

# Previsione di Sintomatologie Post-Dialisi

## Studio e realizzazione di un sistema supervisionato di estrazione regole

Francesco Pontillo  
Università degli Studi di Bari  
Dipartimento di Informatica  
Via E. Orabona, 4 - 70125 Bari, Italy  
francescopontillo@gmail.com

### ABSTRACT

Implementazione Prolog dell'algoritmo di Intelligenza Artificiale C4.5, con k-fold test e confronto con altri sistemi pre-esistenti di classificazione.

### 1. OBIETTIVO

Obiettivo del processo di Data Mining del sistema da sviluppare è di prevedere possibili sintomatologie successive ad una seduta di emodialisi. A partire da specifici dati registrati durante una dialisi, si vuole prevedere quali classi di sintomatologie il paziente potrà riscontrare dal momento in cui la dialisi termina al momento in cui esegue la seduta di dialisi successiva.

In questo modo, il medico può confermare la possibilità di occorrenza di una o più problematiche suggerite, ed eventualmente prescrivere una opportuna terapia per contrastare la sua insorgenza.

### 2. SELEZIONE DEGLI ATTRIBUTI

I dati a disposizione nella base di dati da analizzare sono numerosi, e devono essere selezionati appropriatamente per evitare l'introduzione di attributi poco rilevanti con lo scopo del sistema.

Ogni seduta di dialisi memorizza (1) una **data** di svolgimento, (2) la **durata** della seduta stessa, (3) un identificativo del **paziente**, (4) altri **parametri** registrati durante la sessione e (5) eventuali **sintomatologie** riscontrate.

#### 2.1 Dati del paziente

Le informazioni relative ai pazienti sono ricavate, anonimizzandole, dalla base di dati originale. Ai fini del processo di estrazione delle regole, è opportuno considerare il **sex** del paziente e la sua **età** al momento della seduta di dialisi in analisi<sup>1</sup>.

<sup>1</sup>Ciò non esclude la possibilità di considerare altri dati relati-

#### 2.2 Parametri della seduta di dialisi

I parametri più rilevanti di una seduta di dialisi, al fine di prevedere eventuali sintomatologie successive, sono divisi in più categorie [2] [3].

A. L'efficienza della rimozione dei prodotti di scarto è indotta dai valori dei parametri riportati in Tabella 1.

$KT/V$	indice di efficienza dialitica
$QB$	flusso di sangue trattato
$WS$	peso iniziale
$WE$	peso finale
$PWE$	peso finale ottimale
$PT$	durata ottimale
$T$	durata reale

Table 1: Parametri di efficienza eliminazione scarti

B. L'efficienza dell'eliminazione dell'acqua all'interno del corpo del paziente è indotta dai parametri in Tabella 2.

$SPS$	pressione sistolica iniziale
$SPE$	pressione sistolica finale
$DPS$	pressione diastolica iniziale
$DPE$	pressione diastolica finale
$BV$	volume ematico finale

Table 2: Parametri di efficienza eliminazione acqua

C. Altre tipologie di dati che potrebbero risultare utili a fornire previsioni significative sono riportati in Tabella 3.

$PBF$	flusso sangue teorico
$BF$	flusso sangue reale
$PUF$	ultrafiltrazione media teorica
$UF$	ultrafiltrazione media reale

Table 3: Altri parametri di efficienza dialitica

#### 2.3 Attributi derivati

A partire dalle informazioni disponibili nella base di dati, risulta evidente la presenza di alcuni attributi "nascosti" che possono essere più utili ai fini dell'apprendimento.

In tabella 4 sono elencati gli attributi derivati dalle precedenti tabelle; ad esempio,  $\Delta WL$  rappresenta la differenza vi al paziente; l'algoritmo da realizzare potrebbe essere este-

fra la perdita di peso programmata e quella effettiva, a sua volta calcolata come differenza fra peso iniziale e peso finale.

<i>PWL</i>	perdita peso programmata
<i>RWL</i>	perdita peso reale
$\Delta WL$	differenza perdita peso
$\Delta T$	differenza durata trattamento
<i>SPA</i>	pressione sistolica media
<i>DPA</i>	pressione diastolica media
$\Delta BF$	differenza flusso sangue
$\Delta UF$	differenza UF medio

Table 4: Parametri derivati

## 2.4 Sintomatologie

Il sistema verrà addestrato con istanze di esempio pre-classificate. La classificazione consiste nell'assegnazione, ad ogni esempio, di una o più categorie di sintomi, ad esempio: aritmia sintomatica, aritmia asintomatica, astenia, brividi, brividi e dispnea, cefalea, collasso ( $PA < 30\%$  inizio), conati di vomito, crampi, depressione, ansia, diarrea, dispnea e molti altri.

