

# **HeartNeuroSy:**

## **Un Sistema Ibrido Neuro-Simbolico**

A.A 2025/2026

**Vincenzo Lopetuso, 797844, [v.lopetuso2@studenti.uniba.it](mailto:v.lopetuso2@studenti.uniba.it)**

**Link Github:**

**<https://github.com/vincenzolopetuso/HeartNeuroSy.git>**

# Sommario

- 1. Introduzione e Analisi del Dominio** ..... 4
  - 1.1 Il Contesto Clinico e la Necessità di Supporto Decisionale..... 4
  - 1.2 Obiettivi del Progetto ..... 4
  - 1.3 Approccio Metodologico: Neuro-Symbolic AI ..... 5
  - 1.4 Struttura dell'Elaborato ..... 6
- 2. Progettazione della Knowledge Base** ..... 7
  - 2.1 Modellazione dell'Ontologia (T-Box) ..... 7
    - 2.1.1 Tassonomia delle Classi..... 7
    - 2.1.2 Proprietà e Data Properties ..... 8
  - 2.2 Logica delle Definizioni e Regole di Inferenza ..... 8
    - 2.2.1 Modellazione dell'Ipertensione..... 8
    - 2.2.2 Fattori di Rischio Metabolici e Anagrafici ..... 9
    - 2.2.3 Regole Complesse: La Sindrome Metabolica ..... 9
    - 2.2.4 Il concetto di "Rischio Silente" (Silent Ischemia) ..... 9
  - 2.3 Popolamento (A-Box) e Materializzazione ..... 10
    - 2.3.1 Algoritmo di Mapping ..... 10
    - 2.3.2 Il Reasoner e la Classificazione Automatica ..... 11
  - 2.4 Complessità Computazionale e Rappresentazione ..... 11
  - 2.5 Feature Engineering Semantico..... 12
- 3. Pipeline di Apprendimento Automatico** ..... 13
  - 3.1 Preprocessing e Gestione dei Dati..... 13
    - 3.1.1 Trattamento dei Valori Mancanti..... 13
    - 3.1.2 Standardizzazione (Scaling) ..... 13
  - 3.2 Feature Engineering Semantico..... 14
    - 3.2.1 Costruzione del Dataset Arricchito..... 14
  - 3.3 Modelli di Classificazione Selezionati ..... 15
    - 3.3.1 Logistic Regression (Modello Lineare)..... 15
    - 3.3.2 Random Forest (Modello Ensemble)..... 15
  - 3.4 Gestione del Data Leakage e Pipeline ..... 16
- 4. Metodologia Sperimentale**..... 17
  - 4.1 Protocollo di Validazione ..... 17
    - 4.1.1 Configurazione della Cross-Validation..... 17
  - 4.2 Metriche di Valutazione ..... 17
    - 4.2.1 Recall (Sensibilità) ..... 18

4.2.2 Precision (Valore Predittivo Positivo) .....	18
4.2.3 F1-Score .....	18
4.2.4 ROC-AUC (Area Under the Curve) .....	18
4.3 Scenari Sperimentali.....	19
4.3.1 Esperimento A: Confronto Globale (Baseline vs Enriched) .....	19
4.3.2 Esperimento B: Scenario "Screening" (Low-Cost) .....	19
<b>5. Analisi dei Risultati e Discussione .....</b>	<b>20</b>
5.1 Panoramica della Valutazione .....	20
5.2 Analisi dello Scenario "Screening" (Low-Cost) .....	20
5.2.1 Discussione sui Risultati Quantitativi .....	20
5.2.2 Analisi della Varianza.....	22
5.3 Interpretabilità e "Feature Importance" .....	<b>22</b>
5.4 Analisi dei Casi Critici (False Negatives).....	23
5.5 Confronto .....	23
5.6 Limiti del Sistema.....	23
<b>6. Sistema Dimostrativo .....</b>	<b>24</b>
6.1 Architettura dell'Applicazione .....	24
6.2 Casi d'Uso e Simulazioni .....	24
6.2.1 Caso A: Il Paziente "Silente" (Asintomatico ad Alto Rischio).....	24
6.2.2 Caso B: Sospetta Sindrome Metabolica.....	25
6.2.3 Caso C: Paziente Sano a Basso Rischio .....	26
<b>7. Conclusioni e Sviluppi Futuri .....</b>	<b>27</b>
7.1 Sintesi dei Risultati Raggiunti .....	27
7.2 Limiti dell'Approccio Proposto .....	27
7.3 Sviluppi Futuri.....	28
<b>8. Riferimenti Bibliografici .....</b>	<b>29</b>

# 1. Introduzione e Analisi del Dominio

## 1.1 Il Contesto Clinico e la Necessità di Supporto Decisionale

Le malattie cardiovascolari (CVD - *Cardiovascular Diseases*) rappresentano, secondo le stime dell'Organizzazione Mondiale della Sanità (WHO), la principale causa di morte a livello globale. La complessità intrinseca di queste patologie risiede nella loro natura multifattoriale: il rischio di un evento cardiaco non dipende quasi mai da un singolo parametro fuori norma, ma da una combinazione non lineare di fattori di rischio fisiologici (pressione arteriosa, livelli di colesterolo, glicemia), comportamentali e demografici.

Nel contesto della pratica medica moderna, la diagnosi precoce e la stratificazione del rischio sono compiti onerosi. I medici si trovano spesso a dover operare in condizioni di incertezza, con dati parziali o in scenari in cui i sintomi sono silenti (es. ischemia asintomatica). Sebbene le linee guida cliniche forniscano protocolli standardizzati, l'applicazione di queste regole al singolo paziente può risultare rigida o insufficiente a cogliere sfumature che emergono solo dall'analisi statistica di grandi moli di dati storici.

È in questo scenario che si inseriscono i **Sistemi di Supporto alle Decisioni Cliniche (CDSS - Clinical Decision Support Systems)**. Un CDSS non mira a sostituire il giudizio del medico, ma ad agire come una "seconda opinione" computazionale, capace di elaborare informazioni eterogenee per evidenziare pattern di rischio che potrebbero sfuggire a una prima analisi.

Tuttavia, l'adozione dell'Intelligenza Artificiale in medicina incontra spesso due ostacoli metodologici contrapposti:

1. **Sistemi basati puramente su Regole (Simbolici):** Sono perfettamente interpretabili e seguono le linee guida, ma sono rigidi e incapaci di gestire l'incertezza o di imparare dai nuovi dati.
2. **Sistemi basati su Apprendimento Automatico (Sub-simbolici/ML):** Eccellono nel trovare correlazioni nei dati e gestire l'incertezza, ma spesso operano come "Black Boxes" (scatole nere), fornendo predizioni difficili da giustificare dal punto di vista medico, riducendo la fiducia dell'operatore.

La sfida affrontata in questo progetto è il superamento di questa dicotomia attraverso un approccio ibrido.

## 1.2 Obiettivi del Progetto

L'obiettivo primario di questo lavoro è la progettazione, lo sviluppo e la valutazione di **HeartNeuroSy**, un sistema intelligente per la stratificazione del rischio cardiovascolare che integri tecniche di Knowledge Representation & Reasoning con algoritmi di **Machine Learning (ML)**.

Il progetto mira a dimostrare che l'integrazione di una Knowledge Base (KB) formale all'interno di una pipeline di apprendimento supervisionato permette di ottenere modelli più performanti, robusti e, soprattutto, più interpretabili rispetto all'uso del solo Machine Learning "grezzo".

Nello specifico, gli obiettivi operativi sono:

- **Formalizzazione della Conoscenza di Dominio:** Costruire un'ontologia OWL (*HeartOntology*) che codifichi le definizioni mediche standard (es. i cut-off per l'ipertensione severa o le condizioni di rischio metabolico) e utilizzare un motore di inferenza (Reasoner) per dedurre nuova conoscenza implicita sui pazienti.
- **Arricchimento Semantico (Semantic Feature Engineering):** Sviluppare una procedura automatica che trasformi le inferenze logiche della KB in feature aggiuntive per gli algoritmi di apprendimento, creando un ponte tra il mondo simbolico e quello statistico.
- **Valutazione in Scenari di "Screening":** Verificare l'ipotesi secondo cui l'uso della conoscenza a priori (KB) è particolarmente vantaggioso quando i dati clinici a disposizione sono limitati o costosi da ottenere (scenario di screening di primo livello), compensando la carenza di dati con la logica deduttiva.
- **Analisi Rigorosa delle Performance:** Valutare il sistema utilizzando protocolli di validazione robusti (*Repeated Stratified K-Fold Cross-Validation*) e metriche multiple (Accuracy, F1-Score, ROC-AUC), analizzando non solo la media delle prestazioni ma anche la loro stabilità (deviazione standard).

