UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

DOCUMENTAZIONE DI PROGETTO – CORSO DI INGEGNERIA DELLA CONOSCENZA

A.A. 2022-2023

EM per la classificazione in presenza di dati incompleti all'interno di ontologie

Christian Riefolo
Matricola: 618687
E-Mail: c.riefolo2@studenti.uniba.it
URL progetto: https://github.com/THESCREAMINGMONKEY/MBM-EM.git

INDICE:

INTRODUZIONE 3

Sommario 3

Il dataset 3

Multivariate Bernoulli Model 4

Elenco argomenti di interesse tratti dal programma 5

IMPLEMENTAZIONE ALGORITMO EXPECTATION-MAXIMIZATION 6

Sommario 6

Strumenti utilizzati 6

Decisioni di Progetto 7

Inizializzazione: *7*

Iterazioni: *8*

VALUTAZIONE E DISCUSSIONE RISULTATI 8

CONCLUSIONI 10

Riferimenti Bibliografici 11

INTRODUZIONE

Sfruttare le grandi basi di conoscenza, in particolar modo la loro rappresentazione come knowledge graph [1], permette di aumentare il dominio di applicazioni e la complessità delle stesse, nello specifico parliamo di basi di conoscenza presenti su infrastrutture web. La classificazione in questo ambito mira a mappare entità e relazioni, e quindi ridurre il problema dell'effettuare predizioni a problemi di algebra lineare [2]. Questo assume non poche difficoltà, tra cui estrarre la conoscenza attraverso solo il ragionamento deduttivo o la difficile interpretabilità dei modelli stessi.

Analizzando un'idea nel contesto dell'apprendimento neurale [3], [5], è stato visto come è possibile adattare modelli grafici e quindi creare regole probabilistiche partendo proprio dalla rappresentazione delle istanze nei KG per produrre assiomi.

I modelli probabilistici costruiti e descritti di seguito hanno l'obiettivo di essere più interpretabili e verificabili dagli esperti di dominio.

Quindi i punti cardine su cui si basano sono:

- l'assunzione di indipendenza, tipica dei modelli Naive Bayes [4];
- le prestazioni anche in presenza di conoscenza incerta, quindi dati mancanti, correlata a quella di OWA (Open World Assumption) [6].

In particolar modo, il lavoro svolto riguarda essenzialmente l'ultimo punto, in quanto il reasoner durante la procedura di dimostrazione di appartenenza di un individuo ad una determinata classe, in presenza di dati incerti, non riesce a dare una stima.

Sommario

Il sistema ragiona ed impara attraverso dati presenti all'interno di ontologie web, quest'ultime strutturate come knowledge graph e rappresentate attraverso l'uso di OWL-DL, in cui sono presenti anche dati mancanti o incompleti, cioè senza che sia esplicitata l'appartenenza ad una determinata classe o che abbia alcune feature con valori incerti.

