

POLITECNICO DI TORINO

DIPARTIMENTO DI INGEGNERIA GESTIONALE E DELLA
PRODUZIONE

Corso di Laurea in Ingegneria Gestionale Classe L8 –
Ingegneria dell'Informazione



Relazione dell'Esame Finale

Sviluppo di un'applicazione per la previsione e la gestione dell'occupazione dei posti letto in una struttura ospedaliera

Relatore

Prof. Fulvio Corno

Candidato

Arianna Ravera
237250

A.A. 2018/2019

Indice

1 Proposta di progetto

| | | |
|-----|------------------------------------------------------------------------------------------|---|
| 1.1 | Studente proponente. | 3 |
| 1.2 | Titolo della proposta. | 3 |
| 1.3 | Descrizione del problema proposto | 3 |
| 1.4 | Descrizione della rilevanza gestionale del problema | 3 |
| 1.5 | Descrizione dei data-set per la valutazione | 3 |
| 1.6 | Descrizione preliminare degli algoritmi coinvolti | 3 |
| 1.7 | Descrizione preliminare delle funzionalità previste per l'applicazione software | 4 |

2 Descrizione dettagliata del problema affrontato

| | | |
|-----|-------------------|---|
| 2.1 | Previsione | 5 |
| 2.2 | Simulazione | 6 |

3 Descrizione del data-set

| | | |
|-----|---------------------------------------------------------|---|
| 3.1 | Tabella dei ricoveri | 7 |
| 3.2 | Tabella dei posti massimi disponibili per reparto | 8 |

4 Strutture dati e algoritmi

| | | |
|-----|-------------|---|
| 4.1 | Model | 9 |
|-----|-------------|---|

5 Diagramma delle classi principali

6 Esempi di svolgimento

| | | |
|-----|------------------------------|----|
| 6.1 | Esempio : previsione | 15 |
| 6.2 | Esempio : simulazione | 17 |
| 6.3 | Link del video YouTube | 18 |

7 Valutazioni sui risultati ottenuti e conclusioni

1 Proposta di progetto

Studente proponente

s237250 Ravera Arianna

Titolo della proposta

Previsione e gestione dell'occupazione dei posti letto in una struttura ospedaliera

Descrizione del problema proposto

L'applicazione si propone di fornire una previsione dell'occupazione futura dei posti letto in un ospedale, pianificare la variazione di domanda di ricovero, facilitare la comunicazione interna all'ospedale e con altri ospedali e ridurre la cancellazione dei ricoveri per mancanza di posti letto. Inoltre, si vuole implementare un modo per simulare i ricoveri e avere statistiche sulle quantità di posti letto disponibili e occupati nei singoli reparti, valutando possibili soluzioni alla mancanza di disponibilità. I risultati dell'applicazione potrebbero aiutare nella gestione delle assegnazioni di letti ai pazienti ed evitare problemi di sovraffollamento o inutilizzo di risorse.

Descrizione della rilevanza gestionale del problema

Attualmente in molti ospedali vi è un forte problema di gestione degli spazi e dei letti per i pazienti, questo a causa delle scarse risorse assegnate alle sanità e della cattiva gestione del problema dal punto di vista organizzativo.

Attraverso un data-set dello storico dei ricoveri degli ultimi mesi su un ospedale di medie dimensioni, paragonabile alla maggior parte degli ospedali nel nostro Paese, con formule di forecasting sarà possibile avere previsioni delle occupazioni future dei posti letto. Indispensabile per riuscire a organizzare al meglio la gestione ed evitare problemi, tenendo conto anche di quante urgenze sono smaltibili ed ottimizzando l'accesso tempestivo dei pazienti al setting assistenziale più appropriato.

Descrizione dei data-set per la valutazione

Il data-set è stato fornito da un ospedale e consiste in un archivio dello storico dei ricoveri del suddetto ospedale. I dati sono così divisi: data di accettazione e inizio ricovero, data di dimissione e reparto.

I reparti in osservazione sono: cardiologia, chirurgia, ginecologia, medicina, neurologia, nido, ortopedia, pediatria, psichiatria, rianimazione, U.T.I.C. e urologia.

Descrizione preliminare degli algoritmi coinvolti

L'applicazione implementerà un algoritmo di previsione, per individuare la disponibilità di posti letto in singoli reparti e in dati periodi, e un algoritmo di simulazione basato sulla previsione che cerca la miglior soluzione per il ricovero.

Verranno anche effettuati controlli sull'integrità e la coerenza dei dati, con la presenza di messaggi di errore in caso di fallimento.

Descrizione preliminare delle funzionalità previste per l'applicazione software

L'applicazione software sarà un'interfaccia grafica composta da due diverse parti:

- La prima parte sarà dedicata alla previsione, dato un periodo di tempo limitato e un reparto verrà visualizzata la previsione dell'occupazione dei posti letto di quel reparto in quel periodo sia in forma tabellare che in forma grafica, attraverso un diagramma a torta.
- La seconda parte implementerà invece la funzione di simulazione, fornita una data di inizio ricovero, ed eventualmente anche di fine, e il reparto, verrà visualizzata la migliore soluzione possibile trovata per la sistemazione del paziente in quel periodo.