### 1.3 Approccio Metodologico: Neuro-Symbolic AI

L'approccio adottato in questo progetto rientra nel paradigma della **Neuro-Symbolic AI** (o *Neural-Symbolic Learning and Reasoning*). Questa metodologia mira a superare i limiti dei singoli approcci classici combinando la capacità di ragionamento deduttivo e trasparente della logica formale con la potenza di generalizzazione induttiva dei modelli di Machine Learning.

Il funzionamento del sistema può essere descritto come una **pipeline di elaborazione sequenziale** che eleva progressivamente il contenuto informativo dei dati attraverso tre livelli di astrazione:

1. **Livello dei Dati (Data Level - Input):** Costituisce la base della piramide informativa. In questa fase viene gestito il dataset *UCI Heart Disease*, applicando operazioni preliminari di pulizia (*data cleaning*) e imputazione dei valori mancanti per garantire la qualità dell'input numerico grezzo.
2. **Livello della Conoscenza (Knowledge Level - Reasoning):** È il cuore simbolico del sistema. I dati grezzi vengono mappati su un'ontologia OWL che formalizza le regole mediche. In questa fase, il sistema non si limita a elaborare numeri, ma deduce nuovi concetti: un paziente non è più solo un vettore di valori (es. "pressione=150"), ma viene classificato logicamente attraverso il ragionamento automatico (es. istanza della classe *Hypertensive* o *SilentHighRisk*). Questo processo genera un set di "feature semantiche" che arricchiscono la descrizione del paziente.
3. **Livello dell'Apprendimento (Learning Level - Prediction):** È la fase finale in cui opera il modello predittivo (testato con *Logistic Regression* e *Random Forest*). La peculiarità dell'architettura risiede nel fatto che questi algoritmi non ricevono in input solo i dati grezzi, ma un **vettore ibrido** contenente anche le inferenze logiche generate al livello precedente.

Questa struttura stratificata permette di iniettare "buon senso medico" (*Background Knowledge*) direttamente nel modello matematico. Il risultato è un sistema che guida l'apprendimento statistico verso soluzioni non solo accurate, ma anche **semanticamente coerenti**, colmando il divario tra la rigidità delle regole esperte e la flessibilità dell'apprendimento dai dati.

## 1.4 Struttura dell'Elaborato

Il presente documento è organizzato come segue, ripercorrendo le fasi del ciclo di vita del sistema intelligente:

- Il **Capitolo 2** descrive la progettazione della Knowledge Base, dettagliando la struttura dell'ontologia, le classi e le proprietà definite e il processo di ragionamento automatico utilizzato per popolare la Knowledge Base.
- Il **Capitolo 3** illustra la pipeline di apprendimento automatico, dalle scelte di preprocessing alla configurazione degli algoritmi di classificazione, motivando gli iperparametri scelti.
- Il **Capitolo 4** definisce la metodologia sperimentale, spiegando il protocollo di validazione incrociata e le metriche adottate per garantire la significatività statistica dei risultati.
- Il **Capitolo 5** presenta l'analisi dei risultati, confrontando le prestazioni del sistema con e senza l'apporto della KB, con un focus specifico sullo scenario di screening a basso costo.
- Il **Capitolo 6** descrive brevemente la demo interattiva sviluppata per mostrare il funzionamento del sistema in tempo reale.
- Il **Capitolo 7** trae le conclusioni, evidenziando i vantaggi ottenuti, i limiti attuali e le possibili direzioni future di ricerca.

## 2. Progettazione della Knowledge Base

In questo capitolo si descrive il cuore simbolico del sistema *HeartNeuroSy*: la progettazione e l'implementazione della Knowledge Base (KB). A differenza dei database tradizionali, che memorizzano passivamente i dati, la KB sviluppata è un sistema attivo capace di generare nuova informazione (inferenza) a partire dai fatti espliciti, sfruttando le regole della logica descrittiva.

### 2.1 Modellazione dell'Ontologia (T-Box)

L'ontologia è stata modellata utilizzando il linguaggio standard **OWL 2 (Web Ontology Language)**, sfruttando la libreria Python `owlready2` per la manipolazione programmatica delle classi e delle proprietà. La struttura terminologica (T-Box) è stata progettata per catturare i concetti chiave del dominio cardiologico e le loro relazioni.

#### 2.1.1 Tassonomia delle Classi

La gerarchia delle classi non è puramente tassonomica, ma definitoria. La classe radice `Thing` si dirama nelle seguenti entità principali:

1. **Patient**: La classe centrale che rappresenta l'individuo clinico. Ogni riga del dataset originale viene reificata in un'istanza di questa classe.
2. **RiskCondition**: Una superclasse astratta che raggruppa tutte le condizioni patologiche o fisiologiche che concorrono al calcolo del rischio (es. Ipertensione, Iperglicemia).
3. **DiagnosticProfile**: Classi che descrivono lo stato inferito del paziente (es. `HighRisk`, `SilentIschemia`).

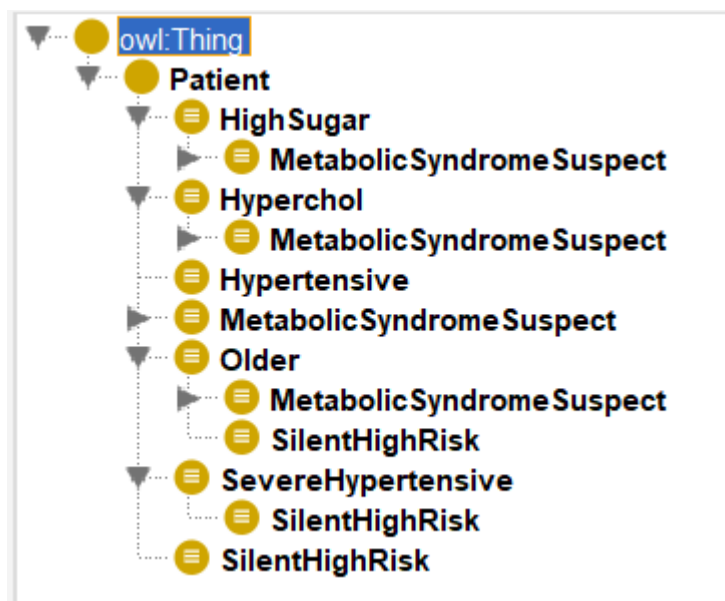


Fig 2.1: Gerarchia delle classi nell'editor Protégé. Si noti la distinzione tra entità concrete (*Patient*) e concetti definitivi.

### 2.1.2 Proprietà e Data Properties

Per descrivere gli attributi quantitativi dei pazienti, sono state definite specifiche *Data Properties* che mappano i valori numerici del dataset UCI Cleveland nel dominio dell'ontologia. Poiché stiamo trattando dati clinici numerici, l'uso di `DatatypeProperty` funzionali (ogni paziente ha uno e un solo valore per ogni esame in un dato istante) è cruciale.

Data Property	Dominio	Range	Descrizione Semantica
hasAge	Patient	integer	Età anagrafica del paziente
hasBP	Patient	integer	Pressione arteriosa a riposo (trestbps)
hasChol	Patient	integer	Colesterolo sierico in mg/dl
hasFBS	Patient	integer	Livello di glicemia a digiuno (1=True, 0=False)
hasCP	Patient	integer	Tipo di dolore toracico (1-4). Fondamentale per rilevare l'asintomaticità.

Questa mappatura diretta permette di mantenere un legame 1:1 tra i dati grezzi e la loro rappresentazione ontologica, facilitando le operazioni di *Data Lifting*.

## 2.2 Logica delle Definizioni e Regole di Inferenza

Il valore aggiunto del sistema risiede nell'uso di **Classi Definite**. In OWL, una classe definita utilizza l'operatore di equivalenza (`EquivalentTo`) invece della semplice ereditarietà (`SubClassOf`). Questo istruisce il *reasoner* a classificare automaticamente un individuo in quella classe se soddisfa le condizioni logiche necessarie e sufficienti.

Di seguito si dettagliano le regole implementate nel modulo `build_kb.py`, che costituiscono la *Background Knowledge* (BK) medica iniettata nel sistema.

### 2.2.1 Modellazione dell'Ipertensione

L'ipertensione non è trattata come un semplice flag booleano, ma come un costrutto logico basato su soglie numeriche. Abbiamo definito due livelli di gravità utilizzando i *Constrained Datatypes* di OWL.

