CASO DI STUDIO ICON 2024

Breast Cancer

Prof. Nicola Fanizzi

AUTORI

Stufano Elisa [737324], e.stufano2@studenti.uniba.it
Duarte Martin Catherine [738731],
c.martinduarte@studenti.uniba.it
Maldera Antonella [758380], a.maldera8@studenti.uniba.it
REPOSITORY GITHUB:
https://github.com/LaureaLaura/ICON-Stufano-Maldera-Martin.git

Sommario

INTRODUZIONE

~INFORMAZIONI TEORICHE SUL PROGETTO:

Il nostro progetto ha l'obiettivo di offrire un supporto medico completo per la diagnosi precoce del tumore al seno, consentendo di determinare tempestivamente se la sua natura sia benigna o maligna. Inoltre, abbiamo cercato di individuare le caratteristiche comuni tra i due tipi di tumore. Per il nostro studio, abbiamo impiegato sia algoritmi di apprendimento supervisionato che non supervisionato, concentrandoci poi sulla costruzione di una rete bayesiana per eseguire le nostre inferenze. Grazie a questi metodi, siamo state in grado di identificare modelli nei dati e formulare previsioni basate sulle informazioni disponibili.

~INFORMAZIONI TECNICHE SUL PROGETTO

Il progetto è stato realizzato con il linguaggio Python in Visual Studio Code. Le librerie utilizzate sono state le seguenti:

- **sklearn** -- per gli algoritmi di apprendimento e la loro valutazione;
- **pandas** e **numpy** -- per la manipolazione dei dati;
- **matplot** e **seaborn** -- per la rappresentazione grafica dei dati;
- **pgmpy** -- per lavorare con modelli grafici.
- **networkx** - per la rappresentazione del modello bayesiano.

ORGANIZZAZIONE DEL DATASET

Link del dataset utilizzato*:* Breast Cancer Dataset | Kaggle

Il tumore al seno è tra le forme di cancro più diffuse nelle donne, costituendo circa il 25% di tutte le diagnosi. Una delle maggiori difficoltà nella sua gestione consiste nel classificare i tumori come maligni o benigni, un fattore determinante per una diagnosi precoce e per scegliere il trattamento più adeguato. Di seguito presentiamo i dettagli del dataset utilizzato, composto da 569 righe e 32 colonne.

```
Campi descritti dalle colonne del dataset:
 Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')

Dimensioni del dataset:  (569, 32)
```

- **id**: Un identificatore univoco per ciascun paziente o caso clinico.
- **diagnosis**: Classificazione del tumore come **benigno (B)** o **maligno (M)**.
- **radius_mean**: Media del raggio del tumore.
- **texture_mean**: Media della variazione nei livelli di grigio (texture) all'interno dell'immagine.
- **perimeter_mean**: Media del perimetro del tumore.
- **area_mean**: Media dell'area del tumore.
- **smoothness_mean**: Media della levigatezza dei bordi del tumore.
- **compactness_mean**: Media della compattezza del tumore.
- **concavity_mean**: Media delle concavità del tumore.
- **concave points_mean**: Media del numero di punti concavi lungo il bordo del tumore.
- **symmetry_mean**: Media della simmetria del tumore.
- **fractal_dimension_mean**: Media della dimensione frattale.
- **radius_se**: Errore standard del raggio, che indica la variabilità nelle misurazioni del raggio tra le diverse immagini.
- **texture_se**: Errore standard della texture.
- **perimeter_se**: Errore standard del perimetro.
- **area_se**: Errore standard dell'area, che rappresenta la variabilità nelle misurazioni dell'area del tumore.
- **smoothness_se**: Errore standard della levigatezza dei bordi.
- **compactness_se**: Errore standard della compattezza.
- **concavity_se**: Errore standard delle concavità del bordo del tumore.
- **concave points_se**: Errore standard del numero di punti concavi lungo il bordo del tumore.
- **symmetry_se**: Errore standard della simmetria del tumore.
- **fractal_dimension_se**: Errore standard della dimensione frattale.
- **radius_worst**: Il valore massimo del raggio osservato tra le varie immagini.
- **texture_worst**: Il valore massimo della texture (variazione dei livelli di grigio) osservato.
- **perimeter_worst**: Il valore massimo del perimetro osservato.
- **area_worst**: L'area massima osservata del tumore.
- **smoothness_worst**: Il valore massimo della levigatezza osservato.
- **compactness_worst**: Il valore massimo della compattezza osservato.
- **concavity_worst**: Il valore massimo della concavità osservato.
- **concave points_worst**: Il massimo numero di punti concavi osservato lungo il contorno del tumore.
- **symmetry_worst**: Il valore massimo della simmetria osservato.
- **fractal_dimension_worst**: Il valore massimo della dimensione frattale osservato.

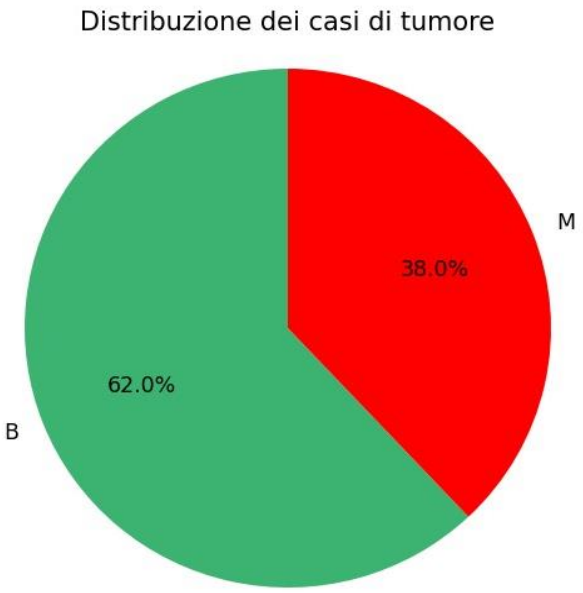
ANALISI E PRE-ELABORAZIONE DEI DATI

La colonna 'id' è inutile ai fini della nostra analisi, quindi è stata eliminata. Le colonne sono tutte di valori continui, tranne la colonna "diagnosis" (la nostra feature target), che è una variabile discreta.

```
Le colonne di tipologia discreta sono:

['diagnosis']


Le colonne di tipologia continua sono:

['radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean', 'concav
e points_mean', 'symmetry_mean', 'fractal_dimension_mean', 'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se
', 'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se', 'fractal_dimension_se', 'radius_worst', 'texture_worst
', 'perimeter_worst', 'area_worst', 'smoothness_worst', 'compactness_worst', 'concavity_worst', 'concave points_worst', 'symmet
ry_worst', 'fractal_dimension_worst']
```