2 Descrizione dettagliata del problema affrontato

L'introduzione di normative stringenti in materia di Gestione dei Posti Letto unite alle necessità di ottimizzazione richieste a livello regionale e nazionale necessitano di essere affrontate sia da un punto di vista amministrativo e gestionale, sia come una situazione a forte impatto sul processo di cura del paziente.

L'applicazione è volta ad occuparsi del problema dal primo dei due punti di vista, è la soluzione ideale per chi desidera raggiungere obiettivi quali:

- aumento dell'efficienza all'interno del proprio reparto e ospedale, e della comunicazione con altri ospedali
- ottimizzazione dell'accesso tempestivo dei pazienti al setting assistenziale più appropriato in base al proprio percorso di cura
- riduzione della cancellazione dei ricoveri per mancanza di posti letto
- stabilizzazione della domanda di ricovero (urgente e programmato) in funzione dei letti disponibili
- pianificazione della variazione di domanda di ricovero

Per garantire ciò si agisce su due differenti livelli: visualizzazione real-time dello stato di occupazione dei posti letto e proiezioni delle degenze sui singoli posti letto, elaborate in base ad algoritmi di previsione applicati ai periodi precedenti.

In seguito, verranno analizzate nel dettaglio le due fasi in cui si struttura l'applicazione.

2.1 Previsione

Per la fase di previsione dell'occupazione, dopo un accurato studio dei dati e della loro distribuzione nel tempo, si è scelto di adottare il metodo dell'Exponential Smoothing.

Il livellamento esponenziale è una tecnica empirica per livellare i dati delle serie temporali utilizzando la funzione di finestra esponenziale. È il metodo adatto in assenza di trend; il peso attribuito diminuisce esponenzialmente per le osservazioni passate.

$$F(t) = \alpha A(t) + (1 - \alpha) F(t - 1)$$

$$f(t + \tau) = F(t)$$

$$F(0) = A(0)$$

Scelto un τ di una settimana per le previsioni da luglio a dicembre, e di un mese per la previsione di gennaio, si necessita dei seguenti dati di input:

- storico dell'occupazione dei reparti di cui l'ospedale tiene traccia in un data-set opportuno;
- α , detto parametro di lisciamiento.

Viene inoltre calcolata la Mean Squared Deviation (MSD), una delle numerose misure per valutare l'accuratezza delle previsioni, quadrando la deviazione di previsione individuale (errore) per ciascun periodo e quindi individuandone il valore medio della somma degli errori al quadrato.

$$MSD = \sum_{t=1}^n \frac{(f(t) - A(t))^2}{n}$$

L'errore di previsione è l'osservazione effettiva per un periodo meno la previsione per il periodo.

L'errore quadratico medio viene utilizzato perché, quadrando i valori di errore, i valori risultanti sono tutti positivi, le deviazioni quadratiche medie sono più facili da gestire matematicamente e pertanto vengono spesso utilizzate nell'ottimizzazione statistica.

In output saranno quindi visualizzati i risultati sopra descritti e due diverse tipologie di grafico:

- un grafico a linee rappresentante l'andamento dell'occupazione sui diversi giorni richiesti;
- un areogramma rappresentante la percentuale di posti mediamente occupati nel reparto e nel periodo in analisi, rispetto alla capienza massima.

2.2 Simulazione

La seconda fase consiste invece nella simulazione di un futuro ricovero. Attraverso l'algoritmo sopradescritto si ottiene una stima dell'occupazione del reparto e, confrontandola con la reale e certa occupazione in quelle stesse date, si valuta la possibilità di accettare il suddetto ricovero.

I casi che si possono presentare sono di tre diversi tipi:

- non vi sono problemi di capienza né per quanto riguarda l'occupazione certa, né per quanto riguarda la previsione, per cui si può procedere con la prenotazione del ricovero o al ricovero stesso;
- si ipotizza, secondo i dati della previsione, possa esserci un problema di sovraffollamento in alcune delle date scelte; sotto questa ipotesi starà al personale addetto decidere come intervenire nel caso in cui ci possano poi essere problemi nei giorni fissati;
- si è certi che il reparto sia al completo per almeno uno dei giorni indicati nella simulazione; in tal caso se il ricovero viene valutato, dal personale e dal paziente, come posticipabile si procede a cercare un lasso temporale disponibile, altrimenti l'ospedale sarà costretto a dover rifiutare il ricovero prendendosi le responsabilità del caso.

Quindi, oltre ai dati già descritti per la funzione di previsione, in input l'algoritmo di simulazione rende accessibile una choiceBox per la scelta del numero di giorni di cui posticipare il ricovero, ma questo solo nell'ultimo dei tre casi previsti.

In output l'applicazione presenterà:

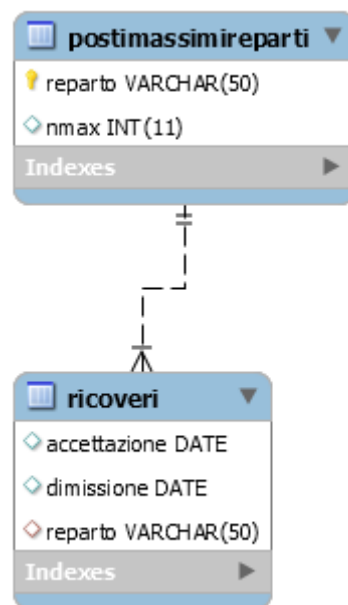
- l'occupazione certa del reparto, basata sui ricoveri già in atto o programmati;
- la previsione dell'occupazione;
- tutte le possibili soluzioni applicabili nel caso in cui si verificassero sovraffollamenti dei reparti.

