Titolo

*Mancini Riccardo, Silvi Francesco, Tarsi Enrico.*

*Abstract*—Questa relazione è incentrata nell’ambito probabilistico e dell’apprendimento automatico sviluppato in Prolog. Nella prima parte viene illustrato il dataset preso in considerazione e le modifiche applicate ad esso per poterci lavorare. Poi è stato trattato l’aspetto probabilistico utilizzando il “pacchetto” Cplint, grazie al quale è stato possibile estrarre la possibilità che un fenomeno di interesse si verifichi applicando diverse leggi della teoria della probabilità. Dopodiché, i dati ottenuti sono stati confrontati con i risultati di classificazione ricavati nella fase di apprendimento automatico. Apprendimento reso possibile grazie alla costruzione di alberi di induzione secondo i criteri del calcolo dell’entropia di Shannon e dell’indice di diversità di Gini. Infine, sono state spese alcune parole sulla possibilità di implementare quest’ultima fase di apprendimento direttamente in Cplint per poter ottenere informazioni sempre più accurate alla realtà (NON SO SE E’ QUESTO CHE TROVEREMO…)

# Introduzione

Il progetto svolto tratta un approfondimento nell’ambito probabilistico e dell’apprendimento automatico in Prolog. Questo è stato svoltoconsiderando lo stesso dataset di riferimento approfondito di seguito.

La prima fase del progetto tratta l’aspetto probabilistico che permette di dare un'indicazione sulle relazioni fra i valori presenti nel dataset. Con l’utilizzo del Cplint vengono calcolate delle probabilità rilevanti (semplici, congiunte e condizionate), che consentono di studiare i legami fra i singoli attributi e la diagnosi del paziente. Questo studio affianca il risultato dell'apprendimento, effettuato sul medesimo dataset, per valutarne la coerenza.  
Sull’apprendimento sviluppato, oltre la valutazione dell’accuratezza di classificazione e di conseguenza il suo errore, viene svolto un confronto fra i due criteri di scelta dell’attributo (Shannon e Gini) per l’induzione dell’albero decisionale.

Inoltre, si è studiato un modo per rendere gli algoritmi meno onerosi in termini di complessità computazionale.

# Dataset

L'Heart Failure Prediction dataset è una collezione di dati costituita da ben 746 casi di pazienti di cui si conoscono 12 caratteristiche (o features) che in un modo o nell'altro influenzano lo stato di salute del cuore. Di fatto, l'insufficienza cardiaca è un evento comune causato da malattie cardiovascolari, e grazie ai campioni raccolti in questo dataset è possibile prevedere una possibile malattia cardiaca. Infatti, le persone con malattie cardiovascolari, o ad alto rischio cardiovascolare (per la presenza di uno o più fattori di rischio come ipertensione, diabete, iperlipidemia o malattie già accertate), necessitano di una diagnosi precoce. Questo rende un modello di machine learning di fondamentale importanza, ed è proprio per questo motivo che è stato ideato il dataset in questione.

Per quanto riguarda la struttura il dataset contiene, come già anticipato, 746 casi, ognuno dei quali descritto da ben 12 valori che possono assumere i vari attributi. Nello specifico i 12 attributi, di cui l'ultimo definisce la classe (ossia la salute del paziente), sono: **Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, ST\_Slope, HeartDisease.**

## Discretizzazione

Nel dataset alcuni attributi assumono valori continui o discreti ma comunque molto ampi per il nostro caso d'uso. Un esempio è l'età, che spazia dai 28 fino ai 77 anni, o il colesterolo che può potenzialmente assumere infiniti valori. Per questo è stato necessario intervenire con una discretizzazione del dataset, poiché le informazioni, derivanti da uno studio che tratta ogni singolo valore di un attributo come se stante, sarebbero potute essere forvianti.

Qui di seguito sono riportate le discretizzazioni applicate.