Successivamente, abbiamo controllato la presenza di valori NULL in ciascuna colonna e utilizzato un grafico a torta per analizzare la distribuzione del dataset rispetto alla variabile target "*diagnosis*", che presenta due categorie: **B** per indicare un tumore benigno e **M** per un tumore maligno.

```
diagnosis                0
radius_mean              0
texture_mean             0
perimeter_mean           0
area_mean                0
smoothness_mean          0
compactness_mean         0
concavity_mean           0
concave points_mean      0
symmetry_mean            0
fractal_dimension_mean   0
radius_se                0
texture_se               0
perimeter_se             0
area_se                  0
smoothness_se            0
compactness_se           0
concavity_se             0
concave points_se        0
symmetry_se              0
fractal_dimension_se     0
radius_worst             0
texture_worst            0
perimeter_worst          0
area_worst               0
smoothness_worst         0
compactness_worst        0
concavity_worst          0
concave points_worst     0
symmetry_worst           0
fractal_dimension_worst  0
```



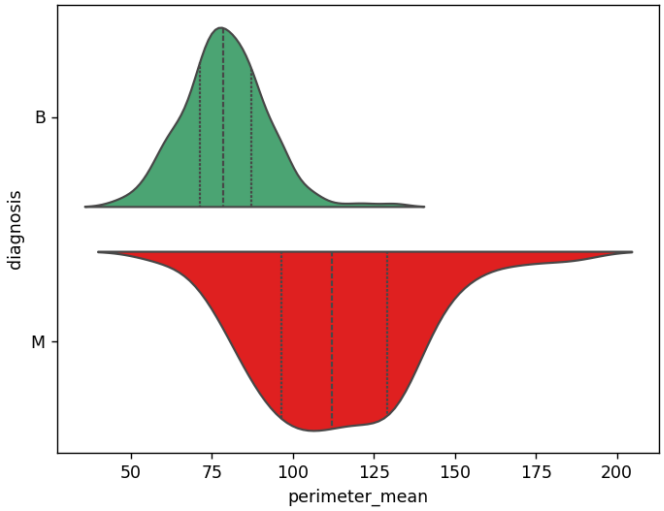
Non sono presenti valori nulli nelle colonne del dataset.

OSSERVAZIONE GRAFICA DEI DATI

Abbiamo effettuato delle analisi grafiche per valutare la correlazione tra i dati e comprendere in che modo questi influissero sulla diagnosi. Dopo aver effettuato qualche ricerca, abbiamo trovato che le caratteristiche più influenti risultano essere: Perimetro medio, Area media, Raggio medio, Texture media, Levigatezza media e Punti concavi medi.

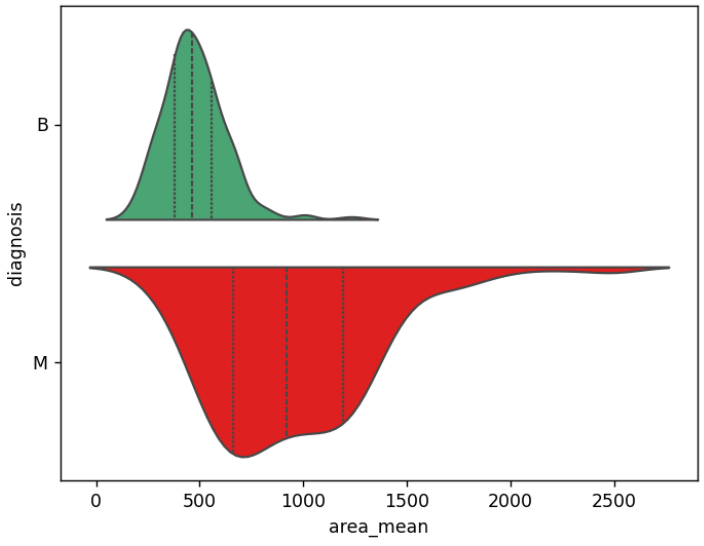
Abbiamo quindi analizzato alcuni dei grafici che abbiamo creato:

Distribuzione casi di tumori rispetto livelli di perimetro medio del tumore:

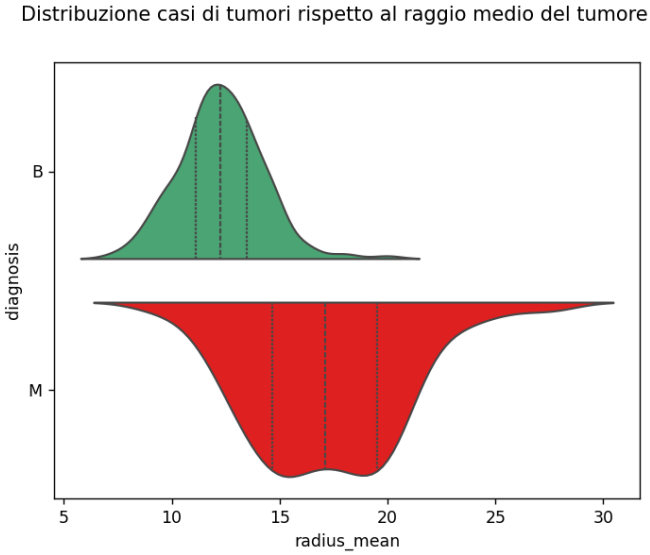


Dal grafico emerge che un **perimetro** compreso tra 50 e 75 è generalmente associato a tumori benigni; al contrario, allontanandosi da questo intervallo, i tumori tendono a essere più frequentemente maligni.

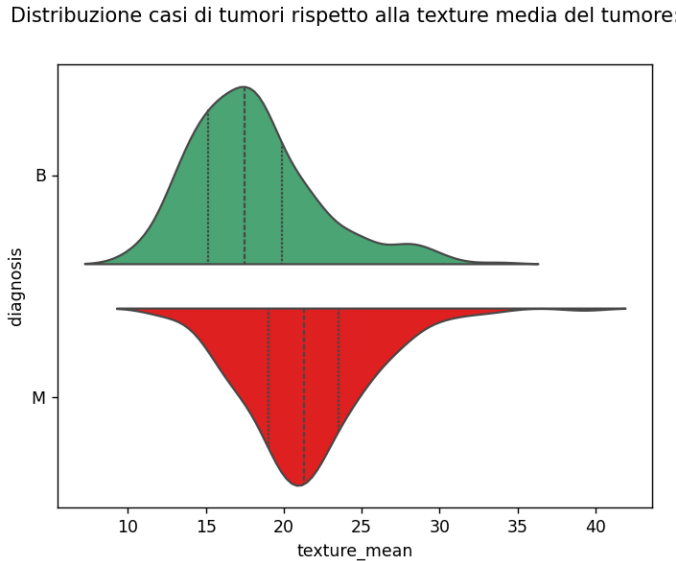
Distribuzione casi di tumori rispetto livelli di area media del tumore:



In questo grafico, abbiamo correlato **l'area media** del tumore con la diagnosi, e i risultati indicano che un'area media di 500 è fortemente associata a tumori benigni. Al contrario, la distribuzione dei tumori maligni inizia a partire da un'area di 500 e oltre.