Inoltre, è prevista la classe 'asintomatico', che definisce una sintomatologia assente corrispondente ad un esempio negativo dal punto di vista della classificazione.

## 3. SELEZIONE DEI DATI

Le informazioni sottoposte all'algoritmo di apprendimento sono state selezionate a partire da una base dati molto ricca<sup>2</sup> e sono stati sottoposti ad una serie di passaggi<sup>3</sup>.

### 3.1 Creazione dei valori derivati

Per poter istanziare i valori degli attributi definiti in 2.3, è stata eseguita una query di tipo **SELECT** che preleva informazioni dalla tabella di origine ed effettua semplici calcoli di trasformazione.

In questo modo, alla fine del processo di trasformazione, gli attributi per ogni seduta di dialisi sono:

- **SESSION\_ID**, l'ID della seduta di dialisi, utile per identificare la seduta in ogni momento
- **SESSION\_DATE**, la data di esecuzione
- **KTV**, il valore di  $KT/V$
- **QB**, il valore di  $QB$
- **PROG\_WEIGHT\_LOSS**, la perdita peso programmata
- **REAL\_WEIGHT\_LOSS**, la perdita peso reale
- **DELTA\_WEIGHT**, la differenza fra la perdita di peso reale e quella programmata
- **PROG\_DURATION**, la durata programmata della dialisi

so andando a considerare anche i dati relativi alle malattie pregresse del paziente ed eventuali comorbidità registrate.

<sup>2</sup>Circa dal 1999 ai primi mesi del 2014.

<sup>3</sup>Tutte le trasformazioni e selezioni di dati descritte in que-

- **REAL\_DURATION**, la durata effettiva della dialisi
- **DELTA\_DURATION**, la differenza fra la durata reale e quella programmata
- **SAP\_START**, la pressione sistolica arteriosa prima della seduta
- **SAP\_END**, la pressione sistolica arteriosa dopo la seduta
- **AVG\_SAP**, la pressione sistolica arteriosa media
- **DAP\_START**, la pressione diastolica arteriosa prima della seduta
- **DAP\_END**, la pressione diastolica arteriosa dopo la seduta
- **AVG\_DAP**, la pressione diastolica arteriosa media
- **BLOOD\_VOLUME**, il volume di sangue trattato
- **DELTA\_BLOOD\_FLOW**, la differenza fra flusso di sangue teorico ed effettivo
- **DELTA\_UF**, la differenza dell'ultrafiltrazione media reale e teorica

Come si nota, sono stati eliminati alcuni attributi originali: il flusso di sangue teorico e reale e l'ultrafiltrazione media teorica e reale.

### 3.2 Associazione con sintomatologie

Nel programma che genera i dati, le sintomatologie vengono comunicate e quindi inserite, dal medico o dall'infermiere, qualche momento prima della dialisi successiva del paziente. Per poter mettere a confronto i dati della seduta di dialisi, di cui sopra, con i dati della sintomatologia rilevata, è stato necessario eseguire una query molto complessa per mettere in correlazione:

- il paziente
- la data di dialisi
- la data di dialisi minore fra quelle successive alla data di riferimento della seduta originaria

### 3.3 Associazione con dati del paziente

Infine, il dato della sintomatologia singola è stato associato univocamente con il paziente di riferimento, tramite l'apposito identificativo.

Tutte queste operazioni sono state eseguite staticamente, ovvero andando a creare una copia dei record in altre tabelle; ciò si è reso necessario in quanto, anche utilizzando macchine potenti, la selezione completa dei record impiegava interi minuti per completare, soprattutto a causa dell'associazione poco ottimizzata con le date (cfr. 3.2).

Questa sezione sono codificate nel `scripts/01-sql-server-tables.sql`.

### 3.4 Migrazione dei dati

Per una gestione più libera dei dati, si è scelto di migrare le tabelle create da Microsoft SQL Server a MySQL<sup>4</sup>, anche in ottica futura (cfr. 6).

## 4. PULIZIA DEI DATI

Una volta spostati i dati su un database MySQL, si è scelto di eliminare alcuni record e mantenerne altri più rilevanti<sup>5</sup>. La base dati originaria, infatti, contiene 185476 record.

### 4.1 Calcolo dello score

Ad ogni riga di rilevazione sintomo è stato associato un punteggio, o *score*, che permetta di capire quanto quella riga è completa (e quindi più o meno rilevante rispetto alle altre).