* **age**: *First, Second, Third;*
* **restingBP**: *Optimal, Normal/High, High, Very high;*
* **cholesterol**: *Desiderable, Moderately, Extremely high;*
* **maxHR**: *1, 2, 3, 4;*
* **oldpeak**: *Low risk, Normal risk, High risk.*

I restanti attributi non necessitano di una discretizzazione, poiché lo erano di propria natura. In particolare, sono:

* **sex**: *M, F;*
* **chest\_pain\_type**: *TA, ATA, NAP, ASY;*
* **fastingBS**: *0, 1;*
* **restingECG**: *Normal, ST, LVH;*
* **excercise\_angina**: *Y, N;*
* **st\_slope**: *Up, Flat, Down;*
* **heart\_disease**: *Y, N.*

Infine, per consentirne l'uso in Prolog è stata costruita la struttura degli esempi in modo tale che assumessero una forma esempio(Classe,[Attributi]), nel quale per "Classe" si intende lo stato di salute ("y" se malato, "n" se non malato), mentre per "[Attributi]" si intende una lista contenente tutti gli attributi con associato il relativo valore assunto nell'esempio.

Sono state anche costruite delle clausole per gli attributi nella forma attributo(NomeAttributo,[ValoriAttributo]) nel quale "[NomeAttributo]" identifica l’attributo in esame, mentre "ValoriAttributo" è la lista contenente i valori che esso può assumere.

**(SOME EXAMPLES)**

# Analisi probabilistica (Cplint)\*

\* Per ulteriori approfondimenti su tale sezione consultare il notebook. (POI SPOSTARE IN FONDO)

È una raccolta di programmi pensati per l'inferenza e l'apprendimento tramite: LPADs, ICL(Indipendent Choise Logic) e CP-logic programs. Nell'attuale progetto ci si è concentrati nella prima forma.

LPAD è un insieme finito di clausole disgiunte annotate. Per annotate si intende che nella testa di ogni atomo è riportata una probabilità. LPAD a differenza del ProbLog consente solamente clausole probabilistiche nella forma di fatti con due sole alternative, una delle quali è implicita.

…

## La libreria Pita

PITA è un sistema integrato nel Cplint avente come fine quello di calcolare la probabilità di una query da un programma probabilistico espresso come LPAD. Per fare ciò il programma sottoforma di LPAD viene prima trasformato in un normale programma contenente delle chiamate ad un diagramma decisionale binario. Lo scopo è poter formare un diagramma decisionale binario che ad ogni subgoal contenga una "spiegazione" per la quale si è arrivati a quel risultato, ossia ne giustifichi la probabilità. Per poter utilizzare questo sistema è prima necessario includerlo attraverso la direttiva:  
:- use\_module(library(pita)).

## Predicati probabilistici elementari

In questa sezione sono riportati i predicati utilizzati per il calcolo delle probabilità semplici di ogni attributo (features) e classe. Con probabilità semplice si intende una proporzione (rapporto) tra il numero dei casi che presentano un determinato valore di un attributo/classe ed il numero totale dei casi presenti nel dataset.

### Probabilità attributi

Per il calcolo delle probabilità relative agli attributi sono stati realizzati i seguenti predicati:

* numero\_persone(-N): che restituisce il numero totale dei casi all'interno del dataset.
* prob\_<attributo>(+Val): per ogni attributo è stato realizzato un predicato con probabilità in cui “*attributo”* è l'attributo che si vuole analizzare e *Val* è il valore dell'attributo che si vuole analizzarea cui viene assegnata la probabilità facendo il rapporto tra il numero dei casi ottenuto con il predicato attributo(+Val,-N) (che restituisce il numero dei casi che presentano il valore (+Val) dell'attributo analizzato) e il numero totale dei casi ottenuto con numero\_persone(-Tot).

### Probabilità classi

Oltre alle probabilità relative agli attributi è sicuramente interessante calcolare quelle relative alla classe, ossia la probabilità di ottenere un caso con problemi cardiovascolari.

Per fare ciò sono stati realizzati tre predicati:

* persone\_malate(-N):restituisce il numero dei casi in cui la classe assume valore ' *y* '.
* prob\_malato(): a cui viene assegnata la probabilità calcolata come il rapporto tra i casi con problemi cardiovascolari ottenuto con persone\_malate(-PM) e il numero totale dei casi ottenuto con numero\_persone(-N).
* prob\_sano(): a cui viene assegnata la probabilità calcolata come il negato della probabilità di prob\_malato().

## Predicati probabilistici più complessi

Nelle sezioni sottostanti sono presenti i predicati utili a calcolare probabilità congiunte e condizionate.

