Valutazioni probabilistiche e tecniche di apprendimento automatico

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 la suite di programmi Cplint, con cui 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 inerenti ai possibili miglioramenti aggiuntivi da poter implementare al progetto in futuro.**

I. Introduzione

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

La prima fase del progetto studia 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 classe che caratterizzano i casi presenti nel dataset. 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.

II. Dataset

L'Heart Failure Prediction[1] dataset è una collezione di dati costituita da ben 918 casi di pazienti di cui si conoscono 12 caratteristiche (o features) che possono influenzare 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 l'insorgere di un'eventuale 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.

Quest'ultimo, come già anticipato, contiene 918 casi, ognuno dei quali descritto da 12 coppie attributo-valore. 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.**

Si è notato che per alcuni pazienti erano presenti delle coppie attributo-valore con valori non utilizzabili (per esempio *Cholesterol* a 0). Perciò, dei 918 casi iniziali, solamente 746 ne sono stati considerati.

A. Discretizzazione

Nel dataset alcuni attributi assumono valori in range 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 a sé 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 sono 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,[Attrs=Vals])`, nella quale per "Classe" si intende lo stato di salute ("y" se malato, "n" se non malato), mentre per "[Attrs=Vals]" si intende una lista contenente tutte le coppie attributo-valore che descrivono l'esempio.

** Per ulteriori approfondimenti sui range di discretizzazione utilizzati consultare il Notebook*

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

Di seguito è riportato il primo esempio contenuto nel dataset, da cui è ben visibile la struttura precedentemente descritta:

```
e(n,[age="Second",sex="M",chest_pain_type="ATA"
,restingBP="High",cholesterol="Extremely high",
fastingBS=0,restingECG="Normal",maxHR=4,exercis
e_angina="N",oldpeak="Lowrisk",st_slope="Up"]).
```

Mentre un esempio di struttura per un attributo è il seguente:

```
a(age,["First","Second","Third"])
```.

III. Analisi probabilistica in Cplint [2]*

Cplint è una raccolta di programmi pensati per l'inferenza e l'apprendimento automatico tramite distribuzioni di probabilità discrete e densità di probabilità continue. Nel caso di studio sono state richieste delle distribuzioni di probabilità prettamente discrete, ed è per questo motivo che nell'attuale progetto ci si è concentrati a sviluppare programmi di tipo LPADs (Logic Programs with Annotated Disjunctions).

Nello specifico, LPAD è un insieme finito di clausole disgiunte annotate. La disgiunzione avviene nella testa della clausola, la quale viene definita da un punto e virgola. Inoltre, gli atomi, sempre della testa, sono associati a delle probabilità separate da due punti. La sintassi è strutturata come segue:

$$h_{i1} : \Pi_{i1}; \ldots; h_{in_i} : \Pi_{in_i} :- b_{i1}, \ldots, b_{im_i}$$

dove $b_{i1}, \ldots, b_{im_i}$ sono letterali, $h_{i1}, \ldots h_{in_i}$ sono atomi e $\Pi_{i1}, \ldots, \Pi_{in_i}$ sono numeri reali compresi tra $[0,1]$. Possono essere interpretate come: " se $b_{i1}, \ldots, b_{im_i}$ sono vere, allora h_{i1} è vera con probabilità Π_{i1} o … o h_{in_i} è vera con probabilità Π_{in_i}".

Mentre per quanto riguarda il corpo della clausola viene utilizzata la sintassi classica del Prolog. La sintassi LPAD prevede che questa sezione contenente clausole probabilistiche venga delimitata da `:- begin_lpad` posto all'inizio e `:- end_lpad` messo in coda.

A. *La libreria Pita*

PITA (Probabilistic Inference with Tabling and Answer subsumption) 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))`.

Viene poi utilizzato il predicato built-in `prob/2` che restituisce la probabilità calcolata di un determinato atomo.

B. *Probabilità elementari*

Con probabilità elementare 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.

Il predicato utilizzato per il calcolo delle probabilità semplici è il seguente:

`prob_semplice(+AC,+Val)`:***NA/N***: che assume probabilità NA/N, dove AC è l'attributo e Val è il valore che si vuole analizzare. La probabilità viene calcolata facendo il rapporto tra il numero dei casi ottenuto con il predicato `numero_val(+AC,+Val,-NA)` ed il numero totale dei casi ottenuto con `numero_persone(-N)`. Nel caso in cui si vuole calcolare la probabilità del valore della classe si richiama il predicato usando la costante "*classe*" in AC (`prob_semplice(+classe,+Val)`).