3 Descrizione del data-set

Il data-set utilizzato è un derivato di un file CSV fornitomi da un'azienda ospedaliera locale riguardante lo storico dei ricoveri iniziati dal 01/06/2018 al 31/12/2018 in 12 diversi reparti dell'ospedale, e la capienza massima dei reparti.

Le due tabelle presenti sono:

- la tabella "ricoveri" nella quale sono presenti le date di inizio e fine ricovero ed il reparto;
- la tabella "postimassimireparti" dove per ogni reparto è indicata la capienza.



3.1 Tabella ricoveri


Nell'immagine sottostante sono riportate le colonne della tabella "ricoveri".

| # | Nome | Tipo di dati |
|---|--------------|--------------|
| 1 | accettazione | DATE |
| 2 | dimissione | DATE |
| 3 | reparto | VARCHAR |

1. *accettazione*: Data di inizio ricovero.
2. *dimissione*: Data di fine ricovero.
3. *reparto*: Nome del reparto del ricovero.

3.2 Tabella dei posti massimi disponibili per reparto

Nell'immagine sottostante sono riportate le colonne della tabella "postimassimireparti".

| # | Nome | Tipo di dati |
|-------------------------------------------------------------------------------------|----------------|----------------|
|  1 | reparto | VARCHAR |
| 2 | nmax | INT |

1. *reparto*: Nome del reparto.

2. *nmax*: Numero di posti letto del reparto.

4 Struttura dati e algoritmi

Il software è suddiviso in tre packages:

- Controller: contiene la classe Main per l'avvio dell'applicazione, le due classi controller (*RicoveriController* e *PieChartController*) che definiscono i metodi che permettono l'interazione dell'utente e i file .fxml (*PostiLetto* e *PieChart*) per la definizione dell'interfaccia grafica.
- DB: contiene la classe DBConnect utilizzata per la connessione al database e la classe DatiDAO che, tramite query SQL, permette il caricamento dei dati presenti nel database all'interno dell'ambiente Java.
- Model: contiene la classe omonima *Model*, che possiede tutta la logica applicativa del software, la classe *Dati*, che definisce la struttura dei dati usati, e le classi *Row* e *Row2*, utili per la rappresentazione grafica dei dati. Infine, la classe *TestModel* per il controllo dell'efficienza delle varie funzioni del Model.

Nella sezione successiva, una descrizione ad alto livello delle principali strutture dati e degli algoritmi implementati.

4.1 Model

```
▼ 📁 Model
  ▫ datidao : DatiDAO
  ▫ n : float
  ▫ previsioni : Map<LocalDate, Double>
  ▫ domanda : Map<LocalDate, Integer>
  ▫ msd : Map<LocalDate, Double>
  ▫ previsioniRicovero : Map<LocalDate, Double>
  ▫ domandaRicovero : Map<LocalDate, Integer>
  ▫ post : boolean
  ▫ colore : char
  ● 📄 Model()
  ● creaPrevisioni(String, double) : void
  ● occupazioneReparto(LocalDate, LocalDate, String) : Map<LocalDate, Integer>
  ● 📄 creaPrevisioniDiGennaio(String, Double) : void
  ● 📄 previsione(LocalDate, LocalDate, String) : List<Row>
  ● datiTortaPrev() : float
  ● simulazione(LocalDate, LocalDate, String) : List<Row2>
  ● dolpotesi(List<Row2>, String) : String
  ● posticipa() : boolean
  ● occMax(String) : Integer
  ● getColore() : char
```

La classe Model presenta i seguenti attributi:

_ datidao - istanza della classe DAO per ricavare i dati del DB utili per l'implementazione.

_ n - float della percentuale di occupazione utile a riempire il PieChart.

_ previsioni e previsioneRicoveri - mappe delle previsioni suddivise per data e utili per i due diversi algoritmi.

- _ domanda e domandaRicoveri - mappe dell'occupazione reale suddivisa per data e utili per i due diversi algoritmi.
- _ msd - mappa dell'msd calcolato per ogni giorno tra previsione e occupazione reale.
- _ post - indicatore della possibilità di posticipazione del ricovero.
- _ colore - carattere del colore dell'output variabile a seconda del risultato della simulazione.

Ed i seguenti metodi:

- _ creaPrevisioni() - Metodo che implementa gli attributi previsione e msd attraverso gli algoritmi descritti al punto 2.1.

```
public void creaPrevisioni(String reparto, double alfa) {
    msd=new HashMap<LocalDate, Double>();
    previsioni=new HashMap<LocalDate, Double>();
    domanda=this.occupazioneReparto(LocalDate.of(2018, 06, 1), LocalDate.of(2018, 12, 31), reparto);
    double err =0.0;
    int numero=0;

    //per giugno F(t)=A(t)
    LocalDate inizioGiugno=LocalDate.of(2018, 06, 1);
    LocalDate fineGiugno=LocalDate.of(2018, 06, 30);

    while(inizioGiugno.isBefore(fineGiugno)||inizioGiugno.equals(fineGiugno)) {
        //do
        previsioni.put(inizioGiugno, (double)domanda.get(inizioGiugno));
        //incremento
        inizioGiugno=inizioGiugno.plusDays(1);
    }

    //da luglio a gennaio uso F(t-1) quindi quelle della settimana precedente
    LocalDate inizio=LocalDate.of(2018, 07, 1);
    LocalDate fine=LocalDate.of(2018, 12, 31);

    while(inizio.isBefore(fine)||inizio.equals(fine)) {
        //do
        double f=alfa*domanda.get(inizio) + (1-alfa)*previsioni.get(inizio.minusWeeks(1));
        previsioni.put(inizio, f);
        err+=Math.pow(previsioni.get(inizio)-domanda.get(inizio), 2);
        numero++;
        err=err/numero;
        msd.put(inizio, err);
        //incremento
        inizio=inizio.plusDays(1);
    }
}
```