### Probabilità congiunte

* congiunzione(+Salute,+Attr,+Val,-N): viene utilizzato per calcolare il numero (-N) di esempi che soddisfino sia la condizione di salute (+Salute) che la condizione per la quale l'attributo (+Attr) abbia valore (+Val). Per riconoscere l'attributo richiesto viene eseguito il predicato dist\_attr(+Attr,+Val,-Valori).
* congiunzione(+Salute,+[Val1,Val2,Val3Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11],-N): è la versione più generale del precedente, infatti utilizzando questo predicato è possibile calcolare il numero (-N) di esempi che assumono una determinata combinazione di attributi descritti dalla lista passata come argomento della clausola. Per tutti gli attributi che non sono rilevanti nel calcolo è sufficiente utilizzare una variabile incognita.
* prob\_congiunta(+Salute,+Attr,+Val):Con/N: utilizzando il predicato congiunzione(+Salute,+Attr,+Val,-N1) e numero\_persone(-N2) si imposta all'atomo una probabilità pari a P = N1/N2. Questa rappresenta la probabilità congiunta di avere la combinazione descritta nella testa.
* prob\_congiunta(+Salute,+[Val1,Val2,Val3Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11],-N):Con/N: funzionamento analogo al precedente, ma con la differenza che viene utilizzato il predicato più generale congiunzione(+Salute,+[Val1,Val2,Val3Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11],-N). Si è così in grado di imporre una probabilità congiunta di combinazioni più articolate.

### Probabilità condizionate

* prob\_salute\_per\_val(+Salute,+Attr,+Val):PAB/PB: con questo predicato si vuole calcolare la probabilità condizionata, ossia la probabilità che si abbia un determinato stato di salute (+Salute) rispetto un determinato valore (+Val) di un dato attributo (+Attr). Per il calcolo si applica la formula P(Salute|Valore) = P(Salute ⋂ Valore)/P(Valore), nella quale:  
  + *P(Salute ⋂ Valore)*, è calcolata utilizzando congiunzione(+Salute,+Attr,+Val,-PAB);
  + *P(Valore)*, utilizzando congiunzione(\_,+Attr,+Val,-PB).

entrambe restituiscono un numero, non una probabilità, ma andare a dividere per il numero totale di esempi sarebbe superfluo, in quanto si otterrebbe una cancellazione del termine.

* prob\_salute\_per\_val(+Salute,+[Val1,Val2,Val3Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11]):PAB/PB: versione più generica del suo omonimo. Consente di calcolare la probabilità condizionata con una combinazione di valori più articolata. Il calcolo viene effettuato seguendo una logica analoga al caso precedente, ed il risultato che si ottiene è anche stavolta posto nella testa.
* prob\_salute\_per\_val(\_,\_,\_):0: se la combinazione degli eventi nella testa non è compatibile, o gli eventi sono disgiunti la probabilità condizionata è nulla.
* teo\_bayes(+Attr,+Val,+Salute,-PAB): con questo predicato si vuole calcolare la probabilità condizionata nel verso opposto dei precedenti predicati, ovvero la probabilità di avere un certo valore (+Val) di un attributo (+Attr), sapendo a priori lo stato di salute (+Salute). Il risultato viene calcolato applicando il teorema di Bayes: *P(Val|Salute) = P(Salute|Val)\*P(Val)/P(Salute)*.

Nello specifico si ha:

* + *P(Salute|Val)*, ottenuto prelevando la probabilità nella testa di prob\_salute\_per\_val(+Salute,+Attr,+Val);
  + *P(Val)*, ottenuto dalla probabilità nella testa di prob\_congiunta(\_,+Attr,+Val);
  + *P(Salute)*, ottenuto dalla probabilità nella testa di prob\_congiunta(+Salute,\_).

Come ultima cosa viene effettuato il calcolo numerico e si restituisce il risultato in (-PAB).

* teo\_bayes(+[Val1,Val2,Val3,Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11],+Salute,-PAB): costruito per ottenere la probabilità condizionata di avere una determinata combinazione di valori contenuta nella lista della testa, conoscendo lo stato di salute (+Salute).

## Rappresentazione grafica dei risultati

L'ultima sezione del notebook è dedicata alla rappresentazione grafica delle probabilità calcolate in precedenza. Si è fatto uso della libreria C3 (approfondita qui di seguito) per poter renderizzare un istogramma. Nello specifico è stato costruito un grafico per ogni attributo del dataset, nel quale sono state riportate le probabilità di avere un'insufficienza cardiaca per ogni valore che può assumere un determinato attributo. Ovviamente questo valore è stato messo a confronto graficamente con la sua controparte, ossia con la probabilità di NON avere un'insufficienza cardiaca per gli stessi valori dello stesso attributo calcolato prima.