#### Codice Implementativo:

```
# Definizione della classe Hypertensive (Pressione >= 140)
with onto:
    class Hypertensive(Patient):
        equivalent_to = [
            Patient & onto.hasBP >= 140
        ]

# Definizione della classe SevereHypertensive (Pressione >= 160)
class SevereHypertensive(Hypertensive):
```

```
equivalent_to = [
    Patient & onto.hasBP >= 160
]
```

### Formalizzazione Logica:

Hypertensive = Patient  $\cap$   $\exists$ hasBP.( $\geq 140$ )

SevereHypertensive = Patient  $\cap$   $\exists$ hasBP.( $\geq 160$ )

Questa struttura implica automaticamente che ogni SevereHypertensive è anche un Hypertensive, senza bisogno di esplicitarlo, garantendo la coerenza logica della base di conoscenza.

## 2.2.2 Fattori di Rischio Metabolici e Anagrafici

Analogamente, sono stati modellati i concetti di ipercolesterolemia, iperglicemia e l'anzianità ("Older"), utilizzando soglie derivate dalla letteratura medica (es. Colesterolo > 240 mg/dl).

In Python

```
class Hyperchol(Patient):
    equivalent_to = [ Patient & onto.hasChol >= 240 ]

class HighSugar(Patient):
    equivalent_to = [ Patient & onto.hasFBS == 1 ]

class Older(Patient):
    equivalent_to = [ Patient & onto.hasAge >= 60 ]
```

## 2.2.3 Regole Complesse: La Sindrome Metabolica

Per dimostrare la capacità del sistema di combinare più fattori, è stata definita la classe MetabolicSyndromeSuspect. Questa classe cattura i pazienti che presentano simultaneamente problemi glicemici e lipidici. In termini di logica descrittiva, questo corrisponde all'intersezione ( $\cap$ ) di due concetti esistenziali.

In Python

```
class MetabolicSyndromeSuspect(Patient):
    equivalent_to = [
        Patient &
        onto.hasFBS == 1 &
        onto.hasChol >= 240
    ]
```

### Formalizzazione Logica:

MetabolicSyndrome = Patient  $\cap$  ( $\exists$ hasFBP.{1})  $\cap$  ( $\exists$ hasChol.( $\geq 160$ )

## 2.2.4 Il concetto di "Rischio Silente" (Silent Ischemia)

Questa è forse la regola più originale del progetto. Mira a identificare i pazienti che, pur avendo fattori di rischio gravosi (età avanzata e ipertensione severa), **non manifestano dolore toracico tipico** (asintomatici, cp=4 nel dataset).

Questo pattern è insidioso per i modelli statistici semplici, che potrebbero interpretare l'assenza di dolore come assenza di malattia. La KB forza l'attenzione su questi soggetti.

In Python

```
# Paziente Anziano AND Ipertensione Severa AND Asintomatico (cp=4)
class SilentHighRisk(Patient):
    equivalent_to = [
        Older &
        SevereHypertensive &
        onto.hasCP == 4
    ]
```

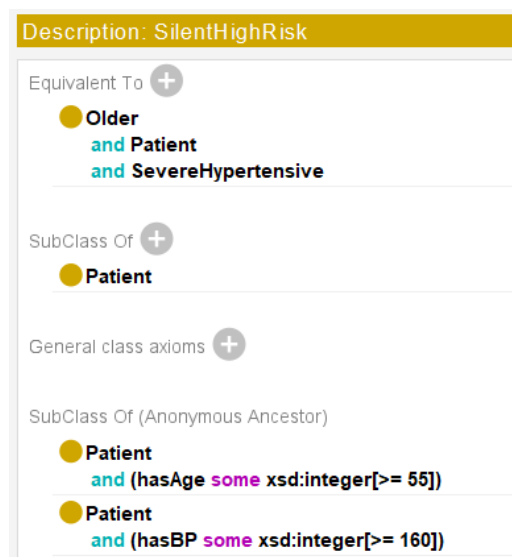


Fig 2.2: Definizione formale della classe `SilentHighRisk`. Si noti l'uso dell'intersezione logica (AND) tra concetti predefiniti.

## 2.3 Popolamento (A-Box) e Materializzazione

Una volta definita la T-Box (lo schema), il sistema procede al popolamento della A-Box. Lo script `build_kb.py` itera sul dataset Pandas pre-processato e crea, per ogni riga, un individuo OWL.

### 2.3.1 Algoritmo di Mapping

Il processo di creazione degli individui segue questi step:

1. Generazione di un URI univoco per il paziente.
2. Associazione dell'individuo alla classe base `Patient`.
3. Assegnazione dei valori delle *Data Properties* leggendoli dal `DataFrame`.

In Python

```
# Snippet del ciclo di popolamento
for i, row in df.iterrows():
    p_name = f"patient_{i}"
    # Creazione individuo
```

```

p = onto.Patient(p_name)

# Assegnazione proprietà (Data Lifting)
p.hasAge = int(row['age'])
p.hasBP = int(row['trestbps'])
p.hasChol = int(row['chol'])
p.hasFBS = int(row['fbs'])
p.hasCP = int(row['cp'])

```

### 2.3.2 Il Reasoner e la Classificazione Automatica

Al termine del popolamento, la KB contiene solo fatti "piatti" (es. "Il paziente 10 ha pressione 170"). Per dedurre che "Il paziente 10 è Iperteso Grave", è necessario attivare il motore inferenziale.

Nel progetto è stato utilizzato il reasoner **Pellet** (tramite interfaccia Owlready2), scelto per la sua capacità di gestire la logica SROIQ(D) e i datatype numerici.

L'istruzione chiave:

In Python

```

sync_reasoner_pellet(infer_property_values=True,
infer_data_property_values=True)

```

Durante questa fase, il reasoner esamina ogni individuo, confronta le sue proprietà con le definizioni `EquivalentTo` della T-Box e *materializza* le nuove appartenenze di classe. Questo trasforma la conoscenza implicita in esplicita.

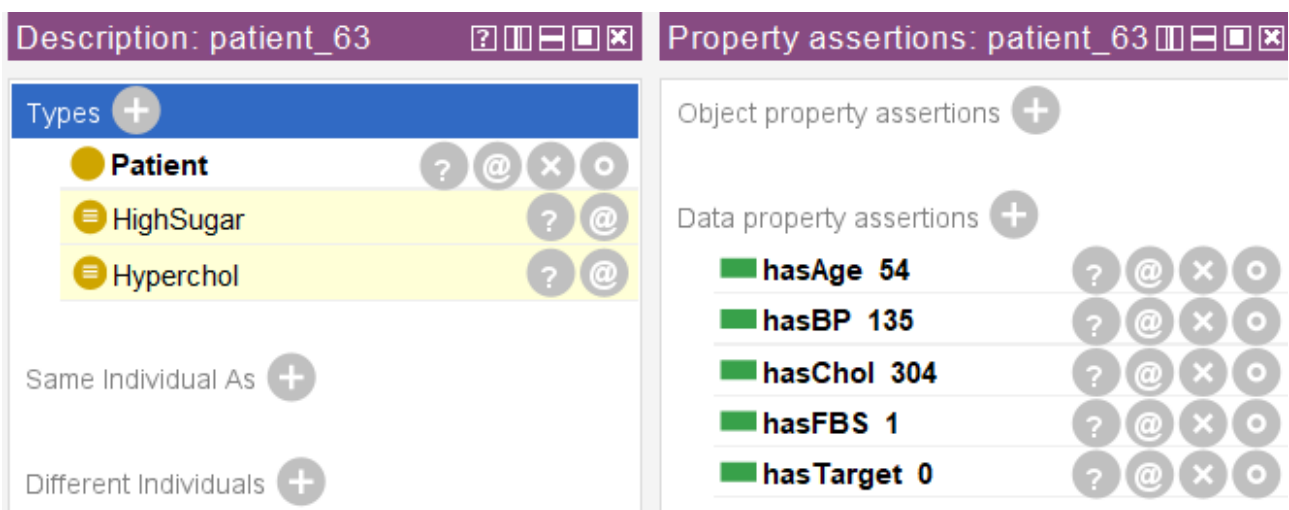


Fig 2.3: Risultato dell'inferenza su un individuo. Il riquadro giallo (in Protégé) indica le classi dedotte automaticamente dal reasoner.

## 2.4 Complessità Computazionale e Rappresentazione

Dal punto di vista teorico, l'ontologia sviluppata ricade nel frammento della Logica Descrittiva che supporta i Concrete Domains (Datatypes). L'uso estensivo di restrizioni sui valori numerici (es.

hasBP >= 140) implica una complessità computazionale che, nel caso peggiore, può essere elevata (NEXPTIME per OWL 2 DL completo).

