Breast Cancer Prediction System

Introduzione

Il progetto è stato realizzato per l'esame di Ingegneria della Conoscenza AA 2022-23 da:

Alberto Saltarelli, matricola 594816, email: a.saltarelli1@studenti.uniba.it

Il sistema è in grado di effettuare una predizione, di tipo probabilistico, su un soggetto potenzialmente a rischio di cancro al seno in base ai valori delle feature presenti nel dataset utilizzato.

Requisiti funzionali

Il progetto è stato scritto interamente in Python (Python 3) utilizzando l'IDE PyCharm sfruttando le seguenti librerie:

- *Scikit-learn*: libreria di apprendimento automatico (contiene algoritmi di classificazione, regressione, SVM, classificatore bayesiano, k-Mean ecc…)
- *Pandas*: per la manipolazione e analisi dei dati
- Numpy: funzioni matematiche per matrici e array multidimensionali
- Matplotlib: per la creazione di grafici
- *NetworkX*: per studiare grafi e reti
- *Pgmpy*: per costruire la rete bayesiana

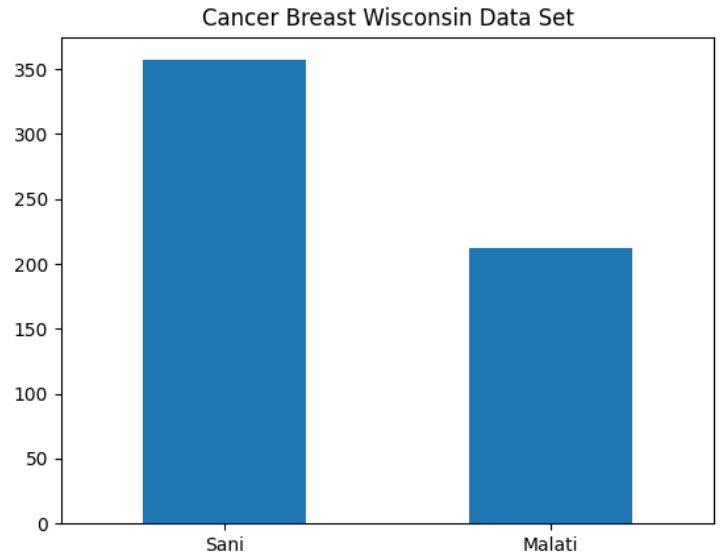
Dopo aver installato tutte le librerie, il progetto può essere eseguito con qualsiasi interprete Python (è stato testato su MacOS da Terminale e attraverso l'IDE PyCharm e su Windows con Visual Studio Code).

Il progetto è disponibile su GitHub al seguente link

Scelte progettuali

Per l'addestramento del sistema di apprendimento automatico è stato utilizzato il data set **Breast Cancer Prediction | Kaggle** disponibile al seguente link.

In fase di avvio il sistema mostra l'intero data set ed effettua un'analisi sul bilanciamento mostrando la numerosità dei pazienti positivi e negativi in percentuale e in forma grafica.



Pazienti malati: 212 (37,26%)

Pazienti sani: 357 (62,74%)

Totale: 569

Apprendimento Supervisionato

Per poter scegliere il modello che più si adatta al data set considerato ho scelto di comparare quattro modelli di classificazione con diversa complessità per trovare il compromesso migliore per gli esempi a disposizione:

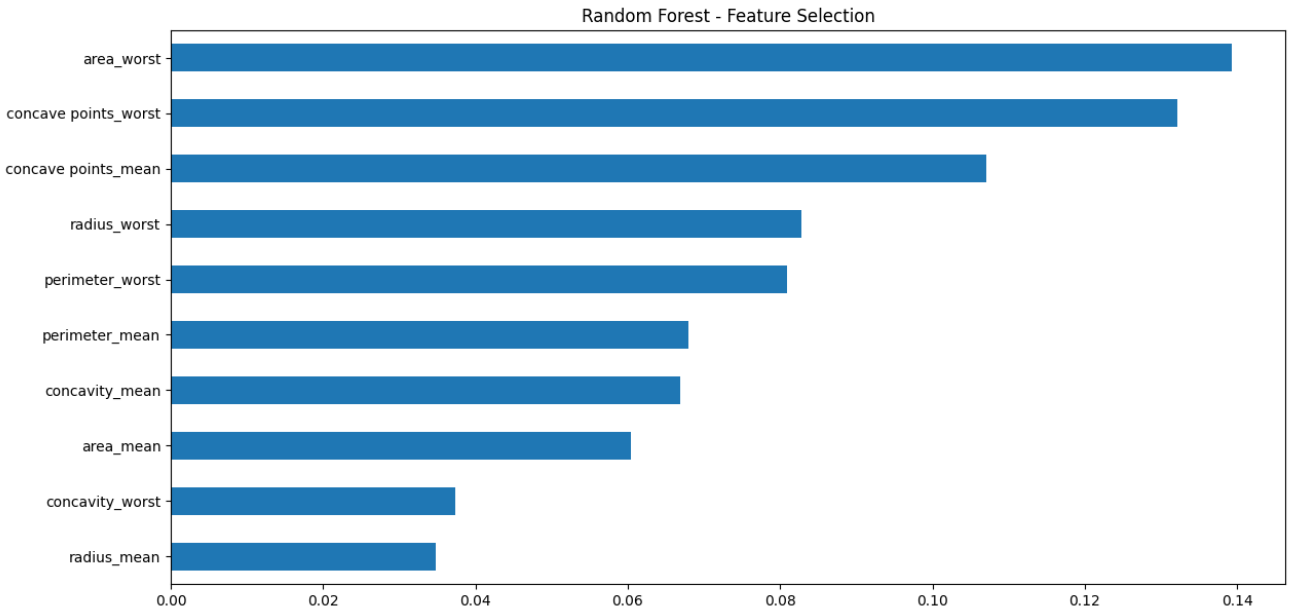
- *k-NN*: algoritmo che si basa sulle feature dei k elementi più vicini al target
- *Decision Tree*: basato su un albero di decisione
- *Random Forest*: modello di ensemble learning composto da più alberi di decisione che effettua il bagging del data set
- *SVM*: basato su kernel, permette di classificare domini non linearmente separabili

Per bilanciare il data set ho deciso di utilizzare il metodo K-Fold Cross Validation, nello specifico lo Stratified K-Fold Cross Validation.
Tale metodo si rivela più specifico per problemi di classificazione preservando il rapporto delle feature target del data set di partenza in tutte le partizioni (in questo progetto è stato utilizzato k = 5).

Classifier	Accuracy	Precision	Recall	F1 Score
KNN	0.929204	0.886364	0.928571	0.906977
Decision Tree	0.902655	0.816327	0.952381	0.879121
Random Forest	0.973451	0.953488	0.976190	0.964706
SVM	0.938053	0.948718	0.880952	0.913580

Si può notare che il Random Forest risulta essere il classificatore che più si adatta al data set utilizzato poiché risulta essere migliore per tutte le metriche utilizzate: accuratezza, precisione, richiamo e F-measure.