- _ occupazioneReparto() - Metodo che implementa l'attributo domanda interrogando la classe DAO.

```
public Map<LocalDate, Integer> occupazioneReparto(LocalDate in, LocalDate fin, String reparto) {
    Map<LocalDate, Integer> occ=new TreeMap<LocalDate, Integer>();

    while(in.isBefore(fin)||in.equals(fin)) {
        //do
        occ.put(in, datidao.occupazioneRepartoPrev(in, reparto));
        //incremento
        in=in.plusDays(1);
    }
    return occ;
}
```

_ creaPrevisioniDiGennaio() - Metodo che implementa l'attributo previsioniRicovero su base mensile invece che settimanale.

```
public void creaPrevisioniDiGennaio(String reparto, Double alfa) {
    previsioniRicovero=new HashMap<LocalDate, Double>();
    domandaRicovero=this.occupazioneReparto(LocalDate.of(2018, 12, 1), LocalDate.of(2019, 01, 31), reparto);

    //per giugno F(t)=A(t)
    LocalDate inizioDic=LocalDate.of(2018, 12, 1);
    LocalDate fineDic=LocalDate.of(2018, 12, 31);

    while(inizioDic.isBefore(fineDic)||inizioDic.equals(fineDic)) {
        //do
        previsioniRicovero.put(inizioDic, (double)previsioni.get(inizioDic));
        //incremento
        inizioDic=inizioDic.plusDays(1);
    }

    //da luglio a gennaio uso F(t-1) quindi quelle del mese precedente
    LocalDate inizio=LocalDate.of(2019, 01, 1);
    LocalDate fine=LocalDate.of(2019, 01, 31);

    //trovo alfa migliore
    while(inizio.isBefore(fine)||inizio.equals(fine)) {
        //do
        double f=alfa*domandaRicovero.get(inizio) + (1-alfa)*previsioniRicovero.get(inizio.minusMonths(1));
        previsioniRicovero.put(inizio, f);
        //incremento
        inizio=inizio.plusDays(1);
    }
}
```

_ previsione() - Metodo che restituisce una Lista di Row contenenti, per ogni giorno nell'intervallo di date selezionato, la previsione, l'occupazione reale e l'msd.

```
public List<Row> previsione(LocalDate inizio, LocalDate fine, String reparto) {
    // stampare previsione e fare i due grafici

    List<Row> previsione=new ArrayList<Row>();
    int nmax=datidao.getNMaxPosti(reparto);
    int somma=0;
    int ngiorni=0;
    LocalDate in=inizio;

    //do previsione
    while(in.isBefore(fine)||in.equals(fine)) {
        Row r= new Row(in,previsioni.get(in),domanda.get(in),msd.get(in));
        previsione.add(r);
        in=in.plusDays(1);
    }

    //per il grafo, occupazione media del reparto nel periodo
    in=inizio;
    while(in.isBefore(fine)||in.equals(fine)) {
        //do
        somma+=previsioni.get(in);
        ngiorni++;
        //incremento
        in=in.plusDays(1);
    }
    n=((float)((float)somma/ngiorni)/nmax)*100;
    return previsione;
}
```

_ datiTortaPrev() - Metodo che restituisce n, percentuale media di occupazione del reparto e nel periodo selezionati.

_ simulazione() - Metodo che restituisce una Lista di Row2 contenenti, per ogni giorno nell'intervallo di date selezionato, la previsione e l'occupazione reale.

```
public List<Row2> simulazione(LocalDate inizio, LocalDate fine, String reparto) {
    List<Row2> previsione=new ArrayList<Row2>();
    LocalDate in=inizio; //piu o meno i giorni che sono disposto a posticipare o anticipare etc

    //do previsione
    while(in.isBefore(fine)||in.equals(fine)) {
        Row2 r= new Row2(in,previsioniRicovero.get(in),domandaRicovero.get(in));
        previsione.add(r);
        in=in.plusDays(1);
    }
    return previsione;
}
```

