istanze classificate come positive (True), circa l'84% sono effettivamente corrette. Questo valore suggerisce che il modello è abbastanza accurato nel fare previsioni positive, riducendo i falsi positivi.

Recall: Il recall è **0.888**, che indica che il modello è in grado di identificare l'88.8% delle istanze positive (True). Questo valore è piuttosto elevato, suggerendo che il modello è particolarmente bravo a trovare i casi positivi, riducendo al minimo i falsi negativi.

F1-score: Il f1-score è **0.864**, che rappresenta una media armonica tra precisione e recall. Questo valore è molto buono, indicando che il modello riesce a mantenere un buon equilibrio tra evitare falsi positivi (precisione) e catturare tutti i casi positivi (recall).





La curva si posiziona ben al di sopra della diagonale blu tratteggiata (linea di riferimento per un classificatore casuale), segnalando che il modello ha buone capacità discriminative.

L'area sotto la curva (AUC) è 0.90, valore elevato che indica un'ottima prestazione complessiva nel distinguere tra le classi.

La curva è ripida all'inizio (vicino allo 0 sull'asse x), suggerendo che il modello raggiunge un alto tasso di veri positivi con pochi falsi positivi, qualità importante in ambito medico.

Anche in questo caso il modello SVM risulta preferibile in quanto abbiamo riscontrato:

1. **Miglioramento di tutte le performance** globali misurate e anche di quelle relative alle classi target
2. **Migliore area ROC-AUC**: passa da 0.85 (vedi codice) a 0.90, indicando una capacità leggermente superiore nel distinguere tra le classi.

3. **Meno overfitting**: pur avendo un accuracy di training inferiore rispetto all'approccio naive, il miglioramento sulle metriche di test suggerisce un modello più generalizzabile.

Reti neurali

l'ultimo modello usato è stato la rete neurale, strumento estremamente versatile in grado di individuare relazioni complesse nei dati. Questo approccio risulta particolarmente utile quando i pattern sottostanti non sono immediatamente evidenti o seguono dinamiche non lineari. Grazie alla sua struttura a più livelli, la rete può apprendere autonomamente le caratteristiche più rilevanti e affinare progressivamente la propria capacità predittiva.

Sono state sviluppate due diverse architetture di rete neurale:

- **Rete naive**: Un modello di base costruito con un approccio immediato, utilizzando gli iperparametri di default senza particolari ottimizzazioni.
- **Rete ottimizzata**: Un modello più sofisticato, in cui gli iperparametri sono stati selezionati per migliorare la capacità di classificazione. Per ottimizzarli, è stata utilizzata la tecnica del grid search, già impiegata negli altri modelli, al fine di individuare la combinazione di parametri che minimizza la loss di validazione e migliora le prestazioni complessive della rete.

Costruzione e Training con Approccio Naive

Per affrontare il problema di classificazione binaria, è stata implementata una rete neurale con Keras.

Prima di addestrare il modello, è stata applicata la standardizzazione ai dati di input, una pratica essenziale nelle reti neurali per garantire una convergenza più stabile e veloce. Per questo scopo, è stato utilizzato lo StandardScaler di Scikit-Learn:

Il modello implementato utilizza la classe sequential di Keras, con la seguente struttura:

- Primo strato nascosto: 100 neuroni con attivazione ReLU
- Secondo strato nascosto: 50 neuroni con attivazione ReLU
- Strato di output: 1 neurone con attivazione sigmoid, adatto alla classificazione binaria

Il modello è stato compilato utilizzando la funzione di perdita binary_crossentropy, adatta a target binari e adam come optimizer.

Il training è stato effettuato utilizzando l'apposita funzione fit offerta da Keras, utilizzando i dati del dataset suddivisi in precedenza. Il batch_size è stato imposto a 32, valore di default se non specificato, mentre il modello è stato allenato su 50 epoche.