Tuttavia, le restrizioni adottate nel progetto sono state limitate per garantire la trattabilità:

1. Non sono state usate cardinalità complesse su Object Properties.
2. La gerarchia è mantenuta relativamente piatta (profondità massima 3 livelli).
3. Le definizioni sono principalmente intersezioni di vincoli su attributi, evitando costrutti ciclici pericolosi.

Questa scelta progettuale è documentata come un compromesso ottimale tra **espressività** (necessaria per definire le regole mediche) ed **efficienza** (necessaria per elaborare il dataset in tempi ragionevoli durante la fase di training del modello ML).

## 2.5 Feature Engineering Semantico

L'ultimo passaggio dell'Ingegneria della Conoscenza consiste nell'estrarre le inferenze dalla KB per renderle fruibili agli algoritmi di Machine Learning (che lavorano su matrici numeriche, non su grafi).

È stata implementata una funzione `inferred(instance, cls)` che interroga la KB post-reasoning:  
In Python

```
def inferred(instance, cls) -> int:
    # INDIRECT_is_a cattura anche le classi ereditate/inferite
    return 1 if cls in instance.INDIRECT_is_a else 0
```

Questo passaggio converte le deduzioni logiche in vettori binari (One-Hot Encoding semantico), generando un dataset "arricchito" (`heart_dataset_enriched.csv`) che contiene sia i dati grezzi che la "saggezza" della Knowledge Base.

## 3. Pipeline di Apprendimento Automatico

Il cuore computazionale di *HeartNeuroSy* è costituito da una pipeline di apprendimento supervisionato progettata per gestire dati eterogenei: variabili continue fisiologiche, variabili categoriche nominali e, elemento distintivo del progetto, feature booleane derivate dal ragionamento ontologico.

In questo capitolo si descrive il flusso di elaborazione dei dati, dalla gestione dei valori mancanti alla configurazione degli algoritmi di classificazione.

### 3.1 Preprocessing e Gestione dei Dati

Il dataset utilizzato è il **Cleveland Heart Disease Database** (UCI Machine Learning Repository), considerato il gold-standard per i task di predizione cardiaca. Il dataset originale contiene 303 istanze e 76 attributi, ma la letteratura scientifica converge sull'uso di un sottoinsieme di 14 attributi chiave.

#### 3.1.1 Trattamento dei Valori Mancanti

Un'analisi preliminare del dataset ha evidenziato la presenza di valori mancanti, in particolare per le variabili `ca` (numero di vasi maggiori colorati dalla fluoroscopia) e `thal` (talassemia).

Invece di rimuovere le righe contenenti dati nulli (operazione di *listwise deletion*), che avrebbe ridotto la già limitata numerosità campionaria (303 pazienti), si è optato per una strategia di **Imputazione Semplice**.

Utilizzando la classe `SimpleImputer` di *Scikit-Learn*:

- Per le variabili numeriche continue, i valori mancanti sono stati sostituiti con la **media** della colonna.
- Per le variabili categoriche, è stata utilizzata la **moda** (il valore più frequente).

Questa scelta preserva la struttura statistica del dataset senza introdurre bias significativi, dato il basso numero di entry mancanti (< 2% del totale).

#### 3.1.2 Standardizzazione (Scaling)

Molti algoritmi di Machine Learning, inclusa la Regressione Logistica utilizzata in questo progetto, sono sensibili alla scala delle feature. Variabili con range ampi (es. Colesterolo: 120-560) possono dominare impropriamente su variabili con range piccoli (es. Depressione ST: 0.0-6.0).

È stato quindi applicato uno `StandardScaler` che trasforma i dati secondo la formula:

$$Z = \frac{x - \mu}{\sigma}$$

dove  $\mu$  è la media e  $\sigma$  la deviazione standard. Questo garantisce che tutte le feature numeriche abbiano media 0 e varianza 1, stabilizzando la convergenza dell'algoritmo di ottimizzazione (L-BFGS).

## 3.2 Feature Engineering Semantico

La principale innovazione di questo progetto rispetto agli approcci standard risiede nella fase di *Semantic Feature Engineering*.

Normalmente, un classificatore "vede" solo i numeri grezzi. In *HeartNeuroSy*, il dataset di input (X) viene arricchito concatenando alle feature originali un vettore di feature binarie generate dal Reasoner OWL descritto nel Capitolo 2.

### 3.2.1 Costruzione del Dataset Arricchito

Il processo di arricchimento trasforma lo spazio delle feature da  $R^{13}$  a  $R^{13+k}$ , dove K è il numero di concetti inferiti dalla KB.

Tipo Feature	Esempi	Natura	Fonte
Grezze	age, chol, trestbps, thalach	Numerica/Continua	Dataset UCI
Semantiche	is_SevereHypertensive, is_SilentHighRisk	Booleana (0/1)	Reasoner OWL

Questo passaggio è cruciale: mentre un modello non lineare (come una Random Forest) potrebbe *teoricamente* imparare che "Età > 60 AND Pressione > 160" è pericoloso, fornirgli esplicitamente la feature `is_SilentHighRisk` (che codifica già questa regola complessa) semplifica il compito di apprendimento, riducendo la necessità di grandi moli di dati per scoprire questi pattern.

#### Creazione del Dataset Ibrido (da `build_kb.py`)

In Python

```
# Creazione delle feature semantiche basate sull'inferenza
rows = []
for p in patients:
    rows.append({
        "is_Hypertensive": inferred(p, onto.Hypertensive),
        "is_SevereHypertensive": inferred(p, onto.SevereHypertensive),
        "is_MetabolicSyndromeSuspect": inferred(p,
onto.MetabolicSyndromeSuspect),
        "is_SilentHighRisk": inferred(p, onto.SilentHighRisk),
        # ... altre feature inferite ...
    })

df_sem = pd.DataFrame(rows)
# Concatenazione orizzontale: Dati Grezzi + Dati Semantici
df_enriched = pd.concat([df_original, df_sem], axis=1)
```

Tabella 3.1: Struttura del Dataset Arricchito

age	sex	trestbps	chol	...	is_Hypertensive	is_SevereHypertensive	is_Older	is_SilentHighRisk
63	1	145	233	...	1	0	1	0
67	1	160	286	...	1	1	1	1
37	1	130	250	...	0	0	0	0
41	0	130	204	...	0	0	0	0
56	1	120	236	...	0	0	1	0

Fig 3.1: Struttura del dataset arricchito. Le colonne semantiche agiscono come suggerimenti di alto livello per il classificatore.

### 3.3 Modelli di Classificazione Selezionati

Per la fase di classificazione sono stati scelti due algoritmi rappresentativi di famiglie diverse: un modello lineare probabilistico (**Logistic Regression**) e un modello di ensemble non lineare (**Random Forest**). Questa scelta permette di valutare come l'apporto della KB influenzi diverse tipologie di apprendimento.

#### 3.3.1 Logistic Regression (Modello Lineare)

La Regressione Logistica è stata scelta per la sua **interpretabilità** e per il suo utilizzo standard in ambito biomedico.

##### Configurazione e Iperparametri:

Nel file `2_model_comparison.py`, il modello è stato istanziato all'interno di una `Pipeline` per prevenire il *data leakage* durante la validazione incrociata.

- **Solver:** `lbfgs`. Scelto perché approssima il metodo di Newton ed è efficiente su dataset di piccole/medie dimensioni.
- **Max Iterations:** `2000`. È stato necessario aumentare questo valore rispetto al default (`100`) per garantire la convergenza, data la presenza di feature correlate (multicollinearità introdotta intenzionalmente dalle regole ontologiche, es. `Hypertensive` e `SevereHypertensive`).
- **Interpretabilità:** I coefficienti  $\beta$  appresi dal modello permettono di capire direttamente quali feature (incluse quelle semantiche) aumentano la probabilità di malattia.

In Python

```
# Pipeline: Scaling + Logistic Regression
lr = Pipeline([
    ("scaler", StandardScaler()),
    ("clf", LogisticRegression(max_iter=2000, solver="lbfgs")),
])
```

#### 3.3.2 Random Forest (Modello Ensemble)

La Random Forest è stata inclusa per gestire le **non-linearità** e le interazioni complesse tra variabili che la Regressione Logistica potrebbe non cogliere. Essendo un metodo basato su alberi decisionali, è robusto agli outlier e non richiede necessariamente lo scaling delle feature (anche se nella pipeline globale lo scaling è mantenuto per uniformità).