Il predicato che si occupa di richiamare direttamente la libreria C3 è grafico(+Attr,-Chart) che definisce il layout dell'istogramma nel quale poi vengono immessi i valori probabilistici calcolati. Questi valori numerici, corrispondenti alle probabilità sopra citate, sono stati ottenuti sviluppando il predicato calcolo\_hist(+Salute,+Attr,+Valori,+Traccia,-TracciaF), il quale ricorsivamente per ogni attributo, va a calcolare la probabilità condizionata di ogni valore.

### La libreria C3

C3 è una libreria JavaScript utilizzata per il rendering di grafici inerenti ai dati utilizzati. Questo perchè i risultati numerici sono spesso più facili da capire quando vengono visualizzati come un grafico rispetto a un elenco di numeri.

La creazione di un grafico C3 richiede l'inclusione della direttiva :- use\_rendering(c3). e il collegamento di una variabile Prolog a un dict con il tag c3 e i dati necessari a *C3*. Il renderer c3 si comporta in due modi con i dati:

* se non viene specificata alcuna dimensione, la larghezza viene impostata sull'85% della larghezza disponibile e l'altezza a larghezza/2+50. Il grafico viene ridimensionato se disponibile lo spazio cambia.
* il renderer esegue alcuni controlli di integrità di base sui dati e può segnalare un errore.

### Risultati graficati

**(qualche istogramma)**

# Apprendimento automatico

Il machine learning, o apprendimento automatico, nasce dalla teoria che i computer possono imparare ad eseguire compiti specifici senza essere programmati per farlo, grazie al riconoscimento di schemi tra i dati. Il machine learning utilizza algoritmi che imparano dai dati in modo iterativo. Permette, ad esempio, ai computer di individuare informazioni anche sconosciute senza che venga loro segnalato esplicitamente dove cercarle.

L'aspetto più importante del machine learning è la ripetitività, perché più i modelli sono esposti ai dati, più sono in grado di adattarsi in modo autonomo. I computer imparano da elaborazioni precedenti per produrre risultati e prendere decisioni che siano affidabili e replicabili. Il rinnovato interesse nel machine learning è dovuto agli stessi fattori che hanno reso data mining e analisi Bayesiane più popolari che mai; questo ha portato sempre di più in primo piano l’estrazione (semi) automatica di conoscenza nascosta in voluminose basi di dati al fine di renderla disponibile e direttamente utilizzabile.

Con il passare del tempo, l’apprendimento automatico ha trovato spazio in diverse aree di applicazione, tra cui la classificazione, ossia l’individuazione delle caratteristiche che indicano a quale gruppo un certo caso appartiene (es. discriminazione tra pazienti con problemi di insufficienza cardiaca e non). Di conseguenza si sono sviluppati alcuni metodi per la classificazione, infatti, si parla di apprendimento induttivo da esempi per imparare la definizione di una funzione di classificazione. Gli esempi usati per l’apprendimento sono descritti come vettori di coppie attributo-valore per i quali è nota la classe. Il metodo di classificazione su cui si è concentrata la relazione è l’Albero di decisone (o Tree Induction).

## Tree Induction

Le funzioni di classificazione sono apprese in forma di albero dove:

* ogni nodo interno rappresenta una variabile;
* un arco verso un nodo figlio rappresenta un possibile valore per quella proprietà;
* una foglia il valore predetto per la classe a partire dai valori delle altre proprietà, che nell'albero è rappresentato dal cammino (path) dal nodo radice (root) al nodo foglia.

Un albero di decisione viene costruito utilizzando tecniche di apprendimento a partire dall'insieme dei dati iniziali (*training set*) per i quali è nota la classe. Esistono diversi tipi di tecniche che permettono di costruire l’albero. Queste tecniche si differenziano principalmente per la scelta dell’attributo utilizzato come nodo subito successivo durante la costruzione dell’albero.

Di seguito verranno approfondite due possibili strategie che sono state implementate per portare a termine la classificazione.