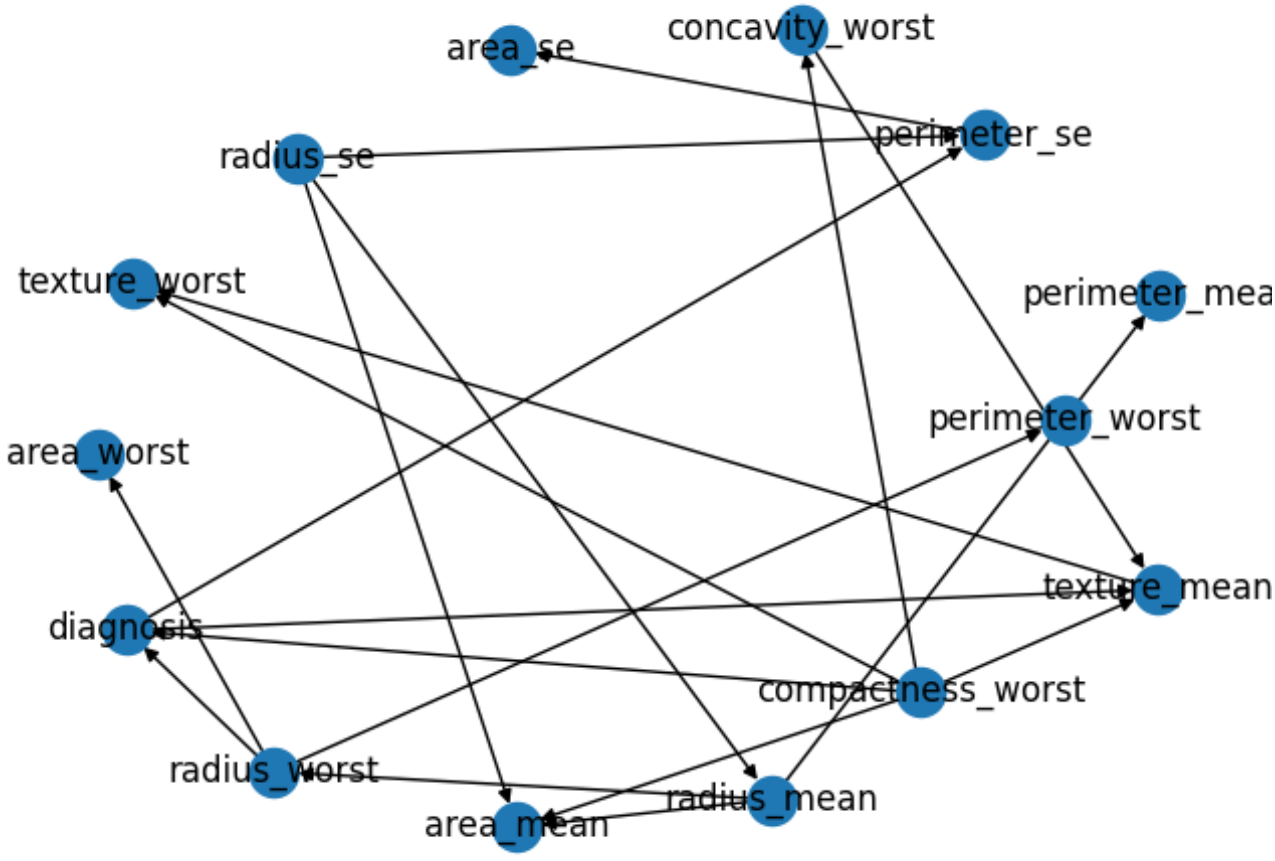
La parte di apprendimento supervisionato si conclude con la valutazione delle feature più rilevanti (*feature selection*) effettuata col classificatore Random Forest che è in grado di calcolare l'importanza di una feature in base alla sua capacità di aumentare la purezza (*purenss*) sulle foglie.



Apprendimento non Supervisionato

Per l'apprendimento non supervisionato ho scelto di far apprendere la topologia della rete bayesiana (*Belief Network*) e la distribuzione di probabilità di ogni feature di input (*CPT*).
Ho utilizzato come metodo di scoring, per trovare il grafo migliore, l'algoritmo *K2-score* che massimizza una misura basata su MAP (Massima probabilità a posteriori) e come stimatore, per calcolare la distribuzione di probabilità di ogni variabile, una funzione di qualità basata su ML (*Maximum Likelihood*, massima verosimiglianza) nota anche come *Hill Climbing.*

Successivamente ho generato un grafico della topologia della rete appresa che mette in evidenza le dipendenze tra variabili del data set:



Infine per testare la rete ho deciso di far eseguire la predizione della malattia su due soggetti di cui uno sano ed uno malato, non facenti parte del data set di training.

```
# Potentially healthy patient
healthy = data.query(variables=["diagnosis"],
                     evidence={"radius_mean": 8, "texture_mean": 25, "perimeter_mean": 48, "area_mean": 181,
                               "radius_se": 0, "perimeter_se": 3, "area_se": 19, "radius_worst": 9,
                               "texture_worst": 30, "perimeter_worst": 59, "area_worst": 269,

# Potentially cancer patient
cancer = data.query(variables=["diagnosis"],
                    evidence={"radius_mean": 21, "texture_mean": 29, "perimeter_mean": 140, "area_mean": 1265,
                              "radius_se": 726, "perimeter_se": 6, "area_se": 86, "radius_worst": 26,
                              "texture_worst": 39, "perimeter_worst": 185, "area_worst": 1821,
                              "compactness_worst": 1, "concavity_worst": 1})
```

I risultati ottenuti sono in linea con quanto atteso: di fatti, come mostrato nelle tabelle di seguito, si evince che per un soggetto potenzialmente sano la probabilità di riscontrare il cancro al seno è del 3,6% mentre per un soggetto potenzialmente malato è pari al 95,6%.

Probabilità di un soggetto sano:

```
+--------------+------------------+
| diagnosis    |   phi(diagnosis) |
+==============|==================+
| diagnosis(0) |           0.9639 |
+--------------+------------------+
| diagnosis(1) |           0.0361 |
+--------------+------------------+
```

Probabilità di un soggetto malato:

```
+--------------+------------------+
| diagnosis    |   phi(diagnosis) |
+==============|==================+
| diagnosis(0) |           0.0441 |
+--------------+------------------+
| diagnosis(1) |           0.9559 |
+--------------+------------------+
```