_ doIpotesi() - Metodo che crea e restituisce ipotesi sulle soluzioni possibili del caso in esame.

```
public String doIpotesi(List<Row2> simulazione, String reparto) {
    Map<LocalDate, Double> situazionePrevista=new HashMap<LocalDate, Double>();
    Map<LocalDate, Integer> situazioneCerta=new HashMap<LocalDate, Integer>();
    List<LocalDate> dateRischio=new ArrayList<LocalDate>();
    List<LocalDate> dateProblema=new ArrayList<LocalDate>();
    int max=datidao.getNMaxPosti(reparto);
    String result="";
    post=false;
    int prob=0;
    int probCerto=0;

    for(Row2 r:simulazione) {
        situazionePrevista.put(r.getData(), (r.getPrevisione())); //giorno e posti liberi secondo domanda+previsione
        if(r.getPrevisione()>=max) {
            dateRischio.add(r.getData());
            prob++;
        }
    }
    for(Row2 r:simulazione) {
        situazioneCerta.put(r.getData(), (r.getDomanda())); //giorno e posti liberi secondo domanda
        if(r.getDomanda()>=max) {
            dateProblema.add(r.getData());
            prob++;
            probCerto++;
        }
    }

    if(prob==0) {
        colore='v';
        result="ACCETTATO.\nNon sono previsti problemi di sovraffollamento del reparto "+reparto.toLowerCase()+" per le date indicate.\n";
    }
    else {
        String r="";
        if(probCerto==0) {
            for(LocalDate ld:dateRischio)
                r+="\n"+ld;
            colore='g';
            result="ACCETTATO.\nIl ricovero viene accettato in quanto l'occupazione certa non supera la capienza massima.\nBisogna però prestare"
                + "attenzione perchè seconda la previsione ci saranno problemi di sovraffollamento nelle date:"+r;
        }
        else{
            for(LocalDate ld:dateProblema)
                r+="\n"+ld;
            colore='r';
            post=true;
            result="PROBLEMA!\nNon ci sono posti disponibili nelle date:"+r+"\nSe possibile, provare a posticipare il ricovero di alcuni giorni o"
                + "selezionare altre date.\nNel caso in cui questo non sia possibile ci troviamo costretti a dover rifiutare il ricovero.";
        }
    }
    return result;
}
```

_ posticipa() - Metodo che restituisce il valore di post.

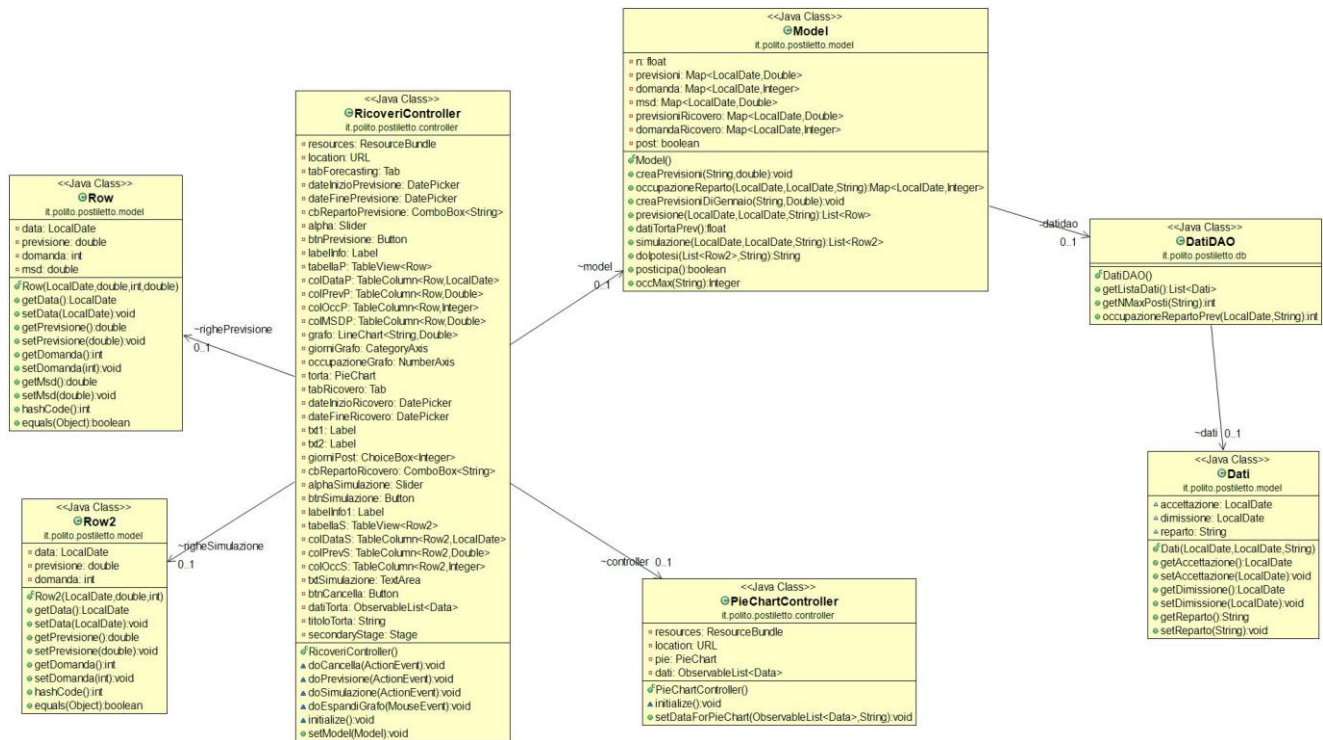
_ occMax() - Metodo che restituisce la capienza del reparto ottenuta tramite interrogazione della classe DAO.

_ getColore() - Metodo che restituisce il valore di colore.

5 Diagramma delle classi principali

L'applicazione è sviluppata secondo il pattern MVC (Model-View-Controller), nel quale la classe Model gestisce la logica del programma, la classe RicoveriController si occupa dell'interazione con l'utente e la classe Dati DAO gestisce l'interazione con il database.