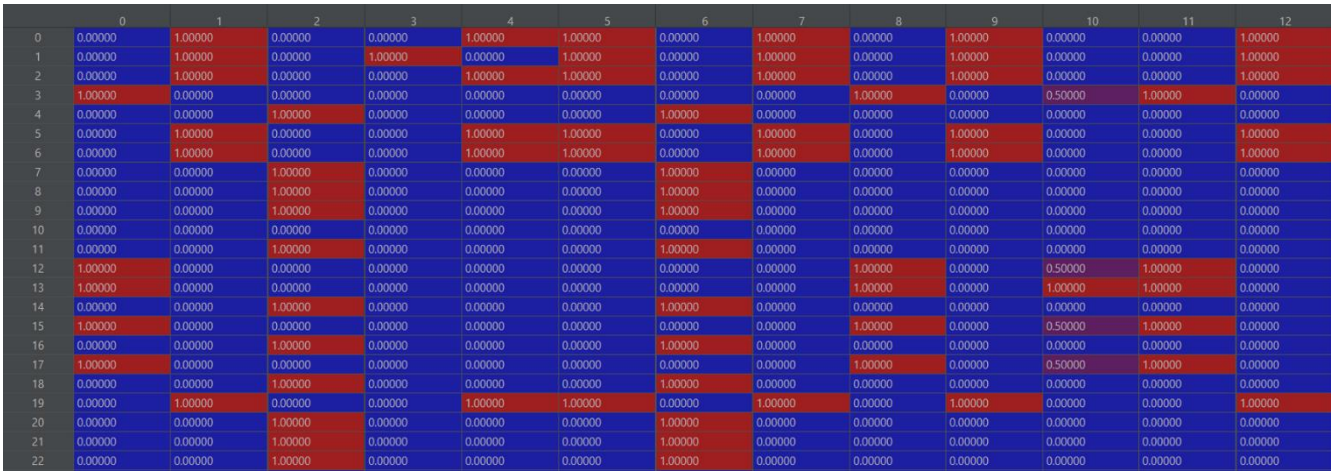
L'obiettivo del sistema è quello di fornire una classificazione in fase di test. Al suo interno sono quindi integrate parti in grado di estrapolare i dati dalle ontologie, analizzarli ed eseguire feature selection, quindi imparare da essi in presenza di incertezza, il tutto anche mediante l'utilizzo di cross validation. Nello specifico il sistema testa due diversi tipi di modelli generativi, di cui uno già implementato (MBM), mettendoli a confronto sulla base di diverse metriche adottate (precision, recall e F1 measure). Gli esperimenti vengono condotti su tre diverse ontologie denominante "LEHIGH UNIVERSITY BENCHMARK" (LUBM), "FINANCIAL" e "NTNAMES" affrontando per ognuna dieci problemi di classificazione.

Il dataset

I dati sono rappresentati da valori discreti, precisamente booleani [1, 0], che determinano rispettivamente l'appartenenza o la non appartenenza alla classe o la presenza di una feature per un individuo (FIG. 1, 2). I valori mancanti o incompleti verranno descritti rispettivamente con -1 e 0.5, valori che determinano l'incertezza.

FIG. 1 ***(estratto da FINANCIAL)***

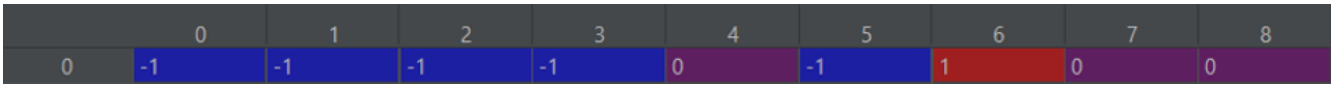
Matrice composta da individui sulle righe e le corrispondenti feature sulle colonne dopo la feature selection



	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
1	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
2	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
3	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
4	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
6	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
7	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
9	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
10	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
11	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
12	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
13	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
14	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
15	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
16	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
17	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
18	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
19	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
20	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
21	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
22	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

FIG. 2 ***(estratto da FINANCIAL)***

Vettore contenente l'appartenenza alla classe per ogni individuo



	0	1	2	3	4	5	6	7	8
0	-1	-1	-1	-1	0	-1	1	0	0

Multivariate Bernoulli Model

Data la rappresentazione degli individui come tuple di dimensione D e valori binari (FIG. 1), e indicando con y l'appartenenza ad una classe (FIG. 2), la distribuzione di ogni individuo può essere descritta attraverso un modello Bayesiano [11], precisamente come Multivariate Bernoulli Model e la probabilità condizionata è così ottenuta:

$$P(\mathbf{x}|y) = \mathrm{Ber}_D(\mathbf{x}|\mathbf{p}_y) = \prod_{i=1}^{D} \mathrm{Ber}(x_i|p_{yi})$$

$$= \prod_{i=1}^{D} (p_{yi})^{x_i} (1 - p_{yi})^{1-x_i}$$

Laddove **x** è considerato un vettore così rappresentato:

$$\mathbf{x} = (x_1, x_2, \ldots, x_D)^T$$

Il problema della classificazione riguarda stimare l'output di y in base ai dati di input rappresentati da tutti i vettori **x**, e quindi dati i parametri di probabilità a priori P(y) e quella condizionata P(**x**|y), la distribuzione viene calcolata usando Bayes:

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \frac{\pi^y \text{Ber}_D(\mathbf{x}; \mathbf{p}_y)}{P(\mathbf{x})}$$

Infine, l'appartenenza dell'individuo viene predetta massimizzando P(y|**x**):

$$\hat{y}(\mathbf{x}) = \underset{y \in \mathbb{B}}{\text{argmax}}\, P(y|\mathbf{x})$$

Il modello già implementato lavora essenzialmente sui dati che sono noti (ovvero dove è noto y), calcolando i parametri, ovvero la probabilità a priori delle classi e della distribuzione delle feature per classe, attraverso il metodo della massima verosimiglianza.

Il MLE per la probabilità a priori viene stimato come segue:

$$\hat{\pi}^y = N_y/N.$$

E per le feature invece (solo se binarie):

$$\hat{p}_{yi} = N_{yi}/N \qquad y = 0, 1;\ i = 1, \ldots, D$$

Elenco argomenti di interesse tratti dal programma

1. Apprendimento supervisionato
   1.1. Valutare le predizioni
   1.2. Sovradattamento
      1.2.1. Cross Validation
2. Ragionare in presenza di incertezza:
   2.1. Probabilità
      2.1.1. Probabilità condizionata
      2.1.2. Teorema di Bayes
   2.2. Indipendenza
3. Apprendimento con incertezza:
   3.1. Apprendimento probabilistico
      3.1.1. Apprendere le probabilità
      3.1.2. Probabilità da esperti
      3.1.3. Classificatori probabilistici
   3.2. Apprendimento non supervisionato
      3.2.1. Expectation Maximization per il soft clustering
4. Ontologie e sistemi basati su ontologie
   4.1. Rappresentazione grafica
   4.2. Ontologie e condivisione della conoscenza
   4.3. Logica descrittiva

IMPLEMENTAZIONE ALGORITMO EXPECTATION-MAXIMIZATION

Sommario

Il modello MBM-EM estende il modello già implementato (MBM) ed ha il compito di apprendere e predire le istanze derivanti dalle ontologie in presenza di dati incompleti o mancanti. Il punto principale di questa classe è l'utilizzo dell'algoritmo EM [7] che in questo caso specifico è utilizzato per effettuare apprendimento semi-supervisionato [8], in quanto non tutti i dati sono da considerarsi incerti e motivo per cui vengono effettuate scelte diverse rispetto al non supervisionato [9]. La struttura principale dell'algoritmo rimane invariata, quindi successivamente all'inizializzazione, succederanno ricorsivamente i due passi di ricalcolo delle responsabilità e massimizzazione dei parametri [10]. In seguito alla fase di fit verranno effettuate le predizioni attraverso l'utilizzo dei parametri massimizzati. I risultati verranno poi confrontati con quelli ottenuti dal modello già implementato (MBM).

Nello specifico, adattando l'algoritmo al caso di studio, il passo E (Expectation) calcola le responsabilità attraverso:

$$\hat{P}(y|\mathbf{x}^t) = \frac{\hat{\pi}^y \mathrm{Ber}_D(\mathbf{x}^t; \hat{\mathbf{p}}_y)}{\hat{\pi}\mathrm{Ber}_D(\mathbf{x}^t; \hat{\mathbf{p}}_1) + (1-\hat{\pi})\mathrm{Ber}_D(\mathbf{x}^t; \hat{\mathbf{p}}_0)}$$

E nel passo M (Maximization) si calcolano i parametri in questo modo:

$$\hat{\pi}^y = \hat{P}(y) = \frac{1}{N}\sum_{t=1}^{N} \hat{P}(y|\mathbf{x}^t)$$

$$\hat{p}_{yi} = \frac{\sum_t \hat{P}(y|\mathbf{x}^t) x_i^t}{\sum_t \hat{P}(y|\mathbf{x}^t)}$$

Strumenti utilizzati

Ambiente di sviluppo:

- Intellij IDEA: IDE nel quale è stata implementato il progetto, con linguaggio di programmazione Python.