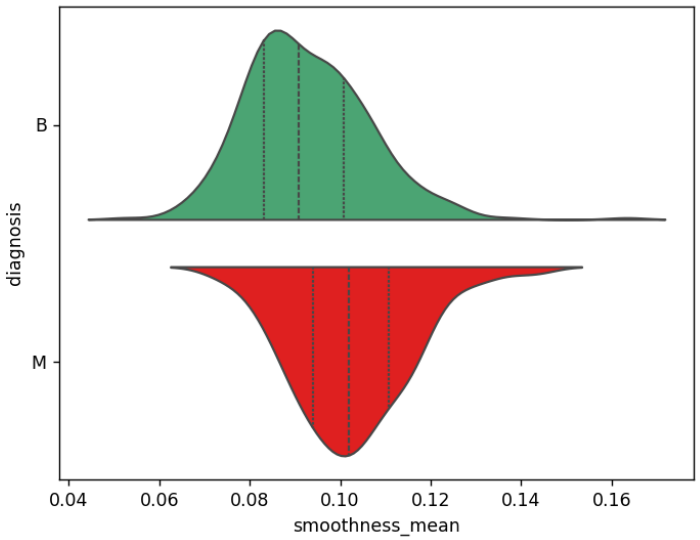


La maggior parte dei tumori benigni ha un **raggio medio** compreso tra 10 e 15. La distribuzione ha una forma simmetrica con un picco attorno ai 12-13, indicando che la maggioranza dei tumori benigni ha un raggio medio in questo intervallo. I tumori maligni tendono ad avere un raggio medio più elevato rispetto ai benigni. La distribuzione è più diffusa, coprendo valori compresi tra circa 15 e 25.



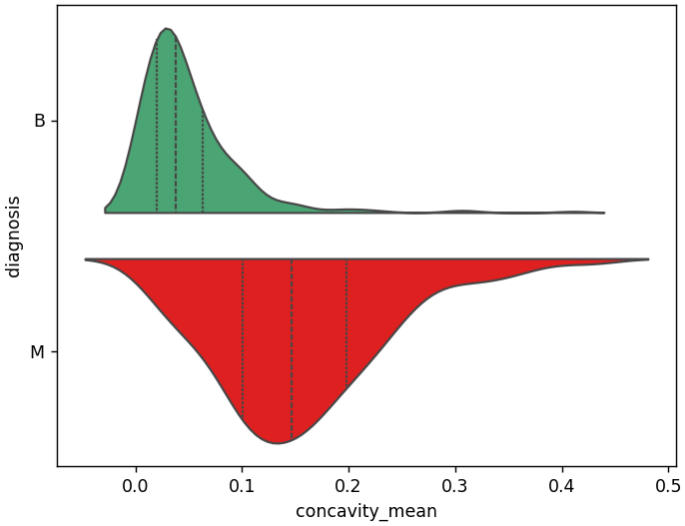
La maggior parte dei tumori benigni ha un valore di **texture_mean** compreso tra 15 e 25. C'è un picco attorno ai 17-18, il che indica che la maggior parte dei tumori benigni ha una texture media intorno a questo valore. La distribuzione dei tumori maligni è più concentrata su valori più alti rispetto a quella dei benigni, con un picco intorno ai 22-23. È interessante notare che la distribuzione dei maligni è più stretta rispetto ai benigni.

Distribuzione casi di tumori rispetto alla legivatezza media del tumore:



La **smoothness_mean** per i tumori benigni è concentrata tra 0.06 e 0.12, con un picco intorno a 0.09-0.10. Tuttavia, anche per i tumori maligni, la smoothness_mean è compresa tra 0.06 e 0.12, con una concentrazione simile a quella dei benigni intorno a 0.10.
La levigatezza media non sembra avere una grande capacità di distinguere tra tumori benigni e maligni, poiché le distribuzioni delle due categorie si sovrappongono quasi completamente. Entrambi i tipi di tumore hanno valori molto simili, con lievi differenze nei picchi, ma non abbastanza da costituire un indicatore significativo per la diagnosi.

Distribuzione casi di tumori rispetto alla concavità media del tumore:

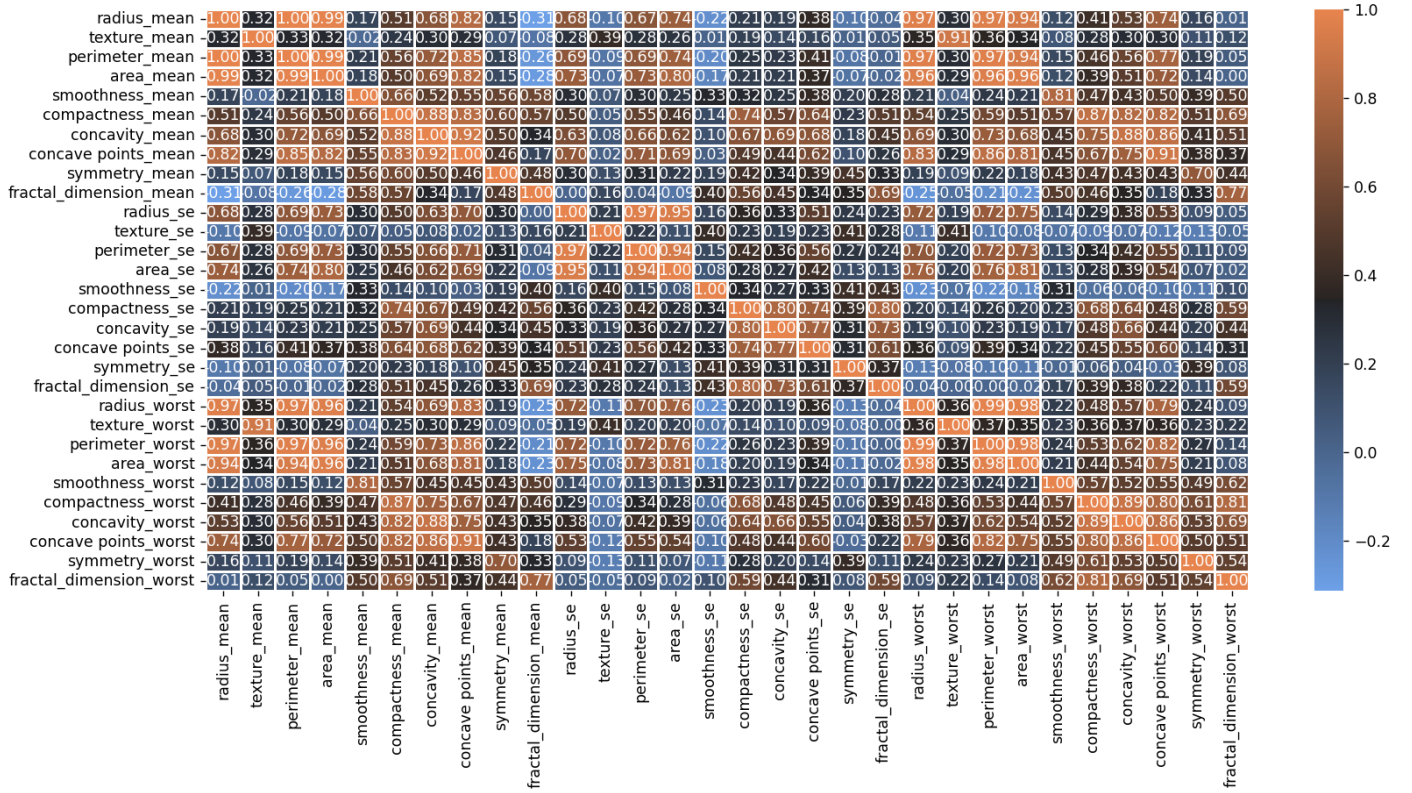


L'ultima relazione che abbiamo scelto di esaminare riguarda la **concavità** media del tumore. I risultati, come nei casi precedenti, mostrano che valori più bassi sono associati a tumori benigni, mentre valori più elevati indicano la presenza di tumori maligni. In questo caso, da 0.1 in poi i casi sono quasi totalmente maligni.

Per concludere la nostra analisi grafica, abbiamo realizzato una 'heatmap' che ci consente di visualizzare in modo chiaro e completo le relazioni tra i dati.

Queste sono state alcune delle nostre osservazioni:

- **radius_mean e perimeter_mean**: hanno un valore di correlazione di 0.97, evidenziando così una forte relazione;
- **area_mean e perimeter_mean**: anch'essi mostrano una correlazione alta di 0.96, suggerendo che un'area maggiore è associata a un perimetro maggiore.
- **concavity_mean e smoothness_mean**: con un valore di -0.69, indicano che maggiore è la concavità media, minore è la liscezza media del tumore.
- **radius_worst e symmetry_worst**: con un valore di -0.51, suggeriscono che tumori con un raggio più grande tendono ad avere una simmetria peggiore.
- Variabili come **fractal_dimension_se** e **fractal_dimension_worst** mostrano correlazioni basse con molte delle altre caratteristiche, indicando che queste metriche potrebbero non essere fortemente influenzate dalle dimensioni o dalla forma del tumore.



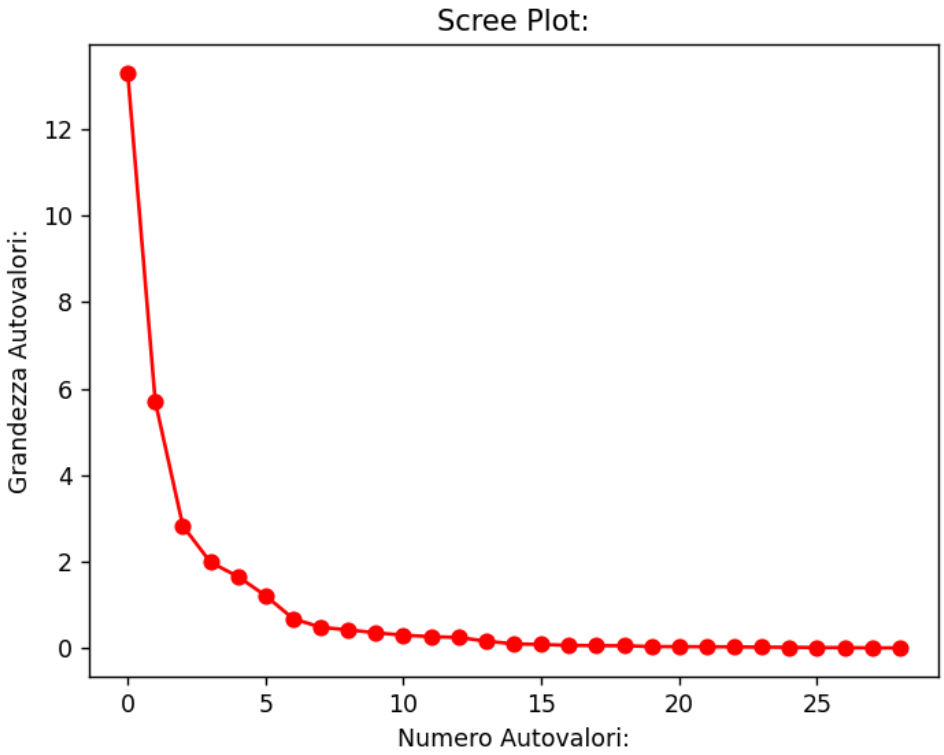
Prima di procedere con le tecniche di apprendimento supervisionato e non supervisionato, **abbiamo realizzato una versione standardizzata del dataset**. Inizialmente, abbiamo selezionato le colonne contenenti valori continui e successivamente abbiamo applicato la funzione **'Standard Scaler'** per garantire una corretta standardizzazione.

APPRENDIMENTO NON SUPERVISIONATO

~PRINCIPAL COMPONENT ANALYSIS:

Inizialmente, abbiamo applicato la PCA ai dati standardizzati. Il dataset utilizzato contiene un numero elevato di variabili (32 colonne), il che potrebbe complicare l'analisi e rendere difficile l'individuazione di pattern significativi. La PCA ci consentirà di ridurre il numero di variabili a un insieme più gestibile, mantenendo le informazioni più rilevanti. Inoltre, riducendo le dimensioni, sarà più facile visualizzare e

interpretare i risultati.
La PCA ci aiuta anche a filtrare il rumore e le variazioni casuali nei dati, migliorando la qualità delle inferenze che si possono trarre. Abbiamo osservato che questo è particolarmente utile in ambito medico, dove i dati possono essere influenzati da misurazioni imprecise o errori.
Abbiamo scelto di utilizzare il dataset standardizzato poiché i test e le metriche indicavano che tali dati miglioravano le performance dell'algoritmo di clustering utilizzato. Per determinare il numero ottimale di componenti principali, abbiamo impiegato lo 'Screeplot'.