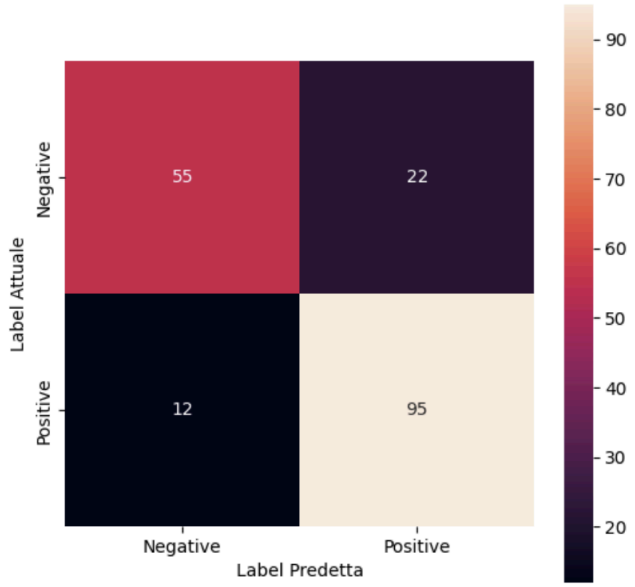
```
Training Loss: 0.0928
Training Accuracy: 0.9755
```

```
tempo training rete neurale naive in secondi: 11.3797
```

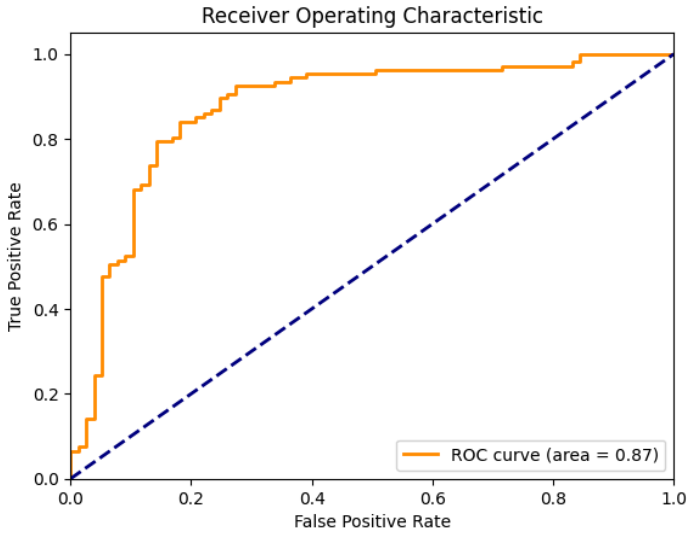
Valutazione performance con rete neurale naive

```
misure di performance rete neurale a livello di classe:

              precision    recall  f1-score   support

       False       0.85      0.73      0.78        77
        True       0.82      0.91      0.86       107

    accuracy                           0.83       184
   macro avg       0.84      0.82      0.82       184
weighted avg       0.83      0.83      0.83       184
```

```
Misure di performance globali rete neurale naive

Accuracy: 0.8478260869565217
Precision: 0.8376068376068376
Recall: 0.9158878504672897
F1-score: 0.875
```




```
Test Loss: 0.6260
Test Accuracy: 0.8261
```



Dall'analisi dei risultati emerge che la rete neurale naïve mostra una elevata accuratezza sul training set (96.87%), ma presenta un Test Loss piuttosto alto (0.6260) rispetto al Training Loss (0.1018). Questo è un chiaro segnale di overfitting, ovvero la rete si adatta molto bene ai dati di addestramento ma fatica a generalizzare su dati nuovi. Inoltre, la differenza tra training e test accuracy conferma questa ipotesi: l'accuratezza sui dati di test è più bassa (82.61%), indicando che la rete potrebbe essersi specializzata troppo sul dataset di training. Tuttavia, le metriche di precisione, recall e F1-score sono comunque buone, con un leggero vantaggio della classe True rispetto alla classe False. La matrice di confusione mostra che la rete commette più errori nel classificare correttamente i falsi negativi. Infine, il valore della AUC-ROC pari a 0.88 è un buon risultato, indicando una discreta capacità di discriminare tra le due classi.