Fissato il numero degli attributi (di dialisi) a 15, un record con *score* più elevato sarà selezionato con più probabilità per avviare il processo di apprendimento.

### 4.2 Pazienti rilevanti

Inoltre, lo *score* è stato utilizzato anche per poter eliminare, dai record già selezionati, tutti quelli che appartengono a pazienti che hanno meno di 5 rilevazioni di sintomi con uno *score* percentuale più basso dell'80%.

Tutti i dati selezionati fino a questo punto, quindi, appartengono a pazienti che hanno almeno 5 rilevazioni di sintomi ottimali.

### 4.3 Gestione dei valori nulli

I valori nulli sono stati gestiti "staticamente", ovvero per ogni paziente sono state calcolate le medie dei valori di ogni attributo (ignorando quindi i valori nulli); in un passo successivo, sono stati scansionati tutti i record e, qualora fosse rilevato un valore nullo, è stato inserito il valore medio relativo al paziente associato.

Tutto ciò, tuttavia, ha portato comunque a mantenere alcuni valori nulli all'interno della base dati. Ad esempio, poche rilevazioni di sintomatologia contengono valori effettivi di KTV, probabilmente perché si tratta di una misura di difficile calcolo da parte dei medici.

Alcuni record, inoltre, non contenevano l'ID del sintomo target rilevato, e si è pertanto assunto che l'utente avesse erroneamente cancellato (dall'interfaccia del sistema) la dicitura "asintomatico", aggiungendo comunque una sintomatologia nulla (con ID uguale a 1).

## 5. APPRENDIMENTO DI REGOLE IN PROLOG

Per apprendere regole utili a classificare appositamente una seduta di dialisi si è scelto di implementare il funzionamento base dell'algoritmo C4.5 di Ross Quinlan [4] [5], in modo tale da:

<sup>4</sup>Lo script di migrazione è presente in `scripts/02-mysql-migration-script.sql` e viene richiamato in automatico, tramite appositi parametri di connessione, dal file batch `03-mysql-copy-migrated-tables.cmd`.

<sup>5</sup>Gli script rilevanti sono contenuti nel file `scripts/04-mysql-scores.sql`.

- utilizzare la sintomatologia come attributo target (da classificare)
- poter sfruttare i numerosi dati disponibili
- generare un albero di decisione
- convertire l'albero in un insieme di regole Prolog

Il programma Prolog è diviso in 6 moduli, ognuno dei quali si occupa di parti differenti del programma<sup>6</sup>.

Si è scelto di utilizzare SWI-Prolog[1] come ambiente Prolog.

### 5.1 Utility

Sono stati realizzati 2 moduli che realizzano funzioni di utilità.

#### 5.1.1 util.pl

`util.pl` contiene brevi regole che implementano:

- timer, con avvio, lettura e stop (`timer_start`, `timer_get`, `timer_stop`, tra gli altri)
- formattazione di millisecondi (`format_ms`) e secondi (`format_s`) nel formato intellegibile `{M}m {S}s {MS}ms`
- stampa a video di generiche liste di elementi o di un elemento singolo, con ritorno a capo (`println`)
- generici *helper* per liste, che realizzano funzioni di minimo e massimo (`list_min`, `list_max`), ricerca dell'elemento più comune (`list_most_common`) e dell'indice di un elemento specifico (`index_of`), oltre che funzionalità di aggiunta e rimozione di elementi
- concatenazione di elementi di una lista in una stringa
- realizzazione del logaritmo in base 2 (`log2`), utilizzato successivamente (cfr. 5.5).

### 5.2 Avvio del programma

Il programma principale è definito nel modulo `main.pl`, che si occupa del caricamento in memoria di tutti i file Prolog necessari e di definire il metodo `main(Config, Symptom)`:

- `Config` dichiara al programma qual è il file di configurazione con il quale si vuole accedere al database.<sup>7</sup> Vedi 5.3 per il l'accesso al database. Si è optato per un file di configurazione che contenesse tutti i parametri di connessione poiché la scrittura degli stessi ad ogni avvio del programma sarebbe risultata troppo verbosa.
- `Symptom` definisce l'ID del sintomo che si vuole utilizzare come esempio positivo per l'attributo target.

<sup>6</sup>Ogni regola definita nei diversi moduli è stata documentata. La documentazione è consultabile aprendo in un browser il file `doc/index.html`. Per un problema nel modulo di generazione della documentazione, i link diretti dalla `index.html` alle regole documentate non funzionano; utilizzare, invece, i collegamenti ai moduli, dai quali si può comunque accedere alla documentazione delle regole.