Per stabilire con precisione il numero di componenti principali da includere, abbiamo fatto riferimento anche alla **Regola di Kaiser**, secondo la quale si considerano nel modello finale tutte le componenti che presentano un autovalore maggiore o uguale a 1. Così, sono state prese come principali componenti le prime 6:

```
Autovalori:
[13.305   5.7014  2.8229  1.9841  1.6516  1.2095  0.6764  0.4775  0.4176
  0.3513  0.2944  0.2616  0.2418  0.1573  0.0943  0.08    0.0595  0.0527
  0.0496  0.0312  0.03    0.0275  0.0244  0.0181  0.0155  0.0082  0.0069
  0.0016  0.0008]
```

~KMEDOIDS:

Abbiamo utilizzato l'algoritmo K-medoids per il clustering, impiegando la funzione **KMedoids()**. Questo approccio costituisce un'alternativa al **K-means**, poiché si fonda sulla minimizzazione della somma delle differenze tra i punti all'interno di un cluster e un punto designato come medoide di quel cluster. Il medoide è il punto che minimizza la somma delle distanze agli altri punti del cluster. Questo rende K-Medoids più

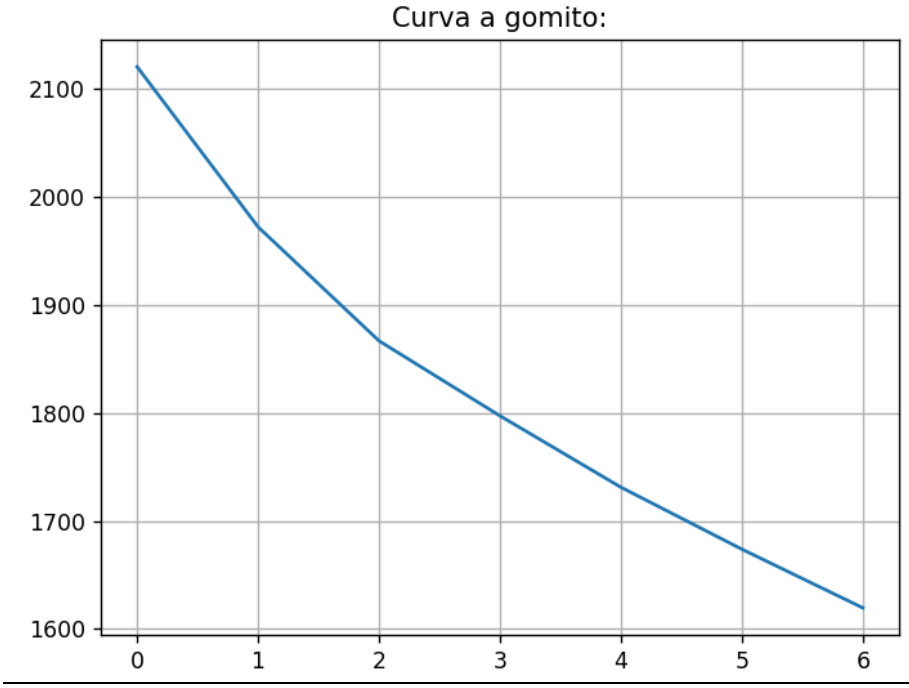
robusto rispetto agli outlier e ai dati rumorosi, perché il medoide è sempre un punto reale.

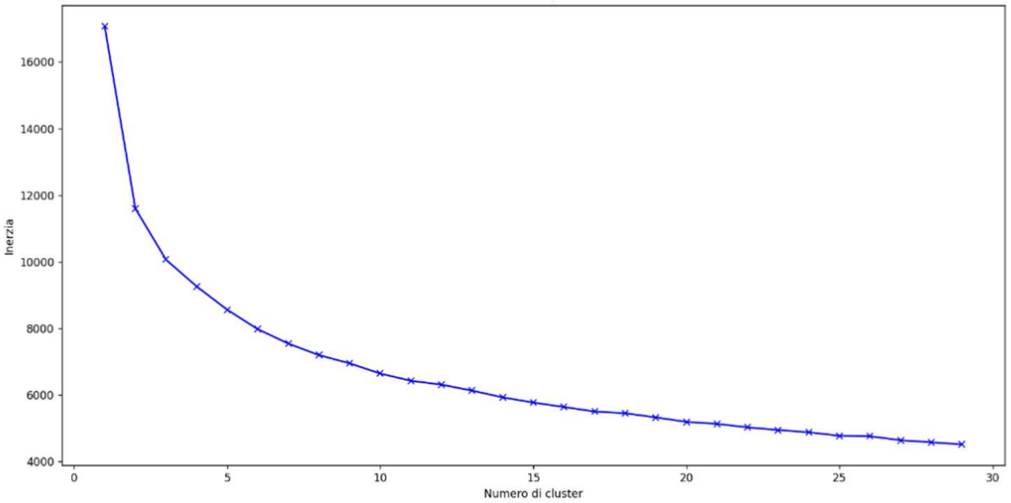
Abbiamo adottato l'algoritmo Partitioning Around Medoids (**PAM**) come metodo specifico per implementare il K-medoids. L'algoritmo PAM è un metodo ben consolidato e noto per la sua efficienza e facilità d'uso.

L'algoritmo si sviluppa nelle seguenti fasi:

- **Inizializzazione**: selezioniamo casualmente k punti dati come medoidi iniziali.
- **Fase di assegnazione**: assegniamo ciascun punto dati al medoide più vicino.
- **Passo di aggiornamento**: per ogni medoide e ogni punto dati associato ad esso, scambiamo il medoide con il punto dati e calcoliamo il costo totale della configurazione (ovvero la differenza media del punto dati rispetto a tutti i punti dati associati al medoide). Selezioniamo il punto dati con il costo più basso come nuovo medoide.

Ripetiamo i passaggi 2 e 3 fino a quando non osserviamo più variazioni nelle assegnazioni dei punti ai cluster. Per stabilire il numero ottimale di cluster (k), abbiamo applicato metodi analitici, tra cui la **curva del gomito**, che fornisce una rappresentazione visiva intuitiva dei risultati, facilitando la comprensione di come la variazione spiegata cambia in funzione del numero di cluster. Così, si può facilmente identificare il punto di flesso, dove i miglioramenti nella compattezza dei cluster cominciano a stabilizzarsi. Inoltre, è una tecnica consolidata e versatile nel campo del clustering. Questo approccio mira a identificare il punto in cui, oltre un certo valore di k, non si registra più una variazione significativa in uno specifico punteggio. Nel nostro caso, abbiamo utilizzato **l'inertia**; valutando l'inertia in relazione a k, possiamo comprendere come le distanze interne ai cluster variano, permettendo di scegliere un k che minimizza l'inertia in modo significativo senza una riduzione eccessiva.





