Previsione di Sintomatologie Post-Dialisi

Studio e realizzazione di un sistema supervisionato di estrazione regole

Francesco Pontillo
Universitá 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 é 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 potrá riscontrare dal momento in cui la dialisi termina al momento in cui esegue la seduta di dialisi successiva.

In questo modo, il medico puó confermare la possibilitá di occorrenza di una o piú 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, é opportuno considerare il **sesso** del paziente e la sua **etá** al momento della seduta di dialisi in analisi[1].

[1]Ció non esclude la possibilitá di considerare altri dati relativi al paziente; l'algoritmo da realizzare potrebbe essere este-

2.2 Parametri della seduta di dialisi

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

A. L'**efficienza della rimozione dei prodotti di scarto** é 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 é 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, ΔWL rappresenta la differenza

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

PWL	perdita peso programmata
RWL	perdita peso reale
ΔWL	differenza perdita peso
ΔT	differenza durata trattamento
SPA	pressione sistolica media
DPA	pressione diastolica media
ΔBF	differenza flusso sangue
ΔUF	differenza UF medio

Table 4: Parametri derivati

2.4 Sintomatologie

Il sistema verrá addestrato con istanze di esempio pre-classificate. La classificazione consiste nell'assegnazione, ad ogni esempio, di una o piú 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, é 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[2] e sono stati sottoposti ad una serie di passaggi[3].

3.1 Creazione dei valori derivati

Per poter istanziare i valori degli attributi definiti in 2.3, é 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
- `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, é 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; ció si é 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).

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

[2]Circa dal 1999 ai primi mesi del 2014.

[3]Tutte le trasformazioni e selezioni di dati descritte in questa sezione sono codificate nel `scripts/01-sql-server-tables.sql`.

3.4 Migrazione dei dati

Per una gestione piú libera dei dati, si é scelto di migrare le tabelle create da Microsoft SQL Server a MySQL[4], anche in ottica futura (cfr. 6).

4. PULIZIA DEI DATI

Una volta spostati i dati su un database MySQL, si é scelto di eliminare alcuni record e mantenerne altri piú rilevanti[5]. La base dati originaria, infatti, contiene 185476 record.

4.1 Calcolo dello score

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

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

4.2 Pazienti rilevanti

Inoltre, lo *score* é stato utilizzato anche per poter eliminare, dai record giá selezionati, tutti quelli che appartengono a pazienti che hanno meno di 5 rilevazioni di sintomi con uno *score* percentuale piú 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, é stato inserito il valore medio relativo al paziente associato.

Tutto ció, 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 é 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 é scelto di implementare il funzionamento base dell'algoritmo C4.5 di Ross Quinlan [4] [5], in modo tale da:

- 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 é diviso in 6 moduli, ognuno dei quali si occupa di parti differenti del programma[6].

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

5.1 Utility

Sono stati realizzati 2 moduli che realizzano funzioni di utilitá.

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 piú comune (`list_most_common`) e dell'indice di un elemento specifico (`index_of`), oltre che funzionalitá 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 é 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 é il file di configurazione con il quale si vuole accedere al database.[7]. Vedi 5.3 per il l'accesso al database. Si é 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.

[4]Lo script di migrazione é 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`.

[5]Gli script rilevanti sono contenuti nel file `scripts/04-mysql-scores.sql`.

[6]Ogni regola definita nei diversi moduli é stata documentata. La documentazione é 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 puó comunque accedere alla documentazione delle regole.

[7]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 modalitá `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 é possibile grazie alla libreria `odbc` di SWI-Prolog [8].

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 é in modalitá `ask`, il file di configurazione verrá chiesto all'utente, e nel caso in cui non esista viene eseguito un *fallback* sul file di default.

La lettura del file `.properties` é 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 proprietá 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 é necessaria la selezione sia degli esempi positivi che di quelli negativi, `update_records/0` esegue lo stesso *statement* di `SELECT`, opportunamente creato e preparato, andando a modificare l'ID della sintomatologia target da ottenere[9].

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

É stata realizzata anche la regola

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

dove `Type` puó essere `positive` o `negative`. Tramite questa modalitá é 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 attributi in range

Gli attributi dei dati di esempio possono essere numerici (a virgola mobile) o categorici, ma in ogni caso sono identificati da un numero. Essi sono stati dichiarati esplicitamente tramite la clausola `data_type(Attribute, Type)`

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 piú 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.

cartella `prolog/config`

[8] Per questa ragione, é necessario che sulla macchina client siano installati i driver di connessione ODBC al database target. Poiché i parametri sono impostati dal programma Prolog, non é necessario creare nessuna connessione sulla macchina.

[9] La regola `get_records/2` prepara lo statement all'esecuzione.