<sup>7</sup>Alcuni file di configurazione di esempio sono presenti nella

Entrambi i parametri supportano i meta-valori **default** e **ask**: **default** esegue un *fallback* dei parametri sul file di configurazione **prolog/config/database.properties** e sul sintomo con ID 2, mentre **ask** imposta il programma in modo da chiedere all'utente, in maniera interattiva e quando necessario, gli stessi parametri.

Sono anche presenti funzioni di avvio rapido: **main\_def/0** (che avvia il programma con parametri di default), **main/0** (che avvia in modalità **ask**) e **make\_doc/0** (che genera la documentazione HTML).

Per uscire dal programma e chiudere in maniera pulita la connessione, basta chiamare la regola **out**.

### 5.3 Lettura dal database

Una volta letti i parametri impostati dal **main** vengono eseguiti i processi di connessione e lettura dei dati utili alla generazione dell'albero di decisione.

La lettura dal database è possibile grazie alla libreria **odbc** di SWI-Prolog<sup>8</sup>.

#### 5.3.1 Connessione

La connessione al database avviene tramite la regola **connect** (nelle varianti) e i parametri nel file di configurazione impostato in precedenza: driver ODBC, indirizzo e porta, username, password e database. Se si è in modalità **ask**, il file di configurazione verrà chiesto all'utente, e nel caso in cui non esista viene eseguito un *fallback* sul file di default.

La lettura del file **.properties** è eseguita da

```
read_database_params(Path, Driver, Server,
    Port, Database, User, Password)
```

che utilizza il costrutto **open\_table** di SWI-Prolog per leggere i campi del file di proprietà e unificare con le variabili passate in input.

#### 5.3.2 Lettura sintomi

Il passo successivo consiste nella lettura di tutte le possibili sintomatologie che potrebbero verificarsi nel corso di una seduta di dialisi. La regola **get\_symptoms/0** esegue una semplice **SELECT** sulla base di dati, ottenendo, salvando in memoria e stampando a video i sintomi.

#### 5.3.3 Lettura degli esempi

La regola **update\_records/0** si occupa di ottenere e salvare in memoria tutti gli esempi che devono essere utilizzati per avviare il processo di apprendimento.

Poiché è necessaria la selezione sia degli esempi positivi che di quelli negativi, **update\_records/0** esegue lo stesso *state-*

cartella **prolog/config**

<sup>8</sup>Per questa ragione, è necessario che sulla macchina client siano installati i driver di connessione ODBC al database target. Poiché i parametri sono impostati dal programma Prolog, non è necessario creare nessuna connessione sulla macchina.

*ment* di **SELECT**, opportunamente creato e preparato, andando a modificare l'ID della sintomatologia target da ottenere<sup>9</sup>.

Il salvataggio dei record avviene andando ad asserire, nella memoria del programma Prolog, strutture del tipo:

```
positive(ID, Attribute, Value)
negative(ID, Attribute, Value)
```

In questo modo è sempre possibile accedere a qualsiasi coppia attributo-valore di un esempio con uno specifico ID, sia esso positivo o negativo.

È stata realizzata anche la regola

```
example(Type, ID, Attribute, Value)
```

dove **Type** può essere **positive** o **negative**. Tramite questa modalità è possibile accedere a tutti gli esempi prelevati dalla base di dati. Sono presenti, inoltre, regole di conteggio e di verifica di esistenza (cfr. documentazione HTML).

### 5.4 Suddivisione categorie in range

### 5.5 Apprendimento e test

## 6. SVILUPPI FUTURI

- Migliore gestione dei valori nulli (a runtime, tramite misura dell'information gain)
- Gestione multi-classe dell'albero di decisione
- Aggregazione dati ottimi da più database italiani e non (quando disponibili)

## 7. REFERENCES

- [1] Swi-prolog, Apr. 2014.
- [2] R. Bellazzi, C. Larizza, P. Magni, R. Bellazzi, and S. Cetta. Intelligent data analysis techniques for quality assessment of hemodialysis services. In *Proc. of the Workshop on Intelligent Data Analysis and Pharmacology*, 2001.
- [3] A. Kusiak, B. Dixon, and S. Shah. Predicting survival time for kidney dialysis patients: a data mining approach. *Comput. Biol. Med.*, 35(4):311–327, May 2005.
- [4] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [5] S. L. Salzberg. Book review: C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Mach. Learn.*, 16(3):235–240, Sept. 1994.

<sup>9</sup>La regola **get\_records/2** prepara lo statement all'esecuzione.