Costruzione e Training con Approccio ottimale

Dopo aver implementato un primo modello di rete neurale con un set di iperparametri scelto manualmente, si è passati a una fase di ottimizzazione più avanzata per migliorare le prestazioni del classificatore. Per ottimizzare gli iperparametri, è stato adottato un approccio basato sulla ricerca esaustiva tramite Grid Search, utilizzando una pipeline che comprendeva la normalizzazione dei dati e l'addestramento della rete neurale. Questa tecnica consente di testare sistematicamente diverse combinazioni di parametri e selezionare la configurazione che garantisce le migliori prestazioni in termini di accuratezza. Gli iperparametri selezionati sono stati:

Funzione di attivazione:
La funzione di attivazione è un componente fondamentale delle reti neurali, in quanto introduce non linearità nel modello, permettendogli di apprendere relazioni complesse tra le variabili di input. Come opzioni nella grid search c’erano tanh e relu

Dimensione del batch:
La dimensione del batch determina il numero di campioni elaborati prima che il modello aggiorni i pesi tramite retropropagazione. Batch più piccoli tendono a introdurre più rumore nella discesa del gradiente, mentre batch troppo grandi possono rallentare il processo di apprendimento e richiedere più memoria. Come opzioni nella grid search c’erano 16 e 32

Numero di epoche
Le epoche rappresentano il numero di volte in cui l'intero dataset viene utilizzato per addestrare la rete neurale. Come opzioni nella grid search c'erano 50 e 100 epoche

Struttura dei layer nascosti:
Questo parametro definisce il numero di neuroni presenti nei layer nascosti della rete. Come opzioni nella grid search c'erano 50 e 100 neuroni

Ottimizzatore:
L'ottimizzatore è l'algoritmo che gestisce l'aggiornamento dei pesi della rete durante l'addestramento. Come opzioni nella grid search c'erano adam e rmsprop

Dropout:
è una tecnica di regolarizzazione utilizzata nelle reti neurali per prevenire l'overfitting e migliorare la capacità di generalizzazione del modello. L'idea principale è disattivare casualmente alcuni neuroni durante l'addestramento, impedendo alla rete di dipendere troppo da specifici percorsi di attivazione e forzandola a imparare rappresentazioni più robuste. Come opzioni nella grid search c'erano 0.2 e 0.3

Iperparametri ottimali

Migliori parametri: {'model__activation': 'relu', 'model__batch_size': 16, 'model__dropout_rate': 0.2, 'model__epochs': 50, 'model__hidden_layer_sizes': (50,), 'model__optimizer': 'adam'}

```
tempo calcolo iperparametri ottimali con grid search in secondi: 834.2395
```

Costruzione e training con approccio ottimale

La costruzione ed il training per la rete ottimale segue lo stesso procedimento della rete naive, ma questa volta vengono utilizzati gli iperparametri ottimali appena ricavati al posto di quelli di default della rete naive.

```
Training Loss: 0.2281
Training Accuracy: 0.9183
```

```
tempo training rete neurale ottimizzata in secondi: 14.8152
```

Rispetto alla rete naïve, abbiamo riscontrato un peggioramento sia della training loss che della training accuracy nel modello ottimizzato. Tuttavia, questo non rappresenta un aspetto negativo, ma piuttosto un'indicazione che la rete naïve soffriva di overfitting. Al contrario, il modello ottimizzato mostra una maggiore capacità di generalizzazione, come confermato dai migliori risultati ottenuti sul test set, riportati di seguito.