Di seguito il diagramma delle classi principali.



Sono state inserite in figura solamente le variabili e i metodi delle classi principali, i quali giocano un ruolo cruciale all'interno dell'applicazione. Nella cartella di progetto sono presenti anche altri classi fondamentali al funzionamento finale del programma ma che rientrano in modo marginale nell'algoritmo portante.

6 Videate dell'applicazione

POSTI LETTO

Previsione

Ricovero

Data di inizio previsione:

Reparto:

Data di fine previsione:

Apha della previsione:

00,250,50,751

Avvia la previsione

| Data | Previsione | Occupazione reale | MSD |
|--------------------------------|------------|-------------------|-----|
| Nessun contenuto nella tabella | | | |

Occupazione

1251007550250

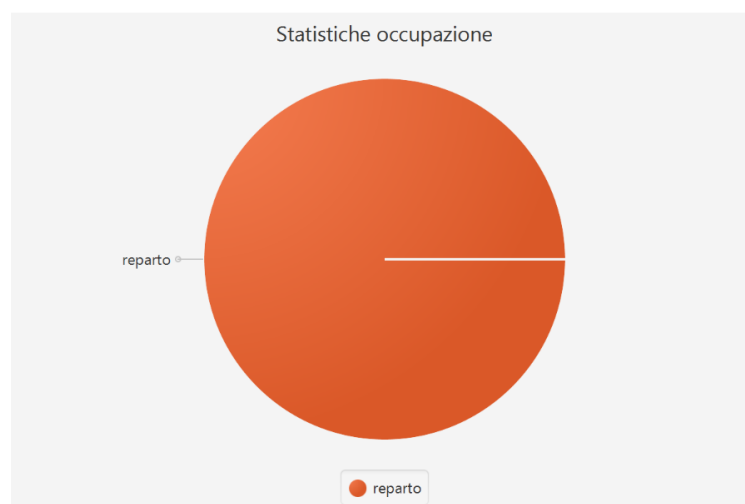
Giorni

Statistiche occupazione

reparto

Cancel

Videata "Previsione"



PieChart della previsione

POSTI LETTO

Previsione Ricovero

Data di inizio ricovero:

Data di fine ricovero:

Reparto del ricovero:

Alpha della previsione: 0 0,25 0,5 0,75 1

Avvia la simulazione

| Data | Previsione | Occupazione attuale |
|--------------------------------|------------|---------------------|
| Nessun contenuto nella tabella | | |

Cancella

Videata "Ricovero"

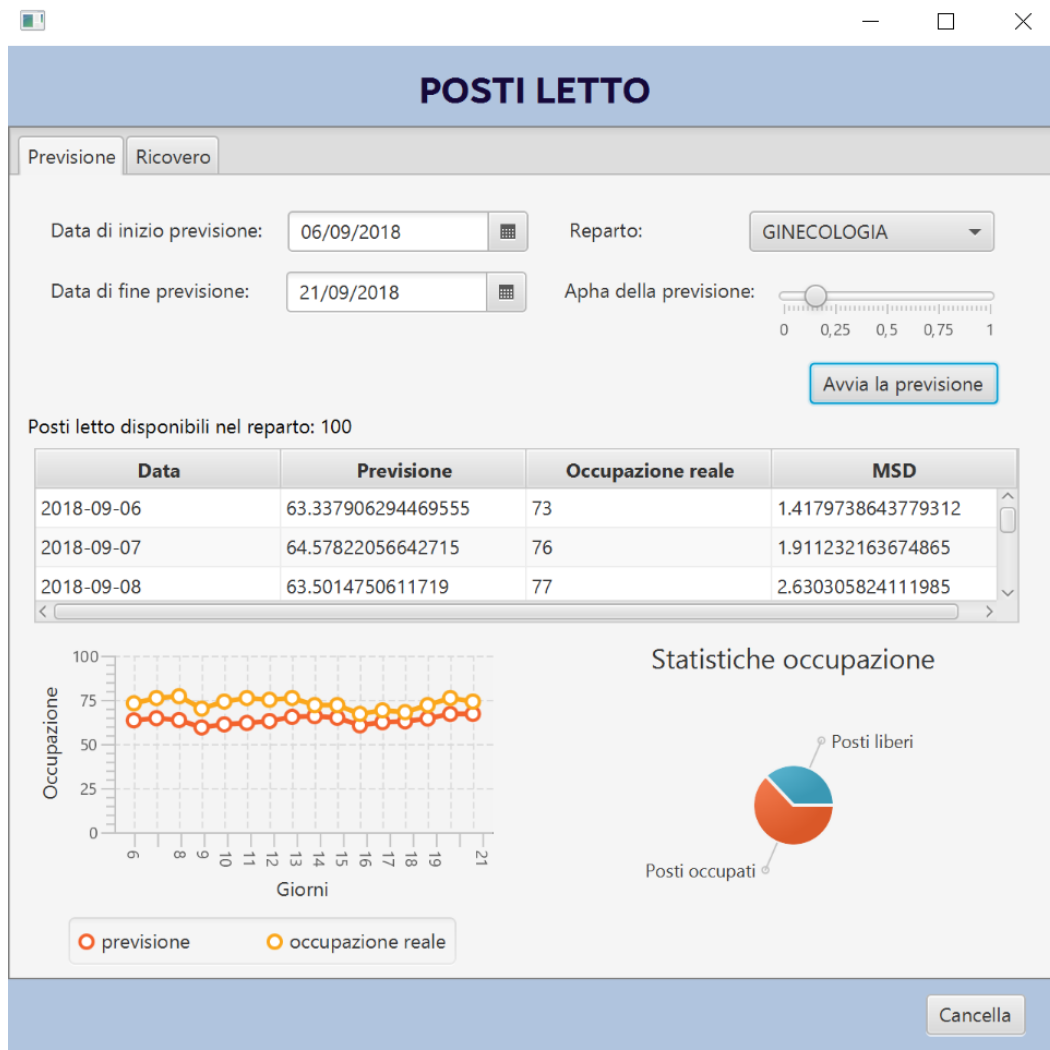
Di seguito la spiegazione dei passaggi principali per il funzionamento dell'applicazione.