C. *Probabilità congiunte*

Una probabilità congiunta si riferisce alla probabilità che due o più eventi si verifichino contemporaneamente.

I predicati dedicati al calcolo di queste probabilità sono:

- `prob_congiunta(+Salute,+Attr,+Val)`:***Con/N***: utilizzando il predicato `congiunzione(+Salute,+Attr,+Val,-Con)` (che restituisce il numero di esempi che soddisfano le condizioni richieste) e `numero_persone(-N)` si imposta all'atomo una probabilità pari a P = Con/N. 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, con la differenza che viene utilizzato il predicato più generale `congiunzione(+Salute,+[Val1,Val2,Val3,Val4,Val5,Val6,Val7,Val8,Val9,Val10,Val11],-N)`. Si è così in grado di imporre una probabilità congiunta di combinazioni più articolate.

D. *Probabilità condizionate*

La probabilità di un evento A condizionata dal verificarsi di un evento B è data da:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

I predicati utili per il calcolo di queste probabilità sono riportati di seguito:

- `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 $P(Salute|Valore) = \frac{P(Salute \cap Valore)}{P(Valore)}$, nella quale:
  - *P(Salute ∩ Valore)*, calcolata utilizzando `congiunzione(+Salute,+Attr,+Val,-PAB)`;
  - *P(Valore)*, calcolata utilizzando `congiunzione(_,+Attr,+Val,-PB)`.

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

** Per approfondimenti sui predicati sviluppati per il calcolo delle probabilità consultare il Notebook.*

- `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
- `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:

$$P(Val|Salute) = \frac{P(Salute|Val) \cdot P(Val)}{P(Salute)}$$

  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).

E. Rappresentazione grafica dei risultati

1) La libreria C3

C3 è una libreria JavaScript utilizzata per il rendering di grafici inerenti ai dati passati. Questo perché i risultati numerici sono spesso più facili da comprendere quando vengono rappresentati in un grafico.
La creazione di un grafico C3 richiede: l'inclusione della direttiva `:- use_rendering(c3)`, il collegamento di una variabile Prolog a un dict con il tag `c3` e i dati necessari da riportare nel grafico.

2) Risultati graficati

A questo punto sono state graficate le probabilità calcolate in precedenza. Si è fatto uso della libreria C3, sopra citata, 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 precedentemente.

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.

Qui di seguito sono stati riportati due dei grafici ottenuti (*Figure Figura III.*A*Figura III.*B), per i risultati completi si rimanda di nuovo al notebook sviluppato.

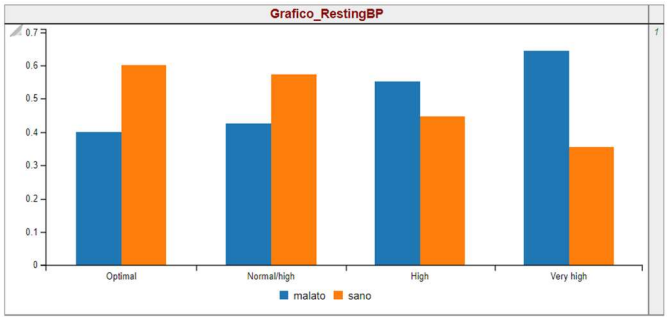

Figura III.A: Grafico restingBP

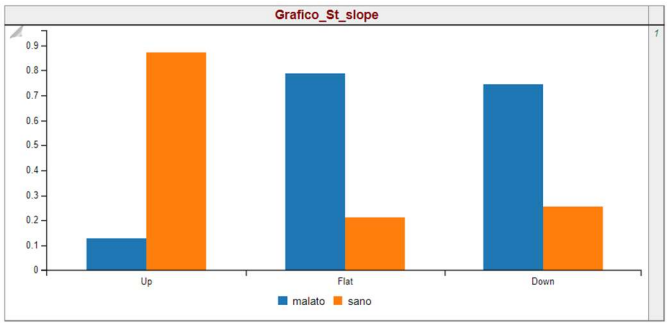

Figura III.B: Grafico st_slope

Si vuole portare particolare attenzione sul secondo istogramma (*Figura III.B*). Da questo si può ben notare come l'attributo '*st_slope*' fornisca da solo un'indicazione molto importante sullo stato di salute presentato dal paziente. Infatti, se il soggetto presenta un tratto st ascendente ('*Up*'), questo sarà in salute con l'87,11% di probabilità. Per i restanti valori di '*st_slope*' invece il risultato è inverso e si ha maggiore probabilità di essere affetti da problemi cardiaci.