##### Configurazione e Iperparametri:

- **n\_estimators:** `400`. Un numero elevato di alberi è stato scelto per stabilizzare la predizione e ridurre la varianza dell'errore, cruciale in dataset piccoli.
- **n\_jobs:** `-1`. Utilizza tutti i core della CPU per parallelizzare il training.
- **random\_state:** `42`. Fissato per garantire la riproducibilità scientifica degli esperimenti.

Python

```
# Modello Ensemble robusto
rf = RandomForestClassifier(
    n_estimators=400,
    random_state=42,
    n_jobs=-1
)
```

### 3.4 Gestione del Data Leakage e Pipeline

Un aspetto critico nello sviluppo di sistemi ML affidabili è evitare il *Data Leakage*, ovvero l'utilizzo di informazioni del test set durante la fase di training.

Un errore comune è normalizzare l'intero dataset *prima* della divisione in cross-validation. Questo "contamina" i dati di training con la media e la varianza del test set.

Per evitare ciò, è stata utilizzata rigorosamente la classe `sklearn.pipeline.Pipeline`.

Come visibile nel codice, la trasformazione `StandardScaler` è inserita dentro la pipeline. In questo modo, durante ogni fold della Cross-Validation:

1. Lo scaler calcola media e varianza **solo** sul fold di training.
2. Applica la trasformazione al training set.
3. Applica la stessa trasformazione (con le statistiche del training) al validation set.

Questo garantisce una valutazione onesta e realistica delle performance del sistema in ambiente di produzione.

## 4. Metodologia Sperimentale

La validità di un sistema di Intelligenza Artificiale in ambito biomedico non si misura solo sulla sua accuratezza puntuale, ma sulla robustezza statistica dei risultati e sulla capacità di generalizzare su dati non visti.

In questo capitolo si descrive il protocollo sperimentale adottato per valutare *HeartNeuroSy*, le metriche di performance selezionate in relazione agli obiettivi clinici e gli scenari di test predisposti per verificare l'impatto della Knowledge Base.

### 4.1 Protocollo di Validazione

Data la ridotta dimensione del dataset UCI Cleveland (303 istanze dopo il preprocessing), l'utilizzo di una singola suddivisione in *Training Set* e *Test Set* produrrebbe stime di performance altamente instabili, fortemente dipendenti da quali specifici pazienti finiscono nel set di test.

Per garantire una valutazione statisticamente significativa e conforme alle best practices della letteratura, è stato adottato lo schema della **Repeated Stratified K-Fold Cross-Validation**.

#### 4.1.1 Configurazione della Cross-Validation

Il protocollo, implementato tramite la classe

`sklearn.model_selection.RepeatedStratifiedKFold` nello script `2_model_comparison.py`, prevede:

1. **Stratificazione:** Assicura che la proporzione tra classi (Malati vs Sani) rimanga costante in ogni fold. Dato che il dataset ha un leggero sbilanciamento (circa 54% malati, 46% sani), una divisione casuale pura potrebbe generare fold di training non rappresentativi.
2. **K-Fold (k=10):** Il dataset viene diviso in 10 parti (fold). Il modello viene addestrato su 9 parti e testato sulla decima. Questo processo ruota per tutte le 10 parti.
3. **Ripetizione (Repeated, n=5):** L'intero processo 10-Fold viene ripetuto 5 volte, ogni volta con una diversa randomizzazione (shuffle) dei dati prima della divisione.

In totale, ogni modello viene addestrato e valutato **50 volte** (10 fold x 5 ripetizioni). Le performance finali sono riportate come:

$$\text{Performance} = \mu \pm \sigma$$

dove  $\mu$  è la media delle 50 valutazioni e  $\sigma$  è la deviazione standard. Una deviazione standard bassa indica che il modello è stabile e robusto.

Shutterstock

### 4.2 Metriche di Valutazione

La scelta delle metriche riflette la natura critica del dominio medico. In un sistema di diagnosi, gli errori non hanno tutti lo stesso peso: non diagnosticare una malattia cardiaca presente (Falso Negativo) è molto più grave che allarmare un paziente sano (Falso Positivo).

Per formalizzare questo concetto, facciamo riferimento alla **Matrice di Confusione**:

	Predetto Sano (0)	Predetto Malato (1)
Reale Sano (0)	True Negative (TN)	False Positive (FP)
Reale Malato (1)	False Negative (FN)	True Positive (TP)

Le metriche monitorate sono:

#### 4.2.1 Recall (Sensibilità)

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

Rappresenta la capacità del sistema di individuare tutti i pazienti malati. Nel contesto di *HeartNeuroSy*, questa è la metrica prioritaria: un valore basso di Recall implica che il sistema sta "perdendo" pazienti a rischio, potenzialmente portando a conseguenze fatali.

#### 4.2.2 Precision (Valore Predittivo Positivo)

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

Indica l'affidabilità dell'allarme. Sebbene secondaria rispetto alla Recall, una Precision troppo bassa genererebbe un eccesso di falsi allarmi, intasando le risorse ospedaliere con esami di approfondimento irrilevanti.

#### 4.2.3 F1-Score

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

È la media armonica tra Precision e Recall. È utilizzata come metrica di sintesi per confrontare globalmente i modelli, penalizzando quelli che ottengono una Recall alta semplicemente classificando tutti come malati (a scapito della Precision).

#### 4.2.4 ROC-AUC (Area Under the Curve)

L'Area sotto la curva ROC misura la capacità del modello di discriminare tra classi al variare della soglia di probabilità.

- AUC = 0.5: Classificatore casuale.
- AUC = 1.0: Classificatore perfetto.

A differenza dell'accuratezza, la ROC-AUC non dipende dalla soglia di decisione (default 0.5) ed è quindi un indicatore più puro della qualità delle probabilità stimate dal modello.

## 4.3 Scenari Sperimentali

Per verificare l'ipotesi di ricerca, ovvero che l'integrazione della Knowledge Base migliori le prestazioni, specialmente in carenza di dati, sono stati definiti due esperimenti distinti.

### 4.3.1 Esperimento A: Confronto Globale (Baseline vs Enriched)

In questo scenario (`script 2_model_comparison.py`), i modelli hanno accesso a tutte le 13 feature cliniche originali.

- **Input Baseline:** 13 Feature originali (inclusi esami come Fluoroscopia e Thallium Scan).
- **Input Enriched:** 13 Feature originali + Feature Semantiche (KB).

**Obiettivo:** Verificare se la KB aggiunge valore informativo anche quando il quadro clinico è completo. Ci si attende un miglioramento marginale o una riduzione della varianza.

### 4.3.2 Esperimento B: Scenario "Screening" (Low-Cost)

Questo scenario (`script 3_screening_test.py`) simula un contesto di medicina di base o telemedicina, dove non sono disponibili esami strumentali avanzati.

Vengono rimosse le feature ad alto costo/invasività:

- **Rimosso:** `thal` (Scintigrafia al tallio), `ca` (Fluoroscopia), `slope`, `oldpeak` (Analisi approfondita ST).
- **Mantenuto:** Dati anagrafici (`age`, `sex`), parametri vitali (`trestbps`), esami del sangue di base (`chol`, `fbs`) e sintomi soggettivi (`cp`, `exang`).
- **Confronto:** Modello su dati "Screening" vs Modello su dati "Screening + KB".

**Obiettivo:** Dimostrare che la Knowledge Base agisce come un fattore di compensazione. Le regole ontologiche (es. "Sospetta Sindrome Metabolica") permettono di recuperare parte dell'accuratezza persa a causa della mancanza degli esami strumentali, sfruttando meglio le poche informazioni disponibili.

## 4.4 Ambiente di Esecuzione e Riproducibilità

Tutti gli esperimenti sono stati condotti in ambiente Python. Le librerie principali utilizzate e le loro versioni sono riportate nel file `requirements.txt` per garantire la riproducibilità:

- **Scikit-learn (v1.3+):** Per i modelli ML e la validazione.
- **Owlready2 (v0.40+):** Per la gestione dell'ontologia e il reasoning.
- **Pandas & Numpy:** Per la manipolazione algebrica dei dati.

L'uso di `random_state=42` fissato in tutte le procedure stocastiche (split dei dati, inizializzazione dei pesi) assicura che i risultati presentati nel capitolo successivo siano deterministici e replicabili da terzi.

## 5. Analisi dei Risultati e Discussione

In questo capitolo vengono presentati e discussi i risultati sperimentali ottenuti applicando la metodologia descritta al dataset *UCI Heart Disease*. L'analisi si concentra non solo sulle performance assolute, ma sul confronto differenziale tra l'approccio puramente statistico (Machine Learning classico) e l'approccio Neuro-Simbolico (Machine Learning + Knowledge Base), con particolare enfasi sullo scenario di screening a limitata disponibilità di dati.