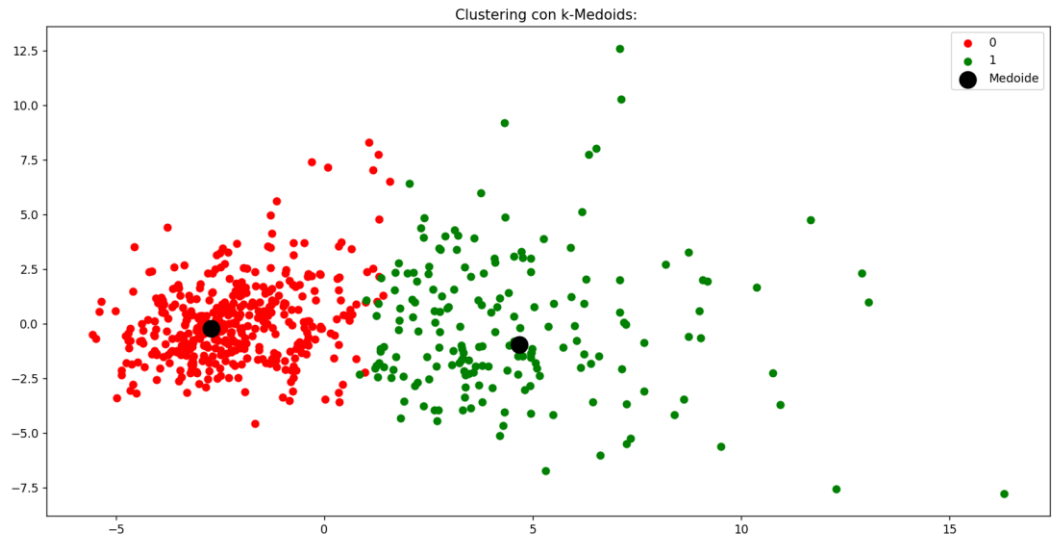
Dalla curva si può osservare che il punto di gomito si trova a 2. Tuttavia, prima di procedere, abbiamo scelto di convalidare questa scelta utilizzando un ulteriore metodo: il *coefficiente di Silhouette*.

~COEFFICIENTE DI SILHOUETTE:

Abbiamo impiegato il coefficiente di Silhouette come misura per valutare la qualità dei risultati del clustering. Il coefficiente varia tra -1 e 1, rendendo facile interpretare i risultati. I punteggi possono essere facilmente confrontati, consentendo di identificare il numero ottimale di cluster e valutare la qualità del clustering in modo intuitivo.
Abbiamo deciso di utilizzarlo anche perché è relativamente facile da calcolare e integrare nel flusso di lavoro di clustering, consentendo una rapida valutazione della qualità del clustering senza richiedere metodi complessi o onerosi in termini di risorse. Questo approccio ci assicura che le decisioni prese siano informate e basate su analisi solide.
Un valore più alto del coefficiente di silhouette indica una migliore separazione dei cluster:

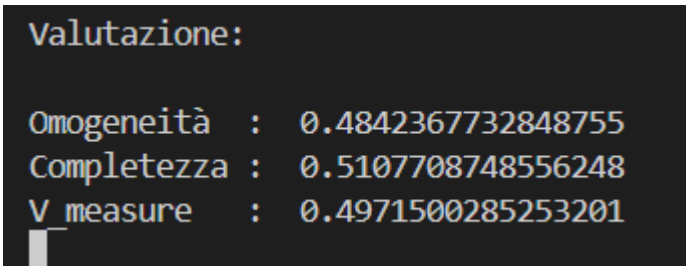
```
Con n_clusters=2, il valore di silhouette 0.3791549355276975
Con n_clusters=3, il valore di silhouette 0.3170743665319534
Con n_clusters=4, il valore di silhouette 0.19110067436236836
Con n_clusters=5, il valore di silhouette 0.1782742099431518
Con n_clusters=6, il valore di silhouette 0.16790482349781574
Con n_clusters=7, il valore di silhouette 0.1488589227875741
Con n_clusters=8, il valore di silhouette 0.143655830488803
```

Poiché sia il Silhouette score che la regola del gomito ci hanno dato come risultato ottimale 2, abbiamo deciso di addestrare l'algoritmo partizionando il dataset in **2 cluster**.
Di seguito riportiamo il clustering eseguito con KMedoids:



Il grafico mostra una separazione chiara tra i due cluster, suggerendo che il modello ha identificato correttamente due gruppi distinti di dati, con i punti che sembrano distribuiti attorno ai medoide.

Metriche di valutazione del clustering:



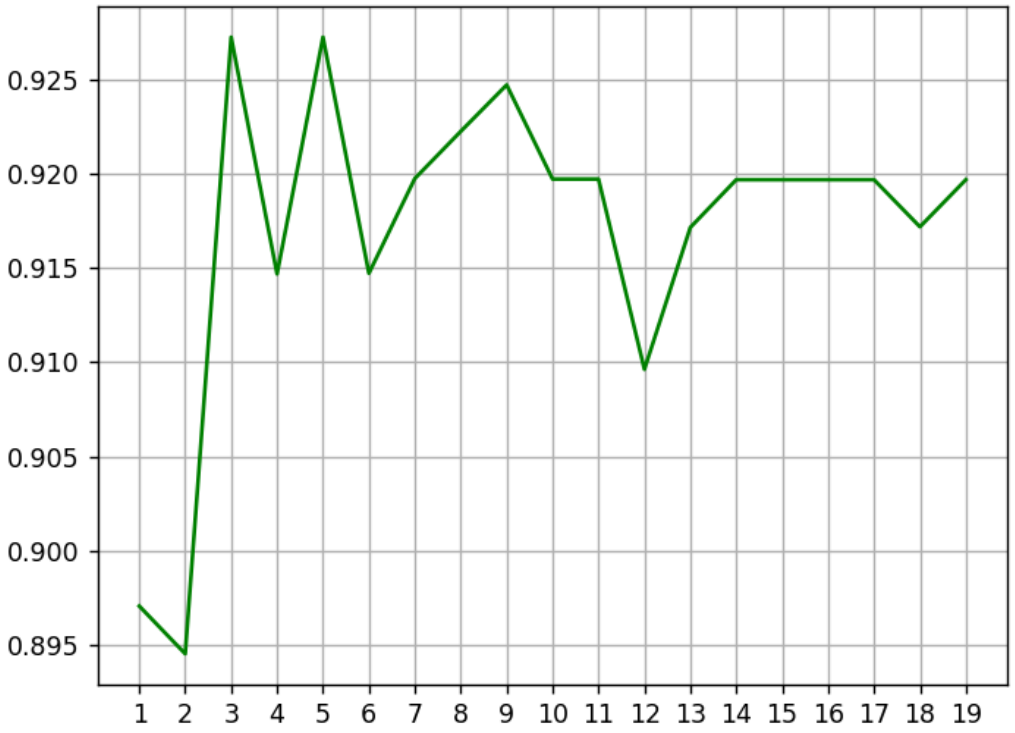
Abbiamo utilizzato diverse misure per valutare la qualità dei risultati del clustering, tra cui l'omogeneità (homogeneity_score), la completezza (completeness_score) e la misura V (v_measure_score).
All'omogeneità, un valore di 0.4842 ci suggerisce che i cluster sono moderatamente omogenei, ma non perfettamente.
Circa il 51% dei membri di ciascuna classe è stato raggruppato correttamente all'interno dello stesso cluster. Tuttavia, essendo solo leggermente superiore al 50%, suggerisce che ci sono margini di miglioramento nell'assegnazione delle classi ai cluster.
Alla V_measure, con un punteggio di 0.4972, i risultati mostrano una qualità moderata nel clustering.