6.1 Esempi di svolgimento: previsione

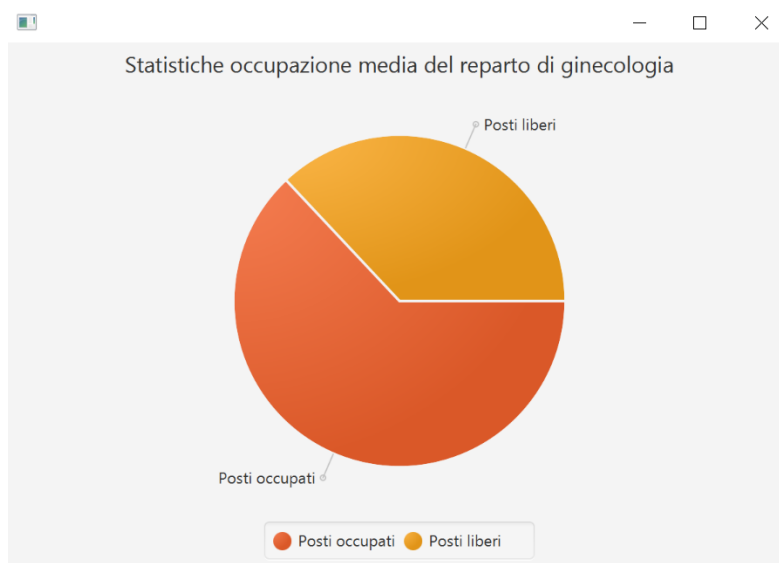
L'utente deve scegliere un lasso temporale di previsione tra le date possibili, dal 1° Luglio 2018 al 31 Dicembre 2018, un reparto tra i dodici disponibili nel data-set e l' α della previsione.

I valori di α vicini ad 1 hanno meno effetto levigante e danno maggior peso alle recenti variazioni nei dati, mentre i valori di α più vicini allo zero hanno un maggiore effetto levigante e sono meno sensibili ai cambiamenti recenti. Non esiste una procedura formalmente corretta per la scelta di α .

Se vengono specificati correttamente tutti i dati, il programma restituisce l'andamento dell'occupazione dei posti letto del suddetto reparto, la sua previsione e l'MSD associato alla previsione effettuata; altrimenti verrà visualizzato un messaggio di errore.



Esempio di previsione



Esempio di PieChart

6.2 Esempi di svolgimento: simulazione del ricovero

Anche in questa parte all'inizio è richiesto all'utente di scegliere un lasso temporale in cui desidera effettuare il ricovero nel mese che si presume essere il mese corrente, quindi tra il 1° Gennaio 2019 e il 31 Gennaio 2019, un reparto tra i dodici e l' α della previsione.

In questo caso si consiglia di usare valori di α più vicini allo zero per avere una previsione più realistica, in quanto se si sceglie un α più vicino ad 1 i dati saranno troppo bassi perché condizionati dall'occupazione reale prevista nel mese corrente e quindi solo quella certa al momento, che comprende esclusivamente ricoveri già in corso o programmati.

Specificati tutti i dati, il programma restituisce l'andamento dell'occupazione dei posti letto del suddetto reparto, la sua previsione e una breve spiegazione del risultato. Nel caso in cui si verifichi un sovraffollamento certo o previsto, vengono valutate tutte le possibili soluzioni ed eventualmente viene richiesto il numero di giorni di cui è possibile posticipare il ricovero.

POSTI LETTO

Previsione

Ricovero

Data di inizio ricovero: 01/01/2019

Reparto del ricovero: ORTOPEDIA

Data di fine ricovero: 04/01/2019

Apha della previsione: 0 0,25 0,5 0,75 1

Selezionare il numero di giorni di cui posticipare del ricovero:

Avvia la simulazione

Posti letto disponibili nel reparto: 146

| Data | Previsione | Occupazione attuale |
|------------|--------------------|---------------------|
| 2019-01-01 | 132.86373258039242 | 146 |
| 2019-01-02 | 133.34688619694737 | 147 |
| 2019-01-03 | 124.61428363276787 | 110 |
| 2019-01-04 | 124.05883025881243 | 109 |

PROBLEMA!

Non ci sono posti disponibili nelle date:

2019-01-01

2019-01-02

Se possibile, provare a posticipare il ricovero di alcuni giorni o selezionare altre date.

Nel caso in cui questo non sia possibile ci troviamo costretti a dover rifiutare il ricovero.