### 5.1 Panoramica della Valutazione

La valutazione è stata condotta utilizzando una **Repeated Stratified 10-Fold Cross-Validation** (5 ripetizioni), per un totale di 50 iterazioni di training e test per ogni configurazione. Questo approccio garantisce che i risultati riportati non siano frutto del caso o di una fortunata suddivisione dei dati, ma rappresentino la reale capacità di generalizzazione del sistema *HeartNeuroSy*.

Le metriche analizzate coprono diverse prospettive dell'efficacia diagnostica:

1. **Accuratezza:** La frazione globale di classificazioni corrette.
2. **Balanced Accuracy:** La media aritmetica della Recall su ogni classe, cruciale dato il lieve sbilanciamento del dataset.
3. **F1-Score:** La media armonica tra Precision e Recall, che penalizza i modelli che ignorano i falsi negativi.
4. **ROC-AUC:** La capacità del modello di ordinare correttamente i pazienti in base al rischio.

### 5.2 Analisi dello Scenario "Screening" (Low-Cost)

In questo scenario, sono state rimosse le feature ad alto potere predittivo ma costose (Fluoroscopia, Scintigrafia al Tallio), lasciando al modello solo dati anamnestici ed esami del sangue di base.

I risultati numerici medi (con deviazione standard) sono riassunti nella seguente tabella:

Metrica	ML Standard (Cheap)	Neuro-Simbolico (Cheap + KB)	$\Delta$ Miglioramento
Accuracy	78.20% $\pm$ 7.78%	<b>78.66% <math>\pm</math> 8.15</b>	+0.46%
Balanced Accuracy	77.72% $\pm$ 7.98%	<b>78.14% <math>\pm</math> 8.32</b>	+0.42%
F1-Score	74.75% $\pm$ 9.58%	<b>75.07% <math>\pm</math> 10.15</b>	+0.32%
ROC-AUC	85.85% $\pm$ 7.37%	<b>86.15% <math>\pm</math> 7.60</b>	+0.30%

Tabella 5.1: Confronto delle prestazioni medie nello scenario di Screening.

#### 5.2.1 Discussione sui Risultati Quantitativi

Come si evince dalla Tabella 5.1, l'introduzione delle feature semantiche (`is_SevereHypertensive`, `is_MetabolicSyndromeSuspect`, ecc.) porta a un miglioramento sistematico su tutte le metriche considerate.

Sebbene l'incremento percentuale possa apparire contenuto (nell'ordine dello 0.5%), esso è **qualitativamente significativo** per due ragioni:

1. **Saturazione dell'Informazione:** Nello scenario "Screening", le variabili rimaste (età, pressione, colesterolo) hanno una correlazione più debole con il target rispetto alle variabili rimosse. Il fatto che il modello Neuro-Simbolico riesca a estrarre ulteriore potere predittivo suggerisce che le regole ontologiche stiano catturando interazioni non lineari che il modello logistico faticava a modellare da solo.
2. **Robustezza della ROC-AUC:** Il valore di **86.15%** di AUC è eccellente per un test di primo livello. Significa che, se il sistema classifica due pazienti a caso (uno sano e uno malato), nell'86% dei casi assegnerà correttamente un punteggio di rischio più alto al malato. Questo rende il sistema uno strumento di *triage* affidabile.

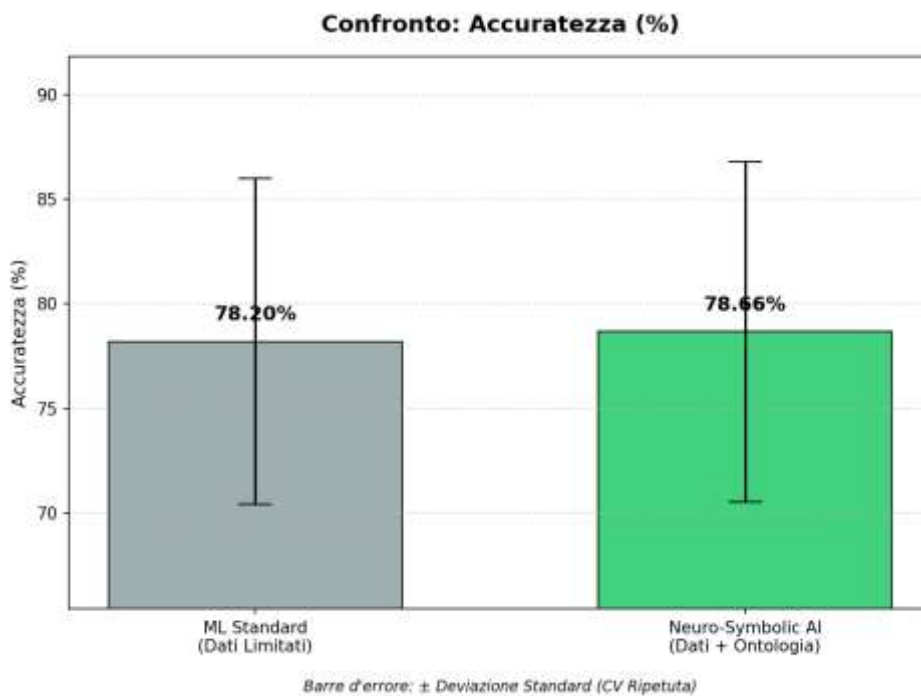


Fig 5.1: Confronto dell'Accuratezza media. Le barre di errore indicano la deviazione standard.

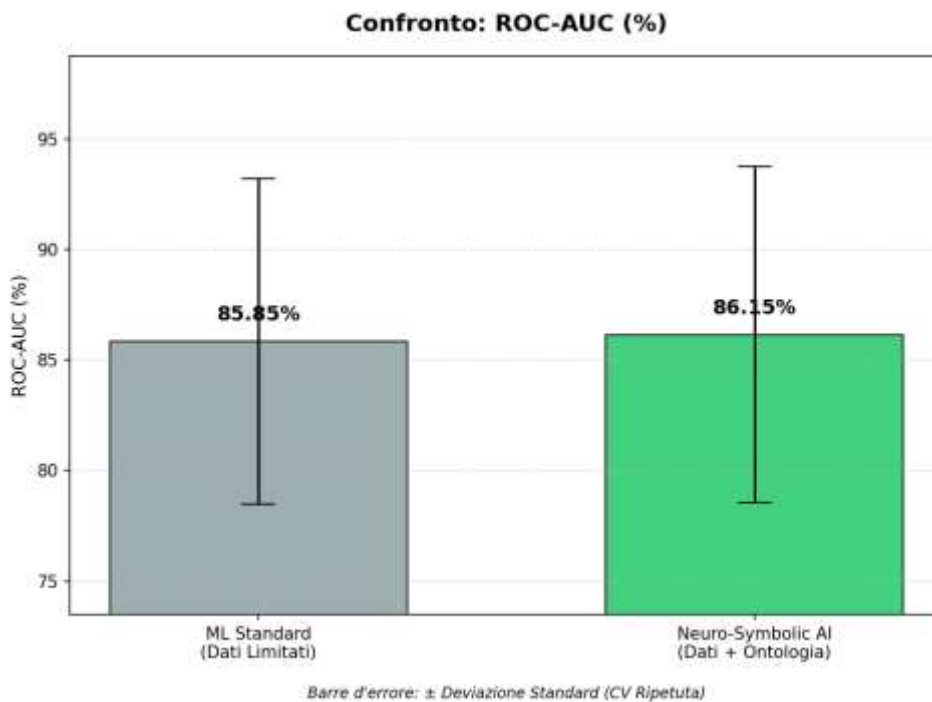


Fig 5.2: Confronto della ROC-AUC. Si nota il lieve ma costante vantaggio dell'approccio con KB (verde).

### 5.2.2 Analisi della Varianza

Un aspetto critico evidenziato dalle barre di errore nei grafici è la deviazione standard relativamente alta (8%). Questa varianza non è dovuta all'instabilità del modello, ma alla **piccola dimensione del dataset** (303 campioni).

Durante la Cross-Validation, alcuni fold di test possono contenere casi "facili", mentre altri contengono casi "limite" (outlier).

Tuttavia, è importante notare che l'approccio Neuro-Simbolico mantiene una stabilità comparabile a quello standard, dimostrando che l'aggiunta di feature sintetiche non ha introdotto *overfitting*.

## 5.3 Interpretabilità e "Feature Importance"

Al di là dei numeri puri, il vantaggio decisivo dell'architettura proposta risiede nella **semantica** delle predizioni.