APPRENDIMENTO SUPERVISIONATO

~KNEIGHBOURS CLASSIFIER

KNeighborsClassifier(): Il KNeighborsClassifier ci garantisce un approccio ottimale per il problema di classificazione supervisionata. Abbiamo scelto questo metodo poiché sfrutta i punti di forza dell'algoritmo KNN e assicura che il modello sia bilanciato e in grado di generalizzare bene sui dati non visti.

Per calcolare la distanza tra gli oggetti, di solito viene utilizzata la distanza euclidea come metrica predefinita (distanza di Minkowski con p=2). Inoltre, è la metrica predefinita nel KNN, in quanto riflette in modo naturale la distanza tra due punti nello spazio n-dimensionale. Durante l'esecuzione dell'algoritmo, è stato utilizzato il dataset standardizzato, poiché è più adatto per gli algoritmi basati sulla distanza. Nel nostro caso la standardizzazione ci assicura che tutte le caratteristiche contribuiscano equamente al calcolo della distanza, evitando che quelle con range di valori più ampi abbiano un peso sproporzionato.
Per determinare il valore di k da utilizzare, l'algoritmo è stato addestrato variando il numero di vicini da 1 a 20, consentendoci così di trovare il miglior compromesso tra complessità e capacità predittiva.
Ad ogni iterazione è stata applicata la tecnica della cross-validation. Più precisamente, è stata utilizzata una 5-fold cross-validation perché ci permette di considerare l'intero dataset e di valutare il modello su diverse suddivisioni del dataset, riducendo la variabilità dei risultati e ottenendo una stima affidabile delle performance del modello.
Come metrica di valutazione abbiamo scelto l'F1-score, perché combina precision e recall in un solo valore. L'algoritmo è stato eseguito per tutte le combinazioni di k e il risultato finale è dato dalla media dei punteggi ottenuti.
Durante la cross-validation, è stato selezionato il valore di k corrispondente al punto in cui l'algoritmo ha raggiunto il valore F1 più elevato, che rappresenta un equilibrio tra precision e recall.
In conclusione, l'algoritmo k-nearest neighbors è stato addestrato mediante cross-validation per identificare il numero ottimale di vicini, selezionando il valore di k che ha ottenuto il punteggio F1 più elevato, garantendo così il miglior equilibrio tra precisione e richiamo.

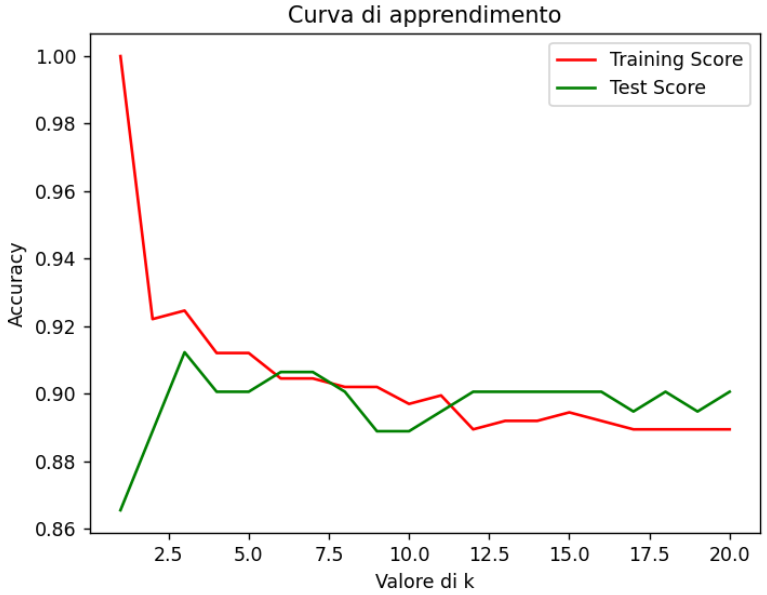


Osservando la curva delle performance ottenuta durante il processo di cross-validation, è possibile notare che k=3 è il primo picco della curva, indicando il punto in cui l'algoritmo raggiunge per la prima volta un livello di accuratezza o bilanciamento tra precision e recall ottimale. Per questo lo abbiamo scelto, così evitiamo anche di rischiare l'overfitting.

+ Risultati test per KNeighborsClassifier(n_neighbors=3) :

	precision	recall	f1-score	support
B	0.96	1.00	0.98	107
M	1.00	0.94	0.97	64
accuracy			0.98	171
macro avg	0.98	0.97	0.97	171
weighted avg	0.98	0.98	0.98	171

Abbiamo successivamente valutato il rischio di overfitting tramite la curva di apprendimento, che ci ha permesso di confrontare le performance del modello su dati di training e di test. Dall'analisi della curva, non sono emersi segnali di overfitting, indicando che il modello riesce a generalizzare bene anche su dati non visti e che le sue prestazioni non sono influenzate da un eccessivo adattamento ai dati di addestramento.