Non a caso, è proprio l'attributo '*st_slope*' ad essere scelto dall'algoritmo d'induzione dell'albero come nodo radice.

IV. Apprendimento automatico

Il machine learning, o apprendimento automatico, nasce dall'esigenza di creare sistemi che imparano ad eseguire compiti specifici senza essere programmati per farlo, in base al riconoscimento di schemi tra i dati che utilizzano. Il machine learning utilizza algoritmi che in modo iterativo apprendono dai dati e 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 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.

Tra le varie tipologie di apprendimento esistenti, in questo progetto ci si è concentrati sull'apprendimento supervisionato. Questa metodologia prevede che al modello di apprendimento automatico siano passati degli esempi in ingresso, ritenuti corretti ed etichettati, dai quali poter estrarre delle regole. Questa tecnica si distingue da quella non supervisionata che riceve in input dei dati non etichettati, ma che vengono organizzati sulla base di caratteristiche comuni per cercare di effettuare ragionamenti e previsioni sugli input successivi.

Con il passare del tempo l'apprendimento automatico supervisionato 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; come, ad esempio, l'apprendimento induttivo che utilizza degli esempi per definire la funzione di classificazione vera e propria. Gli esempi usati per l'apprendimento sono descritti come vettori di coppie attributo-valore per i quali è nota la classe (o label/etichetta). Il metodo di classificazione su cui si è concentrata la relazione è l'Albero di decisone (o Tree Induction).

A. Tree Induction

Con questo metodo, 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 rappresenta il valore della classe predetto a partire dai valori delle altre proprietà, che nell'albero sono rappresentate 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, detto *training set*, per i quali è nota la classe. Successivamente si effettua una sua valutazione attraverso un altro insieme di dati etichettati, detto *test set*, dei quali si effettua una classificazione secondo i criteri stabiliti in precedenza dall'albero. A questo punto non resta che osservare quanti casi sono stati classificati correttamente o meno. Esistono diversi tipi di tecniche che permettono di costruire l'albero. Queste tecniche si differenziano principalmente per la scelta dell'attributo utilizzato come nodo successivo durante la costruzione dell'albero.

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

B. Criterio di scelta dell'attributo

1) Algoritmo ID3 [4]

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, valuta tra gli attributi rimanenti per il ramo corrente, quello che massimizza il guadagno di informazioni.

È infatti noto per definizione che maggiore è l'entropia, minore è la quantità di informazione. il calcolo avviene assegnando ad ogni valore v di ogni attributo A contenuti in un nodo, un valore di entropia dato dalla formula:

$$H_{A_v}(B(q)) = -(q \log_2 q + (1-q)\log_2(1-q))$$

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 v_i*, tali per cui l'entropia complessiva del nodo generato data da:

$$\sum_i q_i * H_{A^*_{v_i^*}}(B(q))$$

sia minima.

2) Algoritmo CART [5]

L'algoritmo in questione misura la frequenza con cui un elemento casuale nell'insieme sarebbe classificato erroneamente se la sua classe 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 della stessa classe della variabile di destinazione.

Perciò, l'algoritmo seleziona l'attributo A con valori v che minimizza la formula:

$$Gini(A) = \sum_v P(v) \sum_{i \neq j} P(i|v) \cdot P(j|v)$$

C. Risultati di classificazione

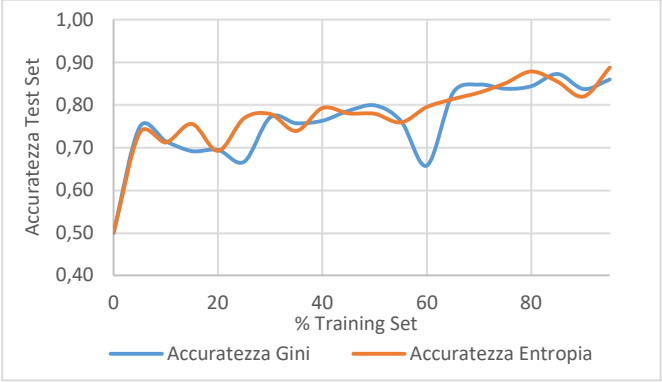

Figura IV.A: Confronto accuratezza tra ID3 e CART

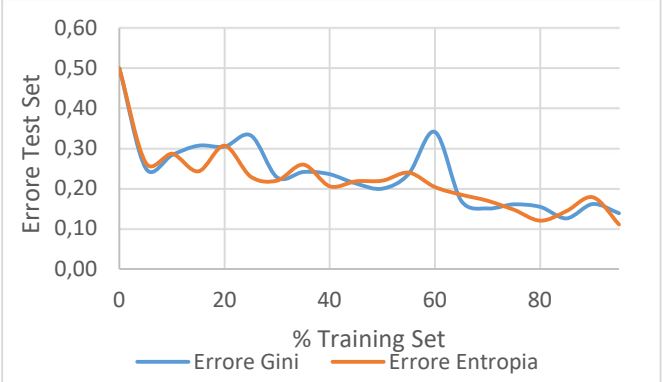

Figura IV.B: Confronto errore tra ID3 e CART

Applicati entrambi i criteri, si è fatto un confronto della loro accuratezza in funzione della porzione del dataset

utilizzata come training set e la restante porzione come test set. I risultati ottenuti (*Figure Figura IV.*A *e Figura IV.*B) mostrano come entrambi godano di una buona accuratezza già da un ridotto training set, e di conseguenza di un errore di classificazione non 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 potrebbero 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.

```
Test effettuati:        111
Test non classificati:  7

                    +-----------------------------+
                    |       CLASSE STIMATA        |
                    +-------------+---------------+
                    |  Negativo   |  Positivo     |
+--------+----------+-------------+---------------+
|CLASSE  | Negativo |     61      |       9       |
|        +----------+-------------+---------------+
|REALE   | Positivo |     6       |       28      |
+--------+----------+-------------+---------------+

Accuratezza:    0.8557692307692307
Errore:         0.14423076923076927
```

Figura IV.C: Matrice di confusione dell'algoritmo ID3 (test-set 15% degli esempi totali)

```
Test effettuati:        111
Test non classificati:  8

                    +-----------------------------+
                    |       CLASSE STIMATA        |
                    +-------------+---------------+
                    |  Negativo   |  Positivo     |
+--------+----------+-------------+---------------+
|CLASSE  | Negativo |     61      |       8       |
|        +----------+-------------+---------------+
|REALE   | Positivo |     5       |       29      |
+--------+----------+-------------+---------------+

Accuratezza:    0.8737864077669902
Errore:         0.12621359223300976
```

Figura IV.D: Matrice di confusione dell'algoritmo CART (test-set 15% degli esempi totali)

```
?- previsione([age = "Third", sex = "F", chest_pain_type = "ASY", restingBP = "Optimal"
, cholesterol = "Extremely high", fastingBS = 1, restingECG = "Normal", maxHR = 4, exer
cise_angina = "N", oldpeak = "Low risk", st_slope = "Flat"],Classe).
Il paziente ha il 100% di probabilità di essere malato!
Classe = y.

?- previsione([age = "Second", sex = "F", chest_pain_type = "ASY", restingBP = "Normal/
high", cholesterol = "Moderately high", fastingBS = 0, restingECG = "Normal", maxHR = 2
, exercise_angina = "Y", oldpeak = "Low risk", st_slope = "Up"],Classe).
Il paziente ha il 100% di probabilità di non essere malato!
Classe = n.
```

Figura IV.E: Previsioni con risultato certo

```
?- previsione([age = "Second", sex = "M", chest_pain_type = "ATA", restingBP = "Normal/
high", cholesterol = "Extremely high", fastingBS = 0, restingECG = "Normal", maxHR = 3,
 exercise_angina = "N", oldpeak = "Low risk", st_slope = "Up"],Classe).
Il paziente ha il 66.66666666666666% di probabilità di non essere malato!
Classe = n.
```