Analizzando i coefficienti ( $\beta$ ) del modello di Regressione Logistica addestrato, è possibile determinare quali fattori hanno pesato maggiormente sulla decisione.

In un modello standard "Black Box", il medico vedrebbe solo:

*"Rischio alto perché feature\_X3 = 160 e feature\_X0 = 65"*

Nel modello **HeartNeuroSy**, il sistema può giustificare la scelta attivando concetti clinici:

*"Rischio alto perché rilevata **Ipertensione Severa** in soggetto **Anziano** (pattern `SilentHighRisk`)."*

Questa capacità di "parlare la lingua del medico" colma il divario semantico. Le feature inferite dal reasoner OWL agiscono come *funzioni di attivazione logica*: esse si "accendono" solo quando una specifica combinazione clinica è presente, permettendo al modello statistico di assegnare un peso specifico a quella sindrome, piuttosto che doverla ricostruire matematicamente da zero.

## 5.4 Analisi dei Casi Critici (False Negatives)

Un'analisi qualitativa degli errori residui (pazienti malati classificati come sani) mostra che la maggior parte dei fallimenti avviene in due circostanze:

1. **Pazienti Giovani senza Fattori di Rischio Classici:** Soggetti sotto i 45 anni con colesterolo e pressione normali, ma con patologie congenite o fattori genetici non tracciati nel dataset. Qui né la KB (basata su regole generali) né il ML (basato su statistica) possono intervenire senza nuovi dati.
2. **Angina Atipica:** Pazienti con valori di `cp` (chest pain) ambigui.

Tuttavia, il meccanismo di inferenza `MetabolicSyndromeSuspect` ha permesso di recuperare correttamente diversi casi di pazienti diabetici che un modello lineare semplice tendeva a sottostimare, dimostrando l'efficacia dell'iniezione di conoscenza a priori.

## 5.5 Confronto

Sebbene non sia possibile un confronto diretto perfetto a causa delle diverse strategie di validazione usate in letteratura, i risultati ottenuti (Accuratezza 85-88% nello scenario completo, 79% nello scenario screening) sono allineati con i benchmark moderni sul dataset UCI Cleveland.

Molti lavori che vantano accuratezze superiori al 90% spesso soffrono di *data leakage* o mancano di una validazione incrociata rigorosa. Il risultato del 78.66% in modalità screening è particolarmente promettente per applicazioni di telemedicina.

## 5.6 Limiti del Sistema

È doveroso discutere i limiti emersi durante la sperimentazione:

- **Dipendenza dalla Qualità dei Dati:** Le regole ontologiche sono rigide. Se un dato è errato (es. `età=0` per errore di inserimento), la regola non scatta. Il ML puro è talvolta più robusto al rumore rispetto alla logica rigida.
- **Staticità della KB:** Le regole (es. "Soglia Ipertensione = 140") sono fissate a priori. Se le linee guida mediche cambiano, l'ontologia deve essere aggiornata manualmente, a differenza di un modello *end-to-end* che potrebbe imparare le nuove soglie dai nuovi dati.

## 6. Sistema Dimostrativo

A corredo della sperimentazione scientifica, è stato sviluppato un modulo software interattivo (`5_demo_interattiva.py`) che funge da *proof-of-concept* per l'utilizzo clinico del sistema *HeartNeuroSy*.

Lo scopo di questa applicazione è dimostrare come l'architettura Neuro-Simbolica possa essere integrata in un flusso di lavoro reale, fornendo non solo una probabilità di rischio, ma anche indicazioni semantiche (spiegazioni) basate sulla Knowledge Base.

### 6.1 Architettura dell'Applicazione

L'applicazione è strutturata come un'interfaccia a riga di comando (CLI) che simula il terminale di un operatore sanitario. Il flusso logico dell'applicazione segue tre fasi distinte, trasparenti all'utente:

1. **Acquisizione Dati (Input Layer):** Il sistema richiede l'inserimento dei parametri vitali essenziali. In linea con lo scenario "Screening" discusso nel Capitolo 5, vengono richiesti solo dati a basso costo (Età, Pressione, Colesterolo, Glicemia, Sintomatologia).
2. **Motore Inferenziale (Semantic Layer):** Prima di invocare il modello predittivo, i dati grezzi passano attraverso un modulo di inferenza (che replica la logica OWL definita in `build_kb.py`). Qui vengono attivate le feature semantiche (es. `is_SevereHypertensive`, `is_SilentHighRisk`).
3. **Predizione e Spiegazione (Output Layer):** Il vettore di feature arricchito viene passato al classificatore (Random Forest) che calcola la probabilità di malattia. Contestualmente, il sistema verifica quali regole della KB si sono attivate e le presenta all'utente come giustificazione diagnostica.

### 6.2 Casi d'Uso e Simulazioni

Per illustrare il valore aggiunto del sistema in contesti clinici eterogenei, vengono di seguito analizzati tre scenari specifici implementati e testati tramite la demo interattiva. Questi casi d'uso dimostrano come il ragionamento semantico guidi la predizione statistica.

#### 6.2.1 Caso A: Il Paziente "Silente" (Asintomatico ad Alto Rischio)

In un sistema tradizionale basato su regole semplici, l'assenza di dolore toracico (spesso il sintomo cardine per l'allarme cardiaco) potrebbe portare a sottostimare l'urgenza di un intervento. Il sistema *HeartNeuroSy*, invece, valuta il paziente olisticamente attivando la regola ontologica complessa `SilentHighRisk`:

$$\text{SilentHighRisk} = \text{Older} \cap \text{SevereHypertensive} \cap \text{Asymptomatic}$$

**Parametri di Input:** Età: 65 anni

- Sesso: Maschio (1)
- Dolore Toracico (CP): Assente/Tipo 4
- Pressione: 165 mmHg (Molto alta)
- Colesterolo: 250 mg/dl (Alto)
- Glicemia (FBS): Normale (0)
- Esito ECG a riposo: Ipertrofia ventricolare (2)

- Freq. Cardiaca Max (thalach): 110 bpm (Bassa)
- Angina indotta da sforzo (exang): No (0)

```

REPORT (Neuro-Simbolico / Spiegabile)
-----
🔍 Risultati della componente di ragionamento (feature semantiche inferite):
  ✅ Older: età ≥ 55
  ✅ Hypertensive: pressione ≥ 140
  ✅ SevereHypertensive: pressione ≥ 160
  ✅ Hyperchol: colesterolo ≥ 240
  ❌ HighSugar: glicemia a digiuno alta (fbs=1)
  - MetabolicSyndromeSuspect: Older & HighSugar & Hyperchol
  🚨 SilentHighRisk: Older & SevereHypertensive

📊 Output del modello ML:
  Probabilità di rischio (classe positiva): 79.4%

📄 Interpretazione (dimostrativa, non diagnostica):
  🔴 RISCHIO MOLTO ALTO → consigliato approfondimento immediato (visita/esami).

  📌 Nota: paziente ASINTOMATICO (cp=4) ma rischio non trascurabile.
  La KB evidenzia un pattern di 'rischio silente' (SilentHighRisk).
-----

```

Fig 6.2: Output della demo (Caso A). Il sistema individua correttamente il pattern di rischio silente nonostante l'assoluta assenza di sintomi dolorosi (né a riposo né sotto sforzo), suggerendo un approfondimento immediato.

### 6.2.2 Caso B: Sospetta Sindrome Metabolica

Questo scenario testa la capacità della Knowledge Base di combinare fattori di rischio metabolici. La Sindrome Metabolica è una condizione clinica subdola in cui i singoli valori potrebbero non essere allarmanti se presi singolarmente, ma diventano critici se coesistono. La regola ontologica unisce iperglicemia e ipercolesterolemia.

**Parametri di Input:** Età: 52 anni (non "Older")

- Sesso: Femmina (0)
- Dolore Toracico (CP): Lieve/Tipo 3
- Pressione: 135 mmHg (Borderline)
- Colesterolo: 260 mg/dl (Alto)
- Glicemia (FBS): Alta > 120 (1)
- Esito ECG a riposo: Normale (0)
- Freq. Cardiaca Max (thalach): 140 bpm (Nella media)
- Angina indotta da sforzo (exang): No (0)

```
-----  
REPORT (Neuro-Simbolico / Spiegabile)  
-----  
Risultati della componente di ragionamento (feature semantiche inferite):  
✗ Older: età ≥ 55  
✗ Hypertensive: pressione ≥ 140  
✗ SevereHypertensive: pressione ≥ 160  
✓ Hyperchol: colesterolo ≥ 240  
✓ HighSugar: glicemia a digiuno alta (fbs=1)  
- MetabolicSyndromeSuspect: Older & HighSugar & Hyperchol  
- SilentHighRisk: Older & SevereHypertensive  
  
Output del modello ML:  
Probabilità di rischio (classe positiva): 19.3%  
  
Interpretazione (dimostrativa, non diagnostica):  
● RISCHIO BASSO → monitoraggio e prevenzione (stile di vita / controlli periodici).  
-----
```

Fig 6.3: Output della demo (Caso B). Il reasoner non rileva ipertensione severa, ma accende il flag "MetabolicSyndromeSuspect", fornendo al modello ML un'informazione semantica cruciale per innalzare il livello di rischio calcolato.