## Criterio di scelta dell’attributo

### Algoritmo ID3

L’ID3 è un algoritmo di classificazione supervisionato, ovvero si basa su esempi già classificati in un insieme di classi per determinare un modello di classificazione. Il modello prodotto da questo algoritmo è un albero decisionale, grazie al quale è possibile classificare nuovi campioni. La costruzione dell’albero viene effettuata ricorsivamente secondo *il calcolo dell’entropia di Shannon*, grazie al quale ad ogni passo della ricorsione, calcola tra gli attributi rimanenti per il ramo corrente, quello che massimizzerà il guadagno di informazioni.

Quindi, sapendo per definizione che maggiore è l'entropia, minore è la quantità di informazione, assegnando ad ogni valore v di ogni attributo A, contenuti in un nodo un valore di entropia dato dalla formula:

Nella quale B rappresenta un valore booleano della classe esaminata e q la probabilità che essa assuma quel valore. Il criterio seleziona l’attributo A\* con valori vi\*, tali per cui l’entropia complessiva del nodo generato data da:

sia minima.

### Algoritmo CART

L’algoritmo in questione misura la frequenza con cui un elemento casuale nell'insieme sarebbe classificato erroneamente se la sua etichetta fosse scelta casualmente in base alla distribuzione delle etichette nel sottoinsieme. Questo algoritmo si affida *all'indice di diversità di* *Gini*, che può essere calcolato sommando la probabilità che ciascun elemento venga scelto, moltiplicato per la probabilità che sia classificato erroneamente. Raggiunge il suo valore minimo (zero) quando tutti gli elementi dell'insieme sono nella stessa classe della variabile di destinazione.

Perciò, l’algoritmo seleziona l’attributo A con valori v che riesce a minimizzare la formula, facendo si che la formulazione matematica sia la seguente:

## Risultati di classificazione

Applicati entrambi i criteri si è fatto un confronto della loro accuratezza in funzione della dimensione del traning set. I risultati ottenuti mostrano come entrambi i criteri godano di una buona accuratezza già da un ridotto traning set, e di conseguenza di un errore di classificazione nemmeno troppo alto. Basandosi esclusivamente su questi risultati, non è possibile determinare quale dei due sia il migliore, in quanto le loro curve si intersecano più volte. Infatti, con il crescere del training set, il loro comportamento è molto simile. C’è però da dire che attuando criteri differenti nella costruzione dell’albero decisionale, effettuando test su delle collezioni di dati differenti, e magari anche più ampie, le differenze potrebbe essere più evidenti.

Di seguito invece vengono riportate come esempio le matrici di confusione di entrambi gli algoritmi e alcune previsioni effettuate sulla base degli alberi decisionali costruiti.

…(Matrici di confusione)

…(Alcune previsioni)

Mentre i dati estratti dalla matrice di confusione vanno direttamente a costruire i grafici precedentemente illustrati, e quindi non ha bisogno di molte spiegazioni, le previsioni riportate meritano un minimo di spiegazione. Queste ultime sono state ottenute iterando i vari attributi dell’albero decisionale con le coppie attributo-valore del caso che si ha interesse di valutare, fino ad arrivare ad una foglia dell’albero associata ad una, nessuna o più classi. Il primo caso è quello che si ottiene più spesso in quanto dalla classificazione si ottiene una sorta di “certezza” sul risultato. Ovviamente questa certezza deve comunque essere presa con le pinze perché c’è una forte dipendenza da come è stato costruito l’albero di decisione e da come sono stati manipolati in precedenza i dati, infatti con test più ampi potremmo osservare che i modelli soffrono di *overfitting* nel generalizzare la classificazione, anche se dai precedenti risultati non c’è molto che lo lasci pensare. Il secondo caso invece, è dovuto da un’insufficiente quantità di informazione per descrivere il valore di un attributo, perciò l’ipotetico caso che finisce in questa foglia verrebbe valutato come “non classificabile”. Mentre il terzo ed ultimo caso è stato risolto riportando direttamente la probabilità che un attributo foglia appartenga alla classe positiva e di conseguenza a quella negativa.

# Miglioramenti aggiuntivi

**Qualcosa riguardo all’apprendimento totalmente sviluppato in Cplint… quello che diceva Dragoni, durante ricevimento, di aggiungere alla presentazione durante il ricevimento.**

##### Riferimenti

1. Alberti, Marco, et al. "cplint on SWISH: Probabilistic logical inference with a web browser." Intelligenza Artificiale 11.1 (2017): 47-64.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
2. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
3. K. Elissa, “Title of paper if known,” unpublished.
4. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
5. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
6. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.