Figura IV.F: Previsione con risultato incerto

```
?- previsione([age = "First", sex = "M", chest_pain_type = "ATA", restingBP = "Normal/h
igh", cholesterol = "Extremely high", fastingBS = 0, restingECG = "Normal", maxHR = 3,
exercise_angina = "N", oldpeak = "Low risk", st_slope = "Up"],Classe).
Non è possibile definire lo stato di salute del paziente!
Classe = nc.
```

Figura IV.G: Previsione non classificabile

I dati estratti dalla matrice di confusione (Figura IV.CFigura IV.D) (test-set 15% degli esempi totali) vanno direttamente a costruire i grafici precedentemente illustrati. Mentre le previsioni vengono effettuate partendo dalle coppie Attributo Valore di un determinato paziente ed hanno come obiettivo quello di fornire la classe del paziente in esame, ovvero specificare se ha o meno problemi cardiovascolari. L'albero decisionale viene percorso in base alle coppie Attributo Valore del paziente partendo dalla radice fino ad arrivare ad una foglia dell'albero. Se nella foglia in esame gli esempi sono tutti della stessa classe, allora viene restituita la classe stessa come risultato della previsione, indicando una certezza del risultato dal punto di vista dell'albero decisionale, un esempio è riportato in *Figura IV.E*. Se invece nella foglia in esame sono presenti esempi con classe diverse tra loro significa che non c'è una certezza per indicare lo stato di salute del paziente. Si procede quindi riportando la classe dominante, indicando la probabilità che un esempio nella foglia appartenga ad essa, come mostrato *Figura IV.F*. Infine, il caso in cui l'esempio da valutare non trova spazio in nessuna delle foglie esistenti, sta ad indicare un'insufficiente quantità di informazione per descrivere il valore di un attributo; perciò, l'ipotetico caso che finisce in questa foglia viene valutato come "non classificabile", come mostrato in *Figura IV.G*. Ovviamente non si tratta di probabilità o certezze assolute in quanto c'è una forte dipendenza da come è stato costruito l'albero di decisione e da come sono stati manipolati in precedenza i dati; con test più ampi potremmo osservare che i modelli soffrono di *overfitting* nel generalizzare la classificazione, anche se nei precedenti risultati non ci sono segnali che lo lascino pensare.

V. Miglioramenti aggiuntivi

Ovviamente oggi giorno, con la ricerca costante posta a migliorare il campo del machine learning, si sono con il tempo sviluppati algoritmi di apprendimento automatico che definiscono dei modelli di reti neurali astratte veri e propri. Queste, grazie all'impatto del deep learning nelle attività di classificazione e dell'Image Processing in generale, ha portato un vero e proprio cambio di marcia per quanto riguarda i risultati attesi di tale processo.

Riprendendo i concetti esposti in questa relazione, dei miglioramenti aggiuntivi da applicare al progetto potrebbero ricercarsi sul fronte della classificazione. Infatti, un aspetto di fondamentale importanza da dover tenere in considerazione nella fase di costruzione dell'albero decisionale è che un albero particolarmente folto potrebbe non essere una struttura parecchio affidabile in termini predittivi. Proprio per questa ragione si prevede una fase di "potatura", in cui, tramite tecniche specifiche, vengono eliminati i rami che non aggiungono un significativo valore (informativo) all'albero. Quindi un aumento di performance dalla fase di classificazione potrebbe essere attuato anche grazie all'applicazione di queste tecniche di potatura.

Riferimenti

[1] Heart failure prediction dataset: "https://www.kaggle.com/fedesoriano/heart-failure-prediction".

[2] Riguzzi, Fabrizio, and Terrance Swift. "Well–definedness and efficient inference for probabilistic logic programming under the distribution semantics." Theory and practice of logic programming 13.2 (2013): 279-302.

[3] C3.js library optimized in Prolog: https://swish.swi-prolog.org/example/render_c3.swinb" – "https://c3js.org/"

[4] Heredia, Diana, Yegny Amaya, and Edwin Barrientos. "Student dropout predictive model using data mining techniques." IEEE Latin America Transactions 13.9 (2015): 3127-3134.

[5] Denison, David GT, Bani K. Mallick, and Adrian FM Smith. "A bayesian cart algorithm." Biometrika 85.2 (1998): 363-377.