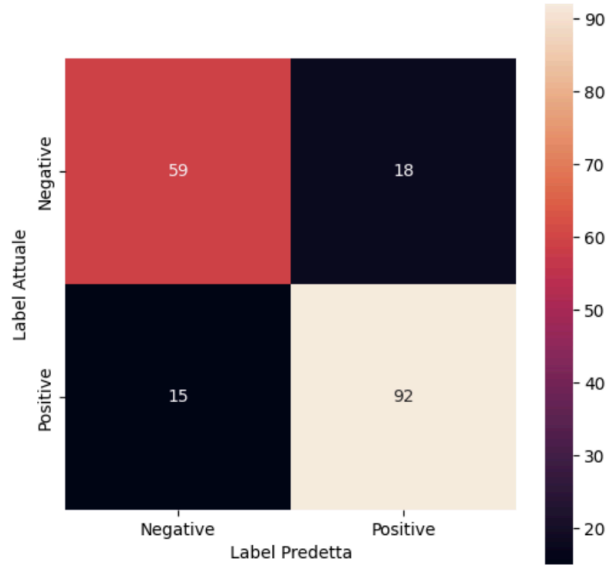
Valutazione performance con rete neurale ottimizzata

misure di performance rete neurale ottimizzata a livello di classe:

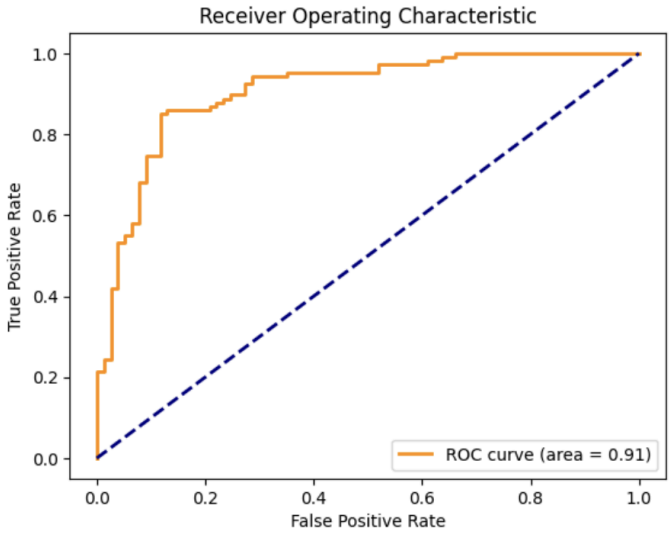
	precision	recall	f1-score	support
False	0.80	0.77	0.78	77
True	0.84	0.86	0.85	107
accuracy			0.82	184
macro avg	0.82	0.81	0.81	184
weighted avg	0.82	0.82	0.82	184

Misure di performance globali rete neurale ottimizzata

Accuracy: 0.8206521739130435
Precision: 0.8363636363636363
Recall: 0.8598130841121495
F1-score: 0.847926267281106



Test Loss: 0.3932
Test Accuracy: 0.8370



la rete neurale ottimizzata presenta un approccio più bilanciato tra training e test, con una Training Accuracy leggermente inferiore (90.74%) rispetto alla rete naïve, ma con una perdita ridotta sui dati di test (0.3932 invece di 0.6260). Questo suggerisce che l'ottimizzazione ha portato a una migliore generalizzazione del modello, riducendo il rischio di overfitting. L'accuratezza sui dati di test è molto simile a quella della rete naïve (82% contro 83%), ma la diminuzione del Test Loss conferma una gestione più efficace degli errori. Anche in questo caso, le metriche di precision, recall e F1-score sono equilibrate, senza variazioni significative tra le due versioni della rete. La matrice di confusione mostra un comportamento simile alla precedente, con una leggera diminuzione dei veri positivi. Tuttavia, un aspetto positivo della rete ottimizzata è che il valore della AUC-ROC è migliorato (0.91 rispetto a 0.87), segno che la capacità di discriminare tra le classi è aumentata.

In conclusione, la rete neurale ottimizzata sembra essere la scelta migliore, in quanto riduce il rischio di overfitting e garantisce una migliore generalizzazione, pur mantenendo metriche di performance molto simili alla rete naïve. Il miglioramento dell'AUC-ROC è un ulteriore indicatore che la rete ha affinato la sua capacità di separare correttamente le classi.