Librerie:

- OwlReady2: libreria usata per gestire ontologie all'interno di Python (https://pypi.org/project/Owlready2/);
- Scikit-learn: libreria utilizzata per il pre-processing e apprendimento supervisionato (https://scikit-learn.org/stable/);

- Scipy: libreria utilizzata per il calcolo del criterio di stop all'interno dell'algoritmo EM.

Decisioni di Progetto

L'algoritmo calcola per ogni individuo la distribuzione di appartenenza ad una determinata classe, in particolar modo di quelli la cui etichetta (y) non è nota. Non vengono calcolati i dati incerti (feature) per ogni **x**, che quindi saranno ininfluenti in fase di apprendimento.

Durante la costruzione del modello sono state prese diverse decisioni in merito all'inizializzazione dei parametri e alla trattazione dei dati incompleti o mancanti rispetto a quelli noti.

Inizializzazione:

per una scelta di progetto si è deciso di sfruttare i dati che sono già noti per calcolare inizialmente i parametri, ovvero la probabilità a priori e quella condizionata attraverso il calcolo delle responsabilità. Nello specifico, dato che siamo in presenza di variabili bernoulliane, ci si è concentrati sul determinare l'appartenenza (C=1) o la non appartenenza (C=0) ad una determinata classe in base ai diversi problemi target affrontati. Ragion per cui nel caso del calcolo delle responsabilità degli individui, vengono utilizzate già le etichette disponibili (quindi non calcolate), così da produrre una matrice contenente tutte le distribuzioni note di appartenenza individui-classe (FIG. 3).

Successivamente vengono utilizzate queste per la massimizzazione dei parametri di cui sopra, così da assicurare risultati migliori rispetto alla totale incertezza.

FIG. 3 ***(estratto da FINANCIAL)***

Matrice delle responsabilità di ogni individuo la cui classe è nota in fase di inizializzazione

	0	1
0	1	0
1	0	1
2	1	0
3	1	0
4	1	0
5	1	0
6	1	0
7	1	0
8	1	0
9	1	0
10	1	0
11	1	0
12	1	0
13	1	0
14	1	0
15	1	0
16	1	0
17	1	0
18	1	0
19	1	0
20	0	1
21	1	0
22	1	0

Iterazioni:

il modello presenta i concetti base dell'algoritmo EM, utilizzando nel ciclo di apprendimento sia i dati noti che non (o incompleti). In particolare, nel passo E, nel caso dei dati le cui appartenenze sono note viene imposto il valore "1" di appartenenza alla classe, per tutti gli altri sono calcolate le responsabilità utilizzando i parametri calcolati al passo M. Notiamo alcuni valori indicare precisamente "1" che risultano essere le appartenenze già note imposte in fase di costruzione della matrice (FIG. 4).

Come criterio di stop viene utilizzata la distanza euclidea ottenuta tramite la differenza delle sommatorie sulla matrice delle responsabilità, oltre che un numero massimo di iterazioni possibili nel caso non ci sia convergenza.