Cancella

Esempio di simulazione

POSTI LETTO

Previsione
Ricovero

Data di inizio ricovero:

Data di fine ricovero:

Selezionare il numero di giorni di cui posticipare del ricovero: 2

Reparto del ricovero: ORTOPEDIA ▼

Apha della previsione:

0 0,25 0,5 0,75 1

Avvia la simulazione

Posti letto disponibili nel reparto: 146

| Data | Previsione | Occupazione attuale |
|------------|--------------------|---------------------|
| 2019-01-03 | 124.61428363276787 | 110 |
| 2019-01-04 | 124.05883025881243 | 109 |
| 2019-01-05 | 122.95179429302249 | 106 |
| 2019-01-06 | 121.95302255972354 | 106 |

ACCETTATO.

Non sono previsti problemi di sovraffollamento del reparto ortopedia per le date indicate.

Cancella

Esempio di simulazione con posticipazione

6.3 Link del video YouTube

Il link al video di presentazione dell'applicazione è:

<https://youtu.be/x6ZqOJrP3Q>

7 Valutazioni sui risultati e conclusioni

L'applicativo implementa due funzionalità: la previsione e la simulazione.

Nel primo caso, il software agisce sostanzialmente da verifica nel caso si abbiano già i dati dell'occupazione reale del periodo indicato che può andare dal 01-07-2018 al 31-12-2018.

Nel secondo caso invece, si vuole cercare di far fronte ad una situazione attuale, e quindi di maggiore incertezza, dando un'indicazione sulla possibilità del verificarsi di problemi di sovraffollamento nel reparto. Le simulazioni hanno come prima data il 01-12-2019 perché ci poniamo nel caso in cui questa sia la data odierna.

Attraverso lo sviluppo di questa applicazione ho voluto tentare di creare un software riguardante la gestione dei posti letto in quanto questo è un problema reale molto importante e discusso al giorno d'oggi.

La causa principale del sovraffollamento dei Pronto Soccorso è il blocco dell'uscita, cioè l'impossibilità di ricoverare i pazienti nei reparti degli ospedali per indisponibilità di posti letto, dopo il completamento della fase di cura in Pronto Soccorso.

Questo fenomeno è largamente conosciuto in tutto il territorio nazionale e in molti altri Paesi dotati di sistemi sanitari avanzati, e si presenta sia negli ospedali delle aree metropolitane con la loro pluralità di offerta di presidi ospedalieri anche iperspecialistici, sia nei contesti rurali o montani.

Il sovraffollamento è quindi un problema rilevante che impatta sulla qualità dell'offerta sanitaria sia in termini di sicurezza e qualità delle cure ai pazienti, che di benessere psicofisico degli operatori. Esso richiede perciò una risposta diversificata attraverso la sperimentazione di iniziative e di soluzioni organizzative integrate e innovative.

Si noti che oggi gran parte del bilancio regionale viene assorbito dalle spese sanitarie, se le regioni vanno in deficit di fatto non possono non evitare di ridurre la spesa sanitaria perché questa rappresenta la principale voce del loro bilancio e le conseguenze di ciò sono disastrose.

L'azione a contrasto di tale fenomeno, deve tendere a favorire la realizzazione di una rete di offerta con interventi articolati su più fronti, ma strettamente interconnessi tra loro.

L'applicazione è quindi da considerarsi un aiuto ad un utente qualificato che si accinga allo studio della gestione dell'occupazione dei reparti all'interno dell'ospedale, cercando quindi di far fronte preventivamente al problema e ci ottimizzare al meglio tutte le risorse della struttura per evitare il problema. Si adatta facilmente a diversi tipi di data-set, su differenti spazi temporali e reparti, ed è semplice da capire e usare. Essa è una versione primaria, che rispecchia le conoscenze e le competenze che ho attualmente acquisito nel mio corso di studi, quindi ancora semplificata in alcuni punti.

Il tempo medio di esecuzione del programma è di 3/4 secondi per entrambe le funzionalità implementate, un dato confortevole e continuo al variare di parametri forniti; questo dimostra una buona stabilità dell'algoritmo.

Un altro punto di forza dell'applicazione è che si fonda su dati reali e ciò rende verosimili i risultati ottenuti.

A sfavore va sicuramente considerato che la scelta di α compromette abbastanza la previsione e in una situazione reale non sarebbe consigliabile lasciare libera scelta del parametro per avere la più ottima delle previsioni. Purtroppo, la mia scelta a tal proposito è stata forzata dalle dimensioni

molto contenute del mio data-set che considera i ricoveri iniziati dal 01-06-2018 al 31-12-2018, escludendomi quindi la possibilità di una previsione su base annuale.

Nel complesso l'obiettivo si può considerare raggiunto, bisogna tuttavia tener conto che, in un caso reale, andrebbero comunque considerati numerosi altri aspetti come la gravità dello stato del paziente, l'urgenza del ricovero, il possibile trasferimento in altri reparti, gli accordi presi con altre strutture ospedaliere, le normative vigenti sul territorio, etc.

Attribuzione - Non commerciale - Non opere derivate 4.0 Internazionale



Quest'opera è distribuita con Licenza Creative Commons Attribuzione - Non commerciale - Non opere derivate 4.0 Internazionale. Per leggere una copia della licenza visita il sito web <http://creativecommons.org/licenses/by-nc-nd/4.0/>.