Conclusioni

L'obiettivo di questo progetto era sviluppare modelli di machine learning per prevedere la presenza di disturbi cardiaci basandosi su dati clinici.

Un aspetto critico emerso è lo sbilanciamento del dataset, con una predominanza maschile tra i pazienti e una frequenza maggiore di pazienti con malattia cardiaca. Questo potrebbe aver introdotto bias nei modelli, suggerendo la necessità di un dataset più ricco per ottenere modelli migliori.

L'analisi esplorativa e la PCA hanno fornito preziose informazioni sulla struttura dei dati e sulla rilevanza delle feature. La riduzione dimensionale ha mostrato che è possibile mantenere l'88% della varianza delle feature continue con quattro componenti principali. Nonostante ciò si è scelto di utilizzare il dataset originale per l'addestramento dei modelli.

Sono stati implementati e confrontati tre modelli principali: alberi di decisione, support vector machines (SVM) e reti neurali, ciascuno testato sia in una versione "naive" che ottimizzata tramite la ricerca degli iperparametri.

I modelli analizzati sono ampiamente utilizzati in ambito medico grazie alla loro capacità di supportare diagnosi e previsioni. I Decision Tree sono semplici da interpretare, facilitando la comprensione clinica, ma possono soffrire di overfitting, come abbiamo visto. Le SVM offrono ottime prestazioni con dati complessi e piccoli set di dati, ma sono meno intuitive e richiedono un'attenta scelta dei parametri. Le reti neurali eccellono nella rappresentazione di dati complessi, ma il loro modello "scatola nera" rende difficile spiegare le decisioni, un limite in contesti dove la trasparenza è cruciale.

I risultati hanno mostrato che l'ottimizzazione degli iperparametri migliora sensibilmente le performance dei modelli rispetto alle configurazioni iniziali. In particolare:

- **Alberi di decisione**: L'approccio ottimizzato ha evidenziato un miglioramento nella generalizzazione rispetto al modello naive, con una riduzione dell'overfitting. L'algoritmo ottimizzato ha raggiunto un accuracy dell'83% circa, e un AUC di 0.90 dimostrando precisione praticamente sovrapponibile a quella di SVM. Oltre alla buona precisione l'albero di decisione ha avuto tempi di esecuzione molto bassi confermandosi un modello molto efficiente
- **Support Vector Machines**: Il modello ottimizzato ha raggiunto un'accuratezza dell'84% e un AUC di 0.90, mostrando ottime capacità discriminative. La capacità di catturare relazioni non lineari grazie all'utilizzo del kernel RBF si è dimostrata particolarmente vantaggiosa. Questo modello ha unito buona precisione a tempi di esecuzione molto contenuti dimostrandosi un'ottima soluzione per l'analisi di questo dataset
- **Reti neurali**: La rete ottimizzata ha evidenziato una migliore capacità di generalizzazione rispetto al modello naive, mantenendo le metriche di performance simile alle rete neurale naive. Rispetto agli altri due modelli le reti neurali hanno evidenziato un tempo di addestramento e di ricerca di iperparametri ottimali di gran lunga superiori. Avendo riscontrato precisione simile ai precedenti due modelli la rete neurale non si è rivelato il modello migliore per l'analisi di questo dataset a causa dei suoi elevati tempi di esecuzione

In conclusione, tutti i modelli ottimizzati hanno fornito risultati soddisfacenti, con le SVM e l'albero di decisione che si sono dimostrati particolarmente efficaci in termini di accuratezza e capacità discriminativa a fronte di tempo di calcolo molto brevi. L'utilizzo di tecniche di machine learning in ambito medico, come dimostrato in questo progetto, può supportare i professionisti nella diagnosi precoce, migliorando la prevenzione e l'efficacia dei trattamenti.