FIG. 4 ***(estratto da FINANCIAL)***

Matrice delle responsabilità di ogni individuo per ogni classe, nota e no, in fase di training

	0	1
0	0.83117	0.16883
1	0.83303	0.16697
2	0.83117	0.16883
3	0.93864	0.06136
4	1.00000	0.00000
5	0.83117	0.16883
6	0.00000	1.00000
7	1.00000	0.00000
8	1.00000	0.00000
9	1.00000	0.00000
10	0.92222	0.07778
11	1.00000	0.00000
12	0.93864	0.06136
13	0.93980	0.06020
14	1.00000	0.00000
15	0.93864	0.06136
16	1.00000	0.00000
17	0.93864	0.06136
18	1.00000	0.00000
19	0.83117	0.16883
20	1.00000	0.00000
21	1.00000	0.00000
22	1.00000	0.00000

VALUTAZIONE E DISCUSSIONE RISULTATI

Come detto in precedenza, gli esperimenti comprendono tre diversi tipi di ontologie denominate FINANCIAL, NEW TESTAMENT NAMES (NTNAMES) e LEHIGH UNIVERSITY BENCHMARK (LUBM).

I risultati dei due modelli confrontati sono misurati in base alle metriche di Precision, Recall ed F1. Tra i documenti allegati vengono forniti i risultati di ogni modello per ogni problema di classificazione affrontato.

Di seguito vengono mostrate solo le medie finali:

RISULTATI: (MEDIE ± DEVIAZIONI STANDARD)

MODELS			
KNWOLEDGE GRAPHS	MISURE	MBM	MBM_EM
FINANCIAL	*Precision*	0.995 ± 0.012	0.962 ± 0.049
	Recall	0.991 ± 0.026	0.943 ± 0.071
	F_1	0.991 ± 0.022	0.944 ± 0.070
LUBM	*Precision*	1.000 ± 1.000	1.000 ± 1.000
	Recall	1.000 ± 1.000	1.000 ± 1.000
	F_1	1.000 ± 1.000	1.000 ± 1.000
NTNAMES	*Precision*	0.992 ± 0.013	0.956 ± 0.060
	Recall	0.991 ± 0.015	0.940 ± 0.086
	F_1	0.991 ± 0.015	0.937 ± 0.089

Come si evince dai risultati il modello MBM ha delle prestazioni leggermente migliori rispetto alla sua estensione che implementa l'algoritmo EM, in particolar modo per FINANCIAL e NTNAMES.

In primis notiamo che le migliori prestazioni per modello si hanno per l'ontologia "LUBM", a seguire "FINANCIAL" e "NTNAMES", questo è probabilmente correlato al numero di individui che è maggiore in "LUBM" poi in "FINANCIAL" e "NTNAMES".

Andando più nel dettaglio ed analizzando i dati specifici per ogni problema possiamo dedurre alcune cose:

partendo dal presupposto che le predizioni dei modelli vengono effettuate solo sui dati noti (quindi eliminando quelli non noti dal test set) e correlandolo alla fase di training del modello MBM che viene effettuata anch'essa solo sui dati noti, le prestazioni del modello in questione possono essere motivate dalla poca varietà dei dati presenti, che quindi caratterizzano training e test set (FIG. 5).

FIG. 5 ***(estratto da FINANCIAL)***

Dati noti

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
1	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
2	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
7	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
9	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
10	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
11	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
12	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
13	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
14	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
15	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
16	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
17	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
18	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
19	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
20	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
21	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
22	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

A differenza, il modello MBM-EM, utilizza in fase di training tutti i dati disponibili, anche se non noti, stimandone l'appartenenza attraverso il calcolo delle probabilità. Questi ultimi si presentano con una varietà molto diversa (FIG. 6) rispetto quelli noti (FIG. 5). L'apprendimento di questi dati potrebbe in qualche senso creare "rumore" fuorviando il modello in fase di test. Infatti, se guardiamo i risultati per LUBM nei problemi target che non presentano dati mancanti (problema 5 e 8) abbiamo risultati identici per entrambi i modelli.

FIG. 6 ***(estratto da FINANCIAL)***

Dati non noti

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
1	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
2	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
3	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
4	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
5	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
7	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
8	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
9	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
10	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
11	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
12	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
13	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
14	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
15	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
16	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.50000	1.00000	0.00000
17	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.50000	0.00000	0.00000	1.00000
18	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
19	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	1.00000	1.00000	0.00000
20	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
21	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000
22	0.00000	1.00000	0.00000	0.00000	1.00000	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000	0.00000	1.00000

Come si può notare, oltre la varietà di distribuzione delle feature note per ogni individuo, ci sono molti più valori incerti per le stesse che, come detto in precedenza, non apportano alcun vantaggio o svantaggio nel calcolo dei parametri.

Un'idea per valutare meglio i risultati di entrambi i modelli sarebbe avere in fase di test dei dati noti che si discostino da quelli già a disposizione, magari avvicinandosi per varietà a quelli in FIG. 6, così da valutare meglio l'apprendimento dei dati incerti tramite EM.

Un altro motivo per cui il modello MBM-EM performa diversamente da MBM potrebbe essere legato al principale difetto che colpisce l'algoritmo EM, ovvero la sua convergenza ai massimi locali, anche se non sempre assicurata [9].

CONCLUSIONI

Uno dei possibili sviluppi che potrebbero portare miglioramenti al modello MBM-EM riguardano principalmente l'implementazione di una logica che calcoli i valori delle feature incerte per tutte le **x**.

In aggiunta può essere cercata anche una soluzione che eviti all'algoritmo di convergere a massimi locali, come per esempio l'implementazione di un algoritmo "random restart".

Infine, può essere implementato un modello su due livelli, come un Mixture Multivariate Bernoulli Model [5], [11], al fine di avere al primo livello un modello che implementi l'algoritmo EM per le componenti nascoste ed al livello superiore un classificatore Naive Bayes.

Riferimenti Bibliografici

[1]. Hogan, E. Blomqvist, M. Cochez, C. D'amato, G. De melo, C. Gutierrez, J. Emilio labra gayo, s. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A. Cyrille Ngonga Ngomo, S. M. Rashid, A. Rula, l. Schmelzeisen, J. Sequeda, S. Staab, A. Zimmermann, "**Knowledge Graphs**" 2020, arXiv:2003.02320

[2]. Q. Wang, Z. Mao, B. Wang and L. Guo, "**Knowledge graph embedding: A survey of approaches and applications**", IEEE *Transactions on Knowledge and Data Engineering*, vol 29, no. 12, pp. 2724-2743, 2017.

[3]. V. Tresp, J. Hollatz, and S. Ahmad, "**Representing probabilistic rules with networks of Gaussian basis functions**", *Machine Learning*, vol. 27, pp. 173–200, 1997.

[4]. P. M. Domingos and M. J. Pazzani, "**On the optimality of the simple Bayesian classifier under zero-one loss**", *Mach. Learn.*, vol. 29, no. 2-3, pp. 103–130, 1997.

[5]. M. Jordan and R. Jacobs, "**Hierarchical mixtures of experts and the EM algorithm**", in *Proceedings of IJCNN*, vol. 2, 1993, pp. 1339–1344.

[6]. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel- Schneider, Eds., "**The Description Logic Handbook: Theory, Implementation and Applications***"*, 2nd ed. Cambridge University Press, 2007.

[7]. [Bishop, 06] C. M. Bishop. "**Pattern Recognition and Machine Learning**". Springer, 2006. (§9.3.3)

[8]. [Ghahramani et al., 93] Z. Ghahramani and M. Jordan, "**Supervised learning from incomplete data via an EM approach**" in Advances in Neural Information Processing Systems, vol. 6. Morgan-Kaufmann, 1993.

[9]. D. L. Poole, A. K. Mackworth "**Artificial Intelligence: Foundations of Computational Agents**", 2018. (§10.2.2)

[10]. [Tresp et al., 97] V. Tresp, J. Hollatz, and S. Ahmad, "**Representing probabilistic rules with networks of Gaussian basis functions**," *Machine Learning*, vol. 27, pp. 173–200, 1997

[11]. N. Fanizzi, C. d'Amato, "**Towards interpretable probabilistic classification models for knowledge graphs**", 2022.