Curva di apprendimento
Training Score
Test Score
Accuracy
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Valore di k

~DECISION TREE CLASSIFIER

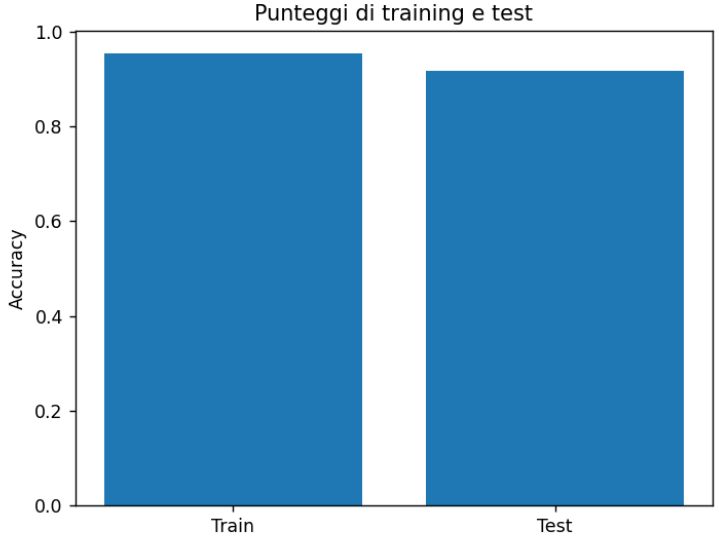
DecisionTreeClassifier() :

Abbiamo utilizzato il **DecisionTreeClassifier** con il criterio di **entropia** nel nostro progetto poiché consente di massimizzare il guadagno informativo ad ogni suddivisione. Questa scelta è giustificata dalla necessità di gestire al meglio il dataset, dove è importante ridurre l'incertezza nei dati e creare suddivisioni che migliorino la capacità del modello di generalizzare.
La scelta dell'entropia permette inoltre di avere un controllo più fine sul processo decisionale del modello rispetto ad altri criteri, come il Gini. Pur essendo leggermente più costoso in termini computazionali, l'**entropia** può migliorare la purezza dei nodi e le prestazioni del modello. In sintesi, l'entropia offre una soluzione più appropriata per il nostro caso specifico, garantendo previsioni più precise.
Abbiamo scelto di utilizzare **RandomizedSearchCV** per ottimizzare il processo di ricerca degli iper-parametri in modo più veloce ed efficiente.
Durante l'esecuzione dell'algoritmo, è stato utilizzato il dataset standardizzato. Questa scelta è motivata dal fatto che l'algoritmo di albero decisionale non è vincolato dalle misure di distanza, il che permette una gestione migliore dei valori anomali e dei valori estremi presenti nel set di dati. Inoltre, dai test eseguiti, il dataset standardizzato ha dimostrato di fornire prestazioni migliori per l'algoritmo.

+ Risultati test per DecisionTreeClassifier(criterion='entropy') :

	precision	recall	f1-score	support
B	0.94	0.96	0.95	107
M	0.93	0.89	0.91	64
accuracy			0.94	171
macro avg	0.94	0.93	0.93	171
weighted avg	0.94	0.94	0.94	171

Visualizzazione dell'overfitting:



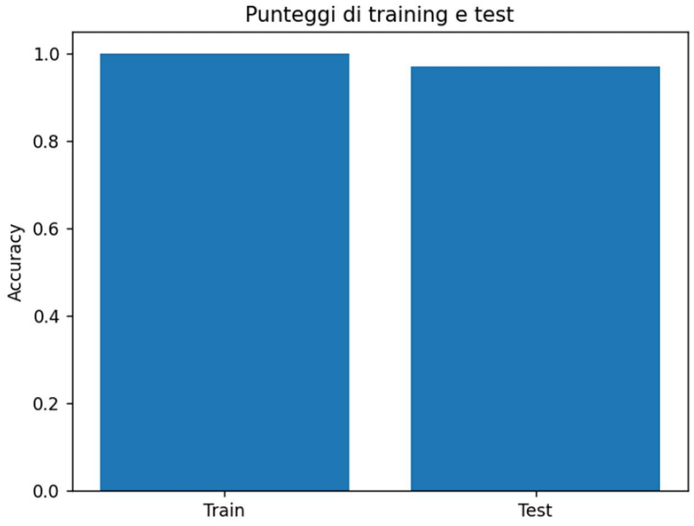
~RANDOM FOREST CLASSIFIER

RandomForestClassifier() : Abbiamo utilizzato il **RandomForestClassifier** per la sua capacità di creare un modello predittivo robusto combinando più alberi decisionali, riducendo il rischio di overfitting e gestendo meglio gli outliers. È stato utilizzato il dataset standardizzato, con la tecnica **RandomizedSearchCV** per ottimizzare gli iper-parametri, tra cui il numero di alberi decisionali (n_estimators). Dai test effettuati, il valore ottimale di 25 estimatori ha garantito le migliori prestazioni, evitando una complessità eccessiva senza ulteriori benefici in termini di accuratezza. Tra i parametri testati, è stato ottimizzato il numero di alberi (**n_estimators**) costruiti nel RandomForest, scegliendo tra i seguenti valori: **[25, 50, 75, 100, 150, 200, 250]**. Durante l'iterazione, è stato selezionato il valore di **25 estimatori**, che ha prodotto le migliori prestazioni del modello.
È risultato, dai test eseguiti, il modello in grado di portare l'algoritmo ad una performance migliore.

Risultati test per RandomForestClassifier(n_estimators=25) :

	precision	recall	f1-score	support
B	0.99	0.98	0.99	107
M	0.97	0.98	0.98	64
accuracy			0.98	171
macro avg	0.98	0.98	0.98	171
weighted avg	0.98	0.98	0.98	171

Visualizzazione dell'overfitting:



~SVM

SVC() : Nel progetto è stato utilizzato l'algoritmo **SVM (Support Vector Machine)** con il modello **SVC()** e, per ottimizzare i parametri, si è ricorso a **RandomizedSearchCV**. La

normalizzazione del dataset è stata fondamentale per ottenere risultati migliori, data la natura **distance-based** dell'algoritmo. Gli SVM, infatti, basano la classificazione sulla distanza dei punti dati rispetto all'**iperpiano** di separazione, rendendo la normalizzazione essenziale per evitare che le feature con range di valori diversi influiscano in maniera sproporzionata.

Attraverso l'uso di **RandomizedSearchCV**, è stata esplorata una gamma di valori per i parametri cruciali dell'SVM:

- **C**: Parametro di regolarizzazione che bilancia il trade-off tra l'accuratezza della classificazione sui dati di addestramento e la generalizzazione del modello. Valori troppo alti possono causare overfitting.
- **Gamma**: Determina l'influenza di un singolo campione. Un valore alto di gamma significa che un campione ha un'influenza vicina, mentre un valore basso implica un'influenza più ampia.

I valori testati erano:

- **C**: [0.001, 0.01, 0.1, 1, 10, 100]
- **Gamma**: [0.001, 0.01, 0.1, 1, 10, 100]

Dopo vari cicli di iterazioni con questi parametri, sono stati scelti i valori:

