Sistema di diagnostica sull'anemia

Antonio Castriotta, Matricola: 745139

Antonio Capicotto, Matricola: 719315

Anno Accademico 2022-2023

Email: a.castriotta5@studenti.uniba.it, a.capicotto@studenti.uniba.it

Indice:

1 Introduzione

L'**anemia**: è una condizione caratterizzata dalla riduzione patologica dei livelli di emoglobina (Hb) al di sotto dei livelli di normalità (Hb < 13.5 g/L nel maschio e 12 g/L nella femmina) , che determina una ridotta capacità del sangue di trasportare ossigeno nella giusta quantità necessaria a soddisfare le esigenze di tutti i tessuti. A causa di questa carenza, i tessuti e gli organi del corpo non ricevono l'ossigeno di cui hanno bisogno per svolgere normalmente le loro funzioni.

Il programma descritto, per diagnosticare l'anemia si avvale di due moduli:

- L'utilizzo di algoritmi di apprendimento supervisionato per la diagnostica dell'anemia.
- Un agente intelligente che tiene conto delle risposte dell'utente per la diagnostica della malattia.

2 Knowledge base

Per la diagnostica dell'anemia abbiamo deciso di utilizzare un sistema esperto, per fare ciò abbiamo utilizzato la libreria python "Experta".

Un sistema esperto è un programma in grado di associare un insieme di fatti ad un insieme di regole, ed eseguire alcune azioni in base alle regole di corrispondenza.

2.1 Funzionamento Experta

"Experta" si basa su **fatti** e **regole**.

Per **fatti** intendiamo l'oggetto sul quale verranno verificate le regole.

Una **regola** è un callable formato da due componenti:

- LHS (sinistro): descrive le condizioni base alle quali la regola dovrebbe essere eseguita.
- RHS (destro): è l'insieme delle azioni da eseguire quando la regola viene eseguita.

Ogni regola affinchè possa essere utilizzata deve essere un metodo di una sottoclasse KnowledgeEngine.

2.2 Gestione dei fatti

Un **fatto** può essere:

- Dichiarato: Aggiunta di un nuovo fatto all'elenco dei fatti.
- Ritrattato: Rimozione di un fatto esistente dall'elenco dei fatti.
- Modificato: Ritiro di un fatto e aggiunta del medesimo fatto modificato.
- Duplicato: Aggiunta di un nuovo fatto all'elenco utilizzando un fatto esistente come modello inserendo alcune modifiche.

2.3 Funzionamento sistema esperto

Durante l'esecuzione del sistema esperto l'utente è sottoposto ad una serie di domande attraverso il quale il sistema dichiara dei facts, cioè poposizioni sulle quali verrà effettuato considerando le Rule utili per la diagnosi dell'anemia.

2.4 Dataset

Per poter realizzare al meglio il sistema abbiamo preso in esame il seguente dataset:

https://www.kaggle.com/datasets/biswaranjanrao/anemia-dataset

Il dataset contiene diverse colonne:

- **Gender**: in questa colonna sono riportati i dati inerenti al sesso dei pazienti (0-maschio, 1-femmina).
- **Emoglobina**: in questa colonna sono riportati per ogni paziente il relativo valore di emoglobina (per i pazienti di sesso maschile il valore minimo di emoglobina non deve essere inferiore a 13,5, per i pazienti di sesso femminile non deve essere inferiore a 12).
- **MCH**: emoglobina media delle cellule, i valori normali sono compresi tra 26 e 32 picogrammi.
- **MCHC**: sono i cosiddetti *indici corpuscolari* e misurano rispettivamente il volume, il contenuto e la concentrazione media di emoglobina nei globuli rossi nel sangue, i valori di riferimento sono 32-36%.
- **MCV**: volume medio delle cellule, i valori normali sono compresi tra 80 e 100 fl.
- **Results**: in questa colonna sono riportati i risultati sulla positività o negatività dei pazienti in esame (0-non anemico, 1-anemico).

2.5 Ontologia

In informatica, un'ontologia è una rappresentazione formale, condivisa ed esplicita di una concettualizzazione di un dominio di interesse. Il termine ontologia formale è entrato in uso nel campo dell'intelligenza artificiale e della rappresentazione della conoscenza, per descrivere il modo in cui diversi schemi vengono combinati in una struttura dati contenente tutte le entità rilevanti e le loro relazioni in un dominio.

Le ontologie distribuite sulla rete vengono descritte attraverso il linguaggio OWL (che sta per Web Ontology Language); esso permette di descrivere il mondo in termini di:

- **Individui**: entità del mondo
- **Classi**: insieme di individui, accomunati da delle caratteristiche comuni
- **Proprietà**: relazioni che associano agli elementi del proprio dominio (individui), un elemento del codominio (valori che può assumere).

In particolare, si dividono in;

- **Datatype property**: se il codominio contiene solo tipi primitivi (interi, stringhe, …)
- **Object property**: quando il codominio contiene altre classi

La strutturazione dell'ontologia è avvenuta tramite il software applicativo **protège** che ne permette la realizzazione in maniera intuitiva e completa, con l'aggiunta di plug-in molto utili al fine progettuale.

Con **protège** è stato possibile creare l'ontologia sull'anemia creando le varie entità corrispondenti a ciascun sintomo con le rispettive proprietà.

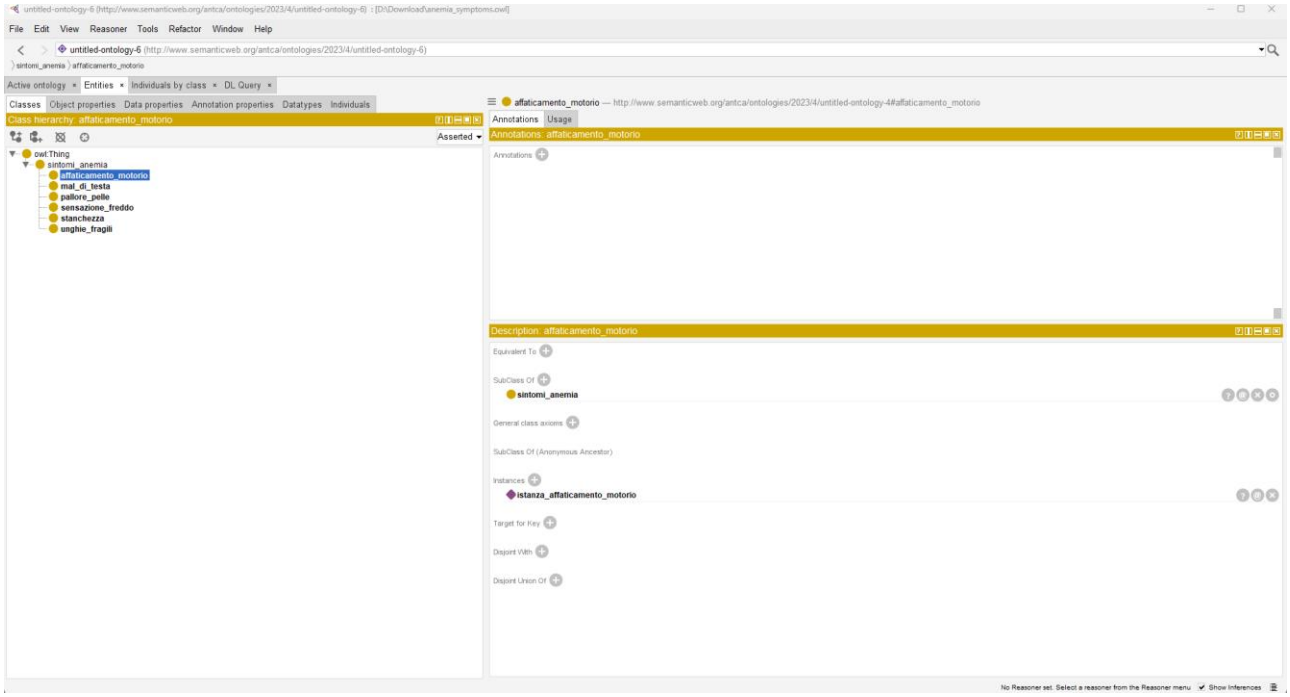


Figura 2.5.1: tool protege

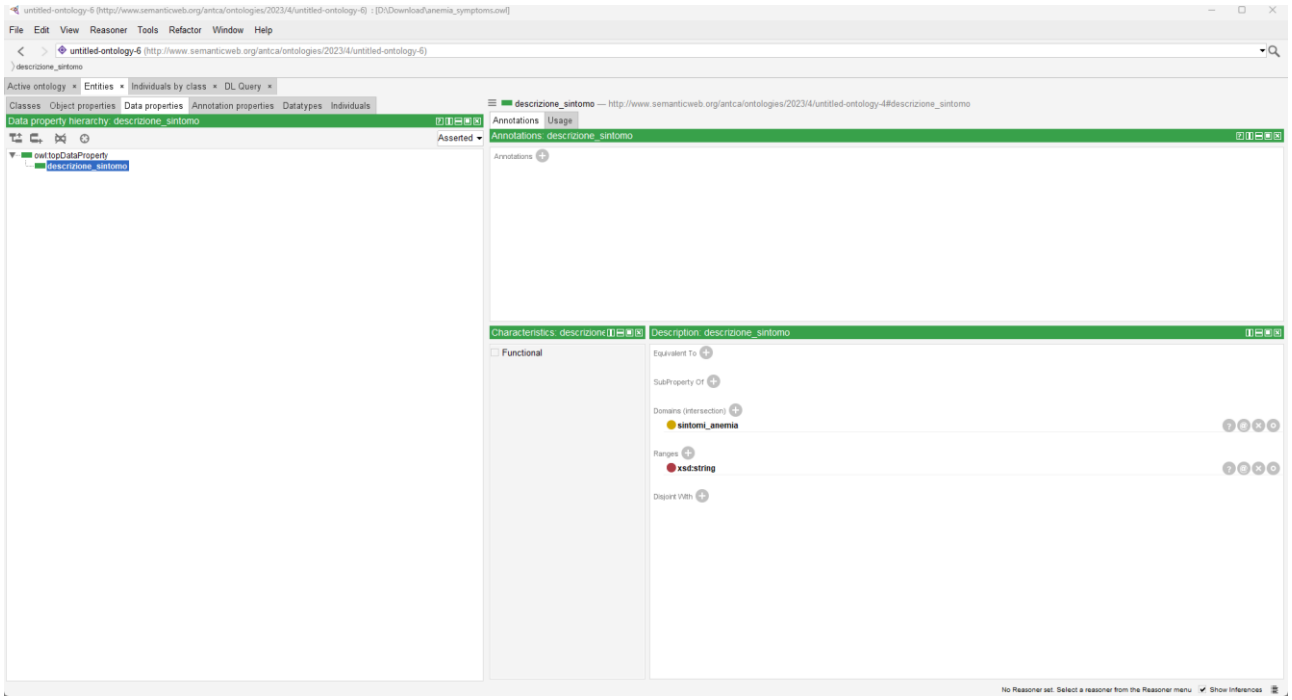


Figura 2.5.2: sezione “data properties” protege

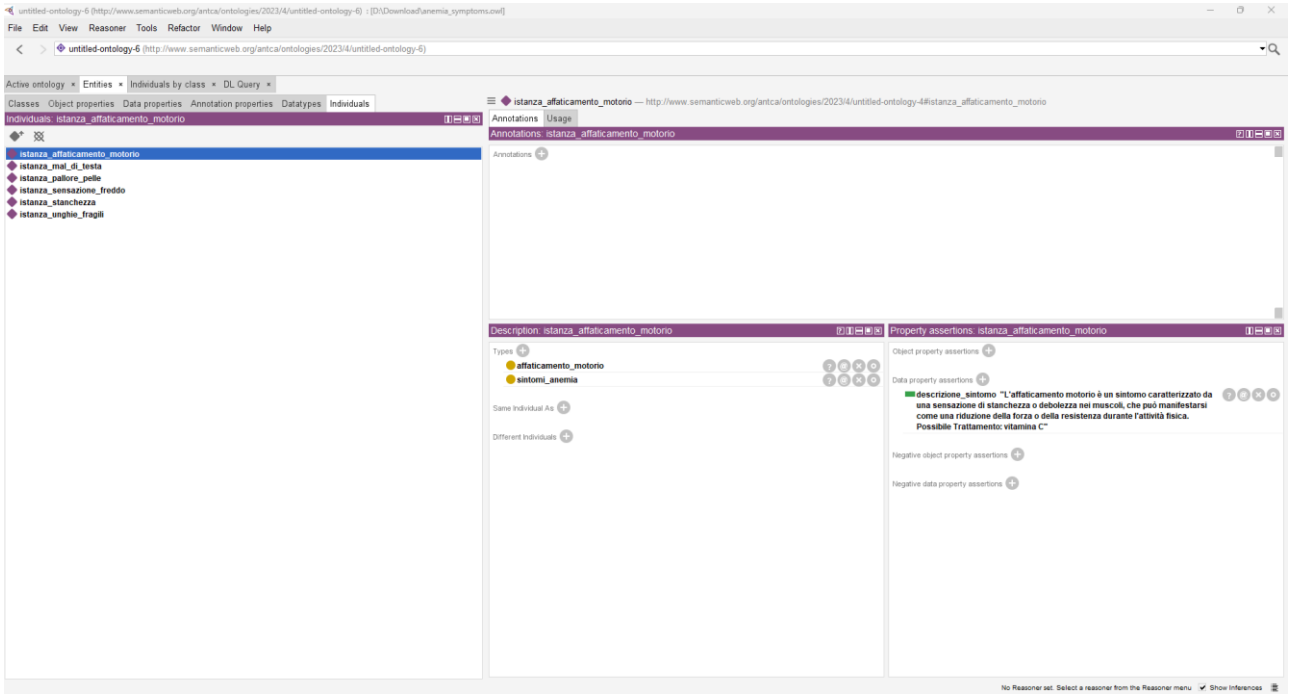


Figura 2.5.3: sezione “individuals” protege

Successivamente abbiamo creato “anemia_symptoms.owl” in modo da gestire l’ontologia tramite la libreria **Owlready2** presente in python.

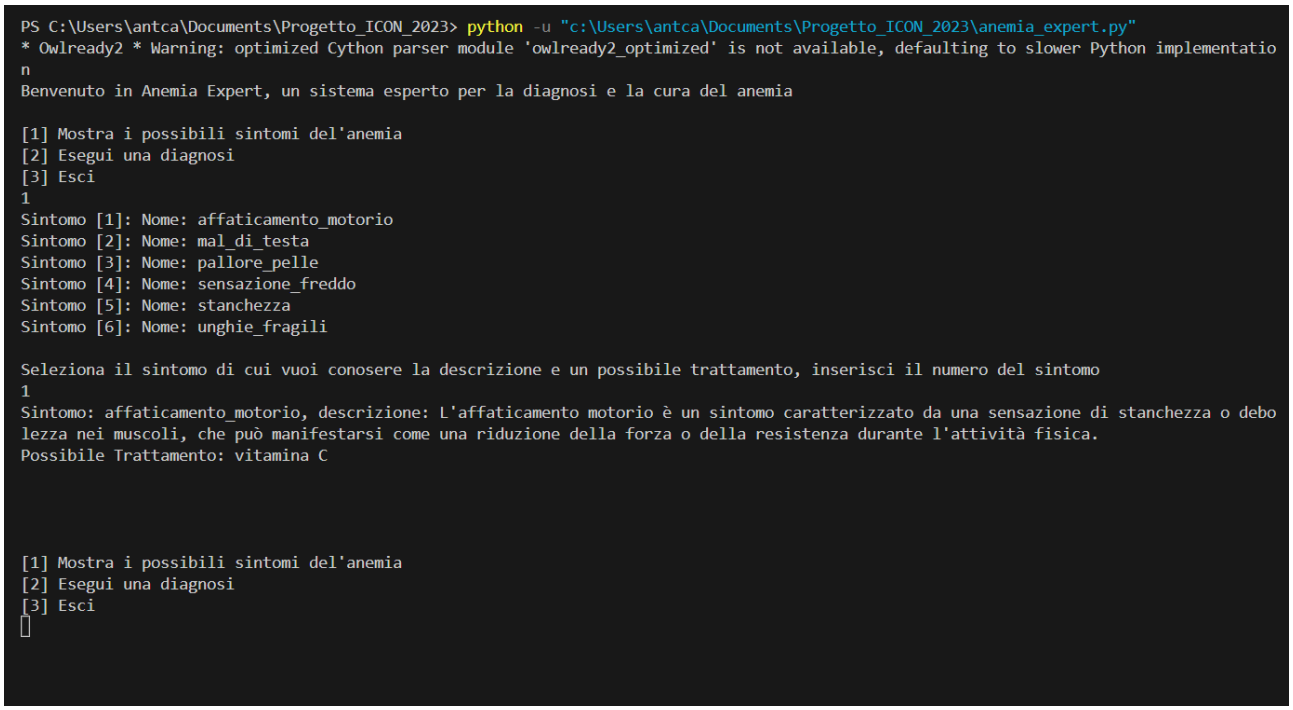


Figura 2.5.4: Lista dei sintomi con la relativa descrizione e un possibile trattamento

2.6 CSP (Constraint Satisfaction Problem)

Molti problemi nell’ambito dell’Intelligenza Artificiale sono classificabili come Problemi di Soddisfacimento di Vincoli (Constraint Satisfaction Problem o CSP); Formalmente, un CSP può essere definito su un insieme finito di variabili (X1,X2, . . . ,Xn) i cui valori appartengono a domini finiti di definizione(D1,D2, . . . ,Dn) e su un insieme di vincoli(C1,C2, . . . ,Cn). Un vincolo su un insieme di variabili è una restrizione dei valori che le variabili possono assumere simultaneamente. Concettualmente, un vincolo può essere visto come un insieme che contiene tutti i valori che le variabili possono assumere contemporaneamente: un vincolo tra k variabili C(Xi1 ,Xi2 , . . . ,Xik), è un sottoinsieme del prodotto cartesiano dei domini delle variabili coinvolte Di1 ,Di2 , . . . ,Dik che specifica quali valori delle variabili sono compatibili con le altre. Questo insieme può essere rappresentato in molti modi, come ad esempio, matrici, equazioni, disuguaglianze o relazioni.

La libreria utilizzata, **constraint**, ci ha permesso di realizzare un CSP in grado di mostrare la disponibilità di un medico convenzionato per effettuare delle visite e per poter prenotare i farmaci utili per i trattamenti dei sintomi in una farmacia convenzionata.

Il funzionamento del CSP è il seguente:

- Alla base vi è una sottoclasse di Problem (classe già definita in constraint, che modella un CSP).
- Vengono aggiunte variabili con proprio dominio associato in maniera esplicita.

- In base alle risposte fornite dall'utente, il sistema decide se mostrare la possibilità di prescrivere una visista dal medico oppure di prescrivere un farmaco in farmacia.
- In caso di risposta affermativa da parte dell'utente enta in gioco il CSP.
- Ad esempio: il sistema indica l'ora e il giorno in cui è possibile effettuare una prenotazione per una visita dal medico oppure in caso di prenotazione di un farmaco in farmacia mostra il nome del farmaco e la quantità prenotabile.

Un possibile esempio:

```
Hai eseguito un test dell'emoglobina?
no
Vuoi prenotare un appuntamento presso un medico convenzionato? [si/no]
si
Disponibilita' del medico

Turno [0], Giorno: lunedi, Orario: 8
Turno [1], Giorno: lunedi, Orario: 9
Turno [2], Giorno: lunedi, Orario: 10
Turno [3], Giorno: lunedi, Orario: 11
Turno [4], Giorno: lunedi, Orario: 12
Turno [5], Giorno: lunedi, Orario: 13
Turno [6], Giorno: lunedi, Orario: 14
Turno [7], Giorno: giovedi, Orario: 15
Turno [8], Giorno: giovedi, Orario: 16
Turno [9], Giorno: giovedi, Orario: 17
Turno [10], Giorno: giovedi, Orario: 18
Turno [11], Giorno: giovedi, Orario: 19
Turno [12], Giorno: giovedi, Orario: 20


Insersci un turno inserendo il numero del turno associato
7
Turno selezionato: [7], Giorno: giovedi, Orario: 15
```

Figura 2.6.1: Esempio di prenotazione di una visita da un medico convenzionato

```
Vuoi prenotare farmaci presso una farmacia convenzionata? [si/no]
si
Farmaci disponibili:

Farmaci disponibili:

1. Farmaco: vitamina_c, Quantità: 2
2. Farmaco: integratore_biotina, Quantità: 2
3. Farmaco: paracetamolo, Quantità: 2
4. Farmaco: integratore_ferro, Quantità: 2
5. Farmaco: vitamina_b12, Quantità: 2
6. Farmaco: vitamina_c, Quantità: 1
7. Farmaco: integratore_biotina, Quantità: 1
8. Farmaco: paracetamolo, Quantità: 1
9. Farmaco: integratore_ferro, Quantità: 1
10. Farmaco: vitamina_b12, Quantità: 1
Inserisci il numero del farmaco da prenotare
3
Hai prenotato il farmaco: paracetamolo, Quantità: 2
```

Figura 2.6.2: Esempio di prenotazione di un farmaco per la cura dei sintomi da una farmacia

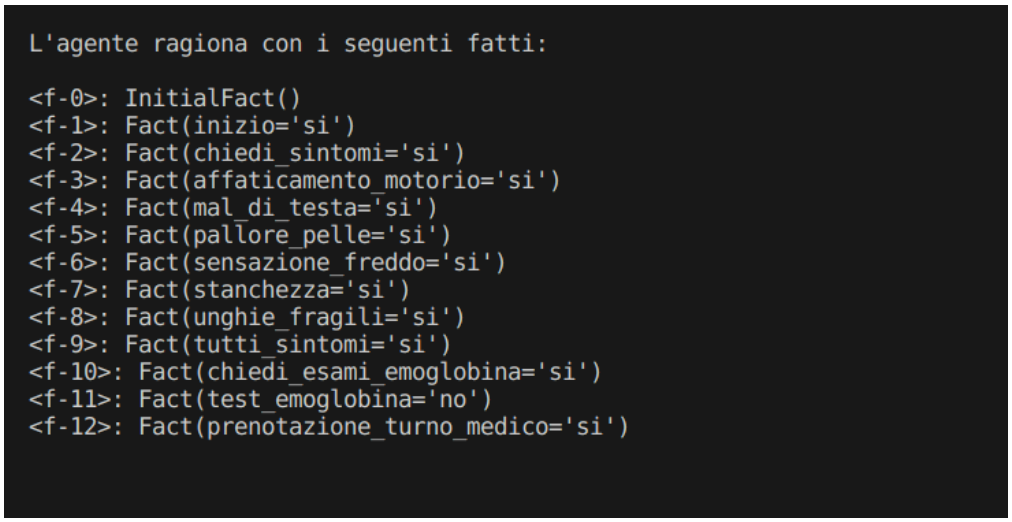


Figura 2.6.3: In base alle risposte fornite dall'utente, l'agente ragiona con i seguenti fatti

3 Apprendimento Supervisionato

In questa sezione di progetto sono stati implementati diversi algoritmi di Apprendimento Supervisionato appartenenti, in modo specifico, alla categoria degli Algoritmi di Classificazione.

Gli algoritmi utilizzati sono:

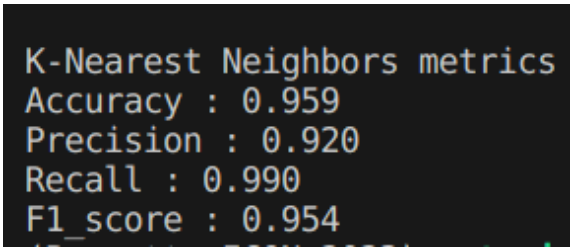
- **K-Nearest Neighbours;**
- **Decision Tree Classifier;**
- **Logistic Regression**

3.1 K-nearest neighbor

Il K-nearest neighbor, abbreviato K-NN, è un algoritmo utilizzato nel riconoscimento di pattern per la classificazione di oggetti basandosi sulle caratteristiche degli oggetti vicini a quello considerato.

La libreria da noi usata per l'implementazione di questi algoritmi è **sklearn**, che ci ha permesso di allenare e testare i modelli di apprendimento supervisionato.

Qui di seguito vengono stampati a video i risultati delle metriche calcolate:



Accuracy: Misura la percentuale delle previsioni esatte sul totale delle istanze. Varia da 0 (Peggiore) ad 1 (Migliore):

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} = 1 - ERR$$

Precision: Percentuale delle previsioni positive corrette (TP) sul totale delle previsioni positive del modello (TP+FP).

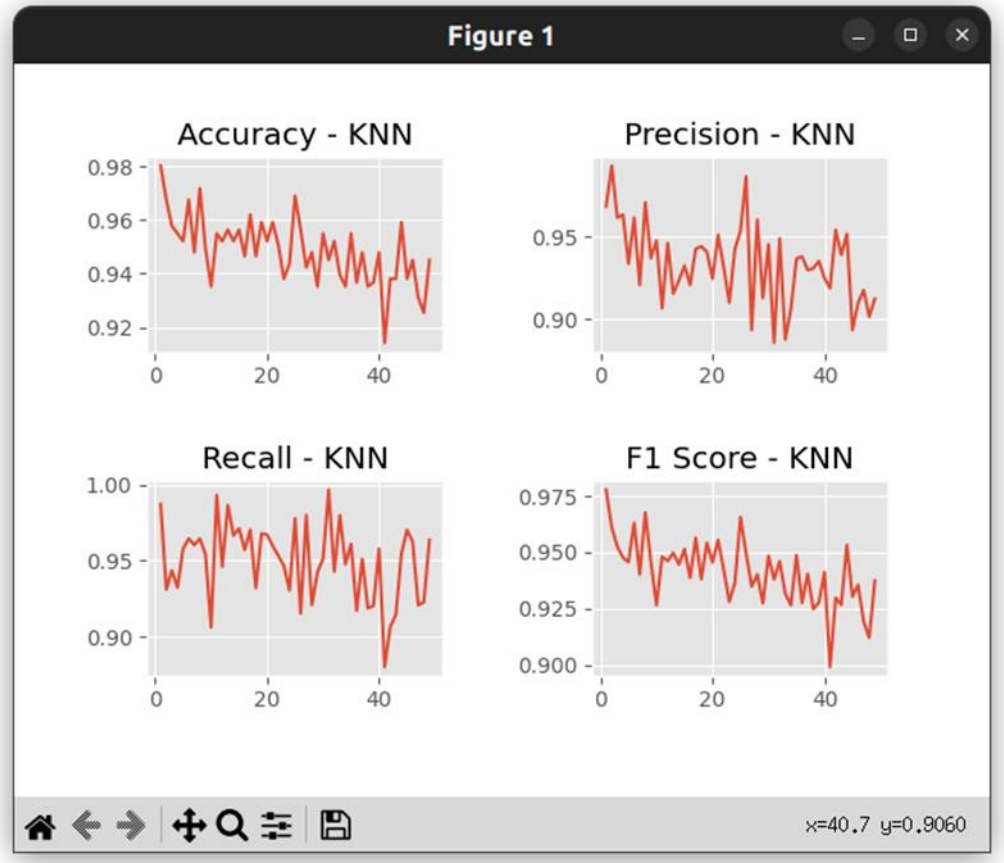
$$PR = \frac{TP}{TP + FP}$$

Recall: Percentuale delle previsioni positive corrette (TP) sul totale delle istanze positive (TP+FN), varia da 0 (Peggiore) ad 1 (Migliore).

$$Recall = \frac{TP}{TP + FN}$$

F1: Media armonica delle metriche Precision e Recall. Varia da 0 (Peggiore) ad 1 (Migliore).

$$FS = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$$

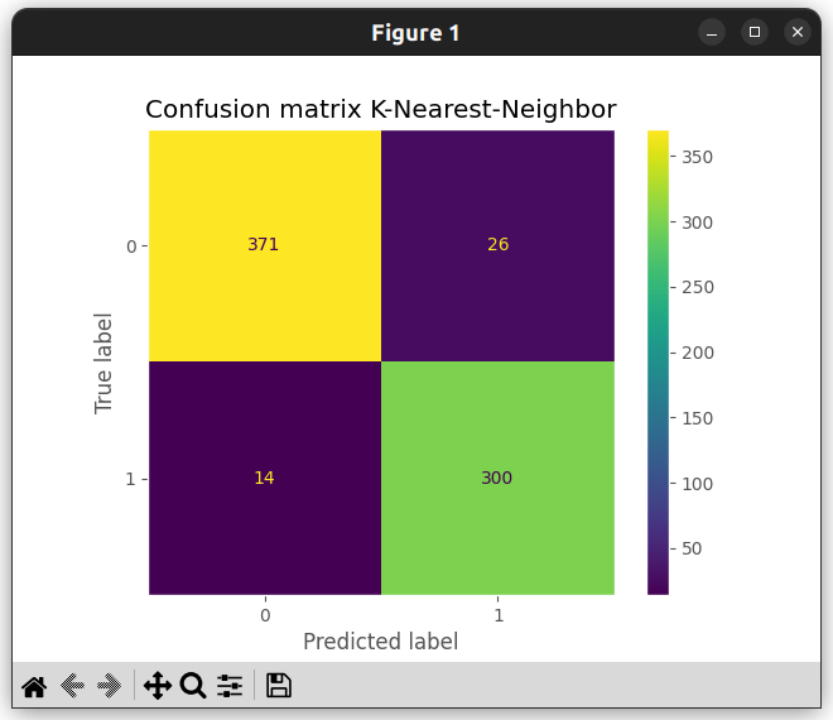


Nel K-nearest neighbor variano i neighbors. In questo caso le performance sono molto altalenanti.

Utilizziamo una Matrice di Confusione per analizzare gli errori compiuti da un modello di machine learning. Più nello specifico è utile per valutare la qualità delle previsioni del modello di classificazione. In particolar modo mette in evidenza, dove sbaglia il modello, in quali istanze risponde meglio e in quali peggio.

Possiamo distinguere quattro casi:

- True Positive (TP);
- True Negative (TN);
- False Positive (FP);
- False Negative (FN).



In questo caso abbiamo 371 casi true-positive, ovvero che appartengono alle persone la cui predizione dell'anemia ha dato esito positivo. Abbiamo 300 true-negative, quindi il modello ha prodotto correttamente che questi soggetti non presentano l'anemia, mentre i 26 false-positive ed i 14 false-negative non sono stati predetti correttamente.

Il nostro obiettivo ora, dopo aver studiato il comportamento e preso visione dei risultati del KNN, è quello di verificare, se utilizzando modelli diversi dal KNN, riusciamo ad ottenere risultati migliori che contengano meno False Positive (FP) e False Negative (FN).

3.2 Decision tree

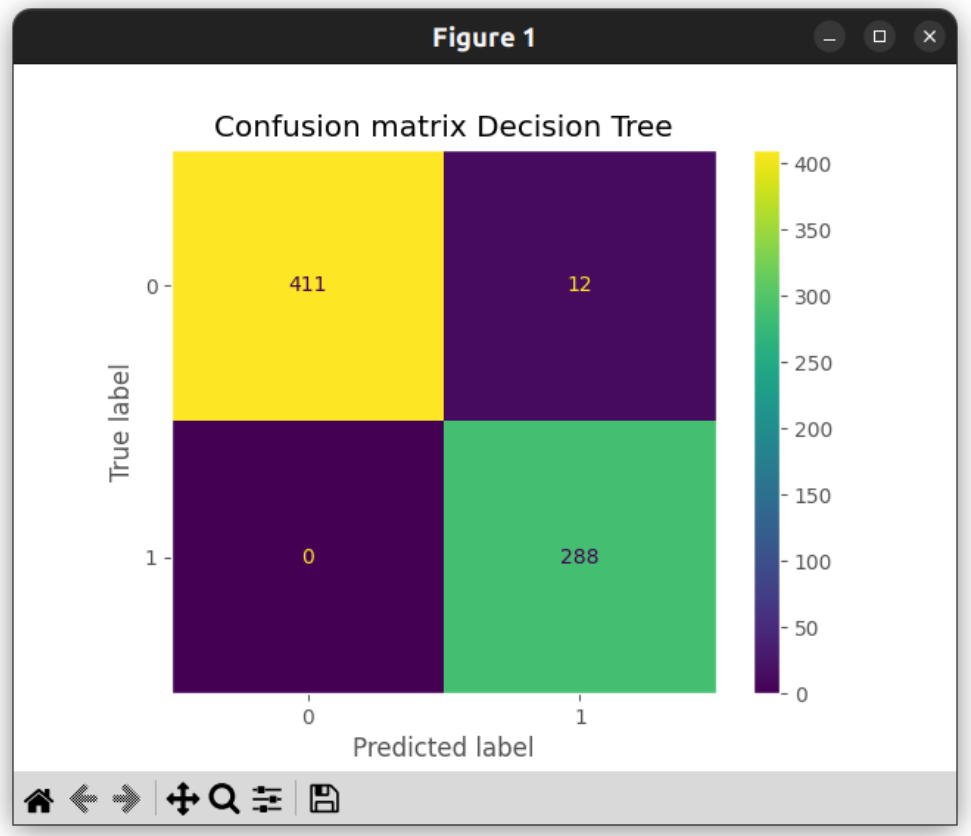
Gli alberi di decisione sono una tecnica di apprendimento automatico (machine learning) che viene utilizzata per la classificazione o la regressione di dati. Si basano su una struttura ad albero in cui ogni nodo rappresenta una caratteristica (o una variabile) del dataset e ogni arco rappresenta una regola di decisione basata su quella caratteristica.

Dal nodo dipendono tanti archi quanti sono i possibili valori che la caratteristica può assumere, fino a raggiungere le foglie che indicano la categoria associata alla decisione.

Qui di seguito vengono stampati a video i risultati delle metriche calcolate:

```
Decision tree metrics
Accuracy : 0.989
Precision : 1.000
Recall : 0.975
F1_score : 0.987
```

Utilizziamo nuovamente una Matrice di Confusione per analizzare gli errori compiuti da un modello di machine learning.



In questo caso abbiamo 411 casi true-positive, ovvero che appartengono alle persone la cui predizione dell'anemia ha dato esito positivo. Abbiamo 288 true-negative, quindi il modello ha prodotto correttamente che questi soggetti non presentano l'anemia, 0 false-negative, mentre i 12 false-positive non sono stati predetti correttamente.

Nell'albero di decisione varia la profondità. Le performance crollano considerando un valore di profondità pari all'incirca a 150.

3.3 Logistic regression

La regressione logistica è un modello statistico usato negli algoritmi di classificazione del machine learning per ottenere la probabilità di appartenenza a una determinata classe. L'algoritmo di basa sull'utilizzo della funzione logistica (sigmoid) che converte i valori reali in un valore compreso tra 0 e 1.

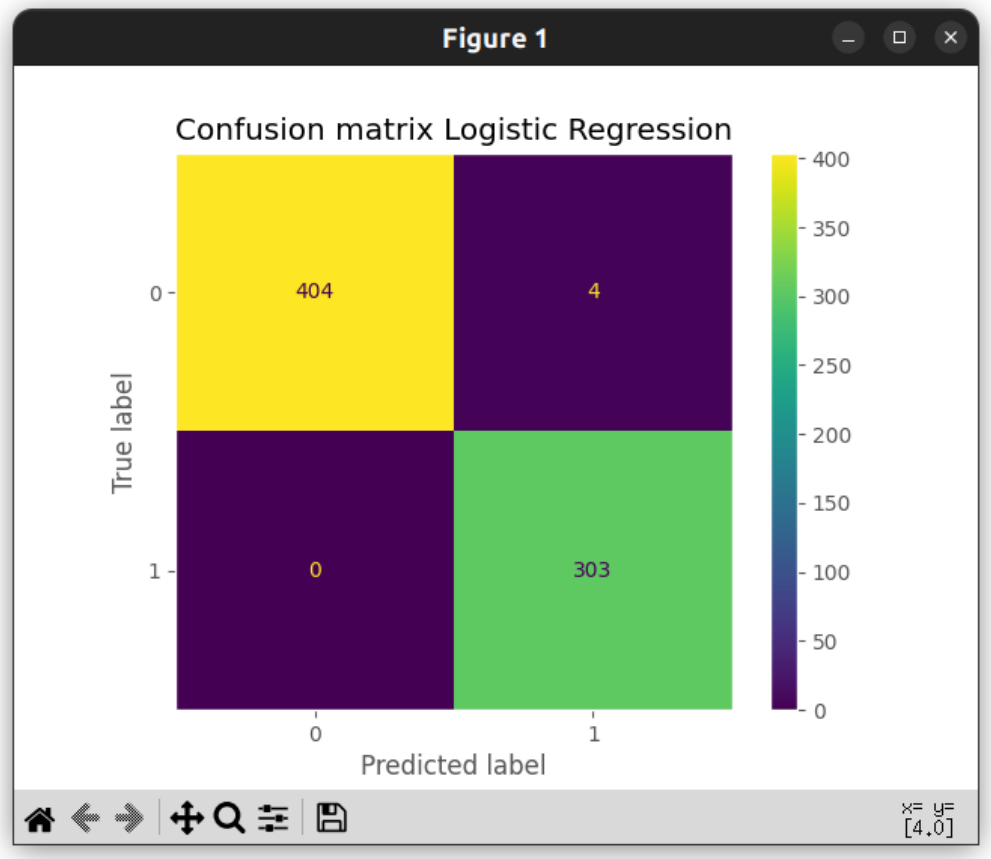
Nella fase di addestramento l'algoritmo riceve in input un dataset di training composta da N esempi.

Ogni esempio è composto da m attributi X e da un'etichetta y che indica la corretta classificazione.

Qui di seguito vengono stampati a video i risultati delle metriche calcolate:

```
Logistic Regression metrics
Accuracy : 0.956
Precision : 0.905
Recall : 1.000
F1_score : 0.950
```

La Matrice di confusione della regressione logistica è la seguente:



Abbiamo 404 casi true-positive, ovvero che appartengono alle persone la cui predizione dell'anemia ha dato esito positivo. Abbiamo 303 true-negative, quindi il modello ha prodotto correttamente che questi soggetti non presentano l'anemia, 0 false-negative, mentre i 4 false-positive non sono stati predetti correttamente.

Nel caso della regressione logistica, il parametro che cambia è il numero di iterazioni fatte dall'algoritmo di classificazione.

4 Considerazioni finali

In conclusione, il sistema realizzato soddisfa gli obiettivi da noi stabiliti inizialmente. Per il sistema esperto con la relativa gestione dell'ontologia, abbiamo riscontrato una funzionalità soddisfacente sia nell'implementazione del CSP con cui gestiamo le prenotazioni delle visite e le prenotazioni dei farmaci e sia con la diagnostica dei sintomi dei pazienti relativa all'ontologia.

Inoltre, abbiamo riscontrato dei buoni risultati riguardo gli algoritmi usati per la fase di apprendimento supervisionato.