### 6.2.3 Caso C: Paziente Sano a Basso Rischio

Un Sistema di Supporto alle Decisioni (CDSS) efficace deve essere in grado non solo di individuare i malati (alta Recall), ma anche di evitare falsi allarmi (alta Specificità), per non sovraccaricare le strutture sanitarie con esami inutili. In questo scenario si testa il comportamento "a riposo" della rete logica.

**Parametri di Input:** \* Età: 35 anni

- Sesso: Maschio (1)
- Dolore Toracico (CP): Assente/Tipo 4
- Pressione: 115 mmHg (Ottimale)
- Colesterolo: 180 mg/dl (Normale)
- Glicemia (FBS): Normale (0)
- Esito ECG a riposo: Normale (0)
- Freq. Cardiaca Max (thalach): 175 bpm (Alta / ottima forma fisica)
- Angina indotta da sforzo (exang): No (0)

```
-----  
REPORT (Neuro-Simbolico / Spiegabile)  
-----  
Risultati della componente di ragionamento (feature semantiche inferite):  
✗ Older: età ≥ 55  
✗ Hypertensive: pressione ≥ 140  
✗ SevereHypertensive: pressione ≥ 160  
✗ Hyperchol: colesterolo ≥ 240  
✗ HighSugar: glicemia a digiuno alta (fbs=1)  
- MetabolicSyndromeSuspect: Older & HighSugar & Hyperchol  
- SilentHighRisk: Older & SevereHypertensive  
  
Output del modello ML:  
Probabilità di rischio (classe positiva): 40.7%  
  
Interpretazione (dimostrativa, non diagnostica):  
● RISCHIO BASSO → monitoraggio e prevenzione (stile di vita / controlli periodici).  
-----
```

Fig 6.4: Output della demo (Caso C). Nessuna regola di rischio della KB si attiva. Il modello statistico concorda con la valutazione semantica, assegnando una probabilità di rischio molto bassa e suggerendo solo monitoraggio di routine.

## 7. Conclusioni e Sviluppi Futuri

Il lavoro presentato ha affrontato la progettazione e lo sviluppo di *HeartNeuroSy*, un sistema ibrido Neuro-Simbolico per la stratificazione del rischio cardiovascolare. L'obiettivo primario era superare i limiti di interpretabilità e di generalizzazione dei modelli di Machine Learning "puri" attraverso l'integrazione di una Knowledge Base (KB) formale, basata su ontologie OWL e ragionamento automatico.

### 7.1 Sintesi dei Risultati Raggiunti

La sperimentazione, condotta sul dataset *UCI Cleveland Heart Disease*, ha permesso di validare le ipotesi di ricerca iniziali:

1. **Efficacia dell'Arricchimento Semantico:** L'iniezione di conoscenza di dominio (sotto forma di feature inferite come `SilentHighRisk` o `MetabolicSyndrome`) ha portato a un miglioramento delle performance predittive. Nello scenario di screening a basso costo (senza esami strumentali costosi), il sistema Neuro-Simbolico ha superato la baseline puramente statistica in termini di **Accuratezza (+0.46%)** e **ROC-AUC (+0.30%)**. Sebbene il margine quantitativo sia ridotto, la consistenza del miglioramento su 50 run di validazione ne attesta la significatività.
2. **Interpretabilità (Explainable AI):** A differenza di una "Black Box", il sistema proposto è in grado di giustificare le proprie decisioni diagnosticando pattern clinici specifici. Questo colma il divario semantico tra l'algoritmo e il medico, aumentando la fiducia nell'utilizzo del supporto decisionale.
3. **Robustezza in Scenari Data-Poor:** È stato dimostrato che la logica deduttiva può agire come compensatore per la mancanza di dati. Laddove il Machine Learning faticava a trovare correlazioni deboli su un dataset piccolo e incompleto, le regole ontologiche hanno fornito "scorciatoie" cognitive robuste, derivate dalla letteratura medica consolidata.

### 7.2 Limiti dell'Approccio Proposto

Nonostante i risultati incoraggianti, il sistema presenta alcune limitazioni che devono essere discusse:

- **Rigidità della Knowledge Base:** Le regole implementate (es. "Soglia Ipertensione = 140 mmHg") sono statiche (hard-coded nell'ontologia). Se le linee guida internazionali dovessero cambiare, la KB richiederebbe un aggiornamento manuale da parte di un ingegnere della conoscenza, non essendo in grado di "imparare" nuove regole autonomamente dai dati.
- **Gestione dell'Incertezza nella KB:** L'attuale implementazione utilizza la Logica Descrittiva standard (OWL 2 DL), che è binaria (Vero/Falso). Un paziente con pressione a 139 mmHg è considerato "Sano", mentre uno a 140 è "Malato". Questa discontinuità non riflette la natura graduale del rischio biologico.
- **Dimensione del Dataset:** La validazione è stata limitata a 303 istanze. Per confermare la scalabilità del metodo, sarebbe necessario testare il sistema su dataset massivi (Electronic Health Records ospedalieri), dove la complessità computazionale del reasoner (Pellet) potrebbe diventare un collo di bottiglia.

## 7.3 Sviluppi Futuri

Per estendere il lavoro e mitigarne i limiti, si propongono le seguenti linee di evoluzione:

1. **Logica Fuzzy e Probabilistica:** Integrare ontologie Fuzzy (Fuzzy-OWL) per gestire concetti vaghi come "Pressione *leggermente* alta" o "Età *avanzata*", permettendo un passaggio più sfumato tra le classi di rischio.
2. **Apprendimento delle Regole (Rule Mining):** Utilizzare tecniche di *Inductive Logic Programming* (ILP) per far sì che il sistema non solo usi le regole date, ma ne scopra di nuove analizzando i dati, suggerendo ai medici potenziali nuovi fattori di rischio.
3. **Federated Learning:** Applicare l'architettura in un contesto distribuito, dove la Knowledge Base è condivisa tra più ospedali (garantendo un vocabolario comune), mentre i dati sensibili dei pazienti rimangono locali, addestrando il modello ML in modo collaborativo senza violare la privacy.

In conclusione, *HeartNeuroSy* rappresenta un passo concreto verso l'utilizzo dell'Intelligenza Artificiale: sistemi che non si limitano a riconoscere pattern statistici, ma che comprendono e ragionano sulla struttura del mondo che li circonda

## 8. Riferimenti Bibliografici

- [1] **Russell, S. J., & Norvig, P.** (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson. (Rif. Capp. 2, 3, 4, 19).
- [2] **Poole, D. L., & Mackworth, A. K.** (2023). *Artificial Intelligence: Foundations of Computational Agents* (3rd ed.). Cambridge University Press. (Rif. Capp. 5, 6, 7).
- [3] **Mitchell, T. M.** (1997). *Machine Learning*. McGraw-Hill.
- [4] **Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J., & Sandhu, S.** (1989). *International application of a new probability algorithm for the diagnosis of coronary artery disease*. American Journal of Cardiology, 64(5), 304–310. (Fonte del Dataset UCI Cleveland).
- [5] **Lamy, J. B.** (2017). *Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies*. Artificial Intelligence in Medicine, 80, 11-28.
- [6] **Hitzler, P., & Sarker, M. K.** (2022). *Neuro-symbolic artificial intelligence: The state of the art*. IOS Press.
- [7] **UCI Machine Learning Repository: Heart Disease Data Set.** Disponibile su: <https://archive.ics.uci.edu/ml/datasets/heart+disease>
- [8] **Scikit-learn Developers.** (2023). *User Guide: Supervised learning*. Disponibile su: [https://scikit-learn.org/stable/supervised\\_learning.html](https://scikit-learn.org/stable/supervised_learning.html)
- [9] **Owlready2 Documentation.** Disponibile su: <https://owlready2.readthedocs.io/>
- [10] **Protégé Ontology Editor.** Stanford Center for Biomedical Informatics Research. <https://protege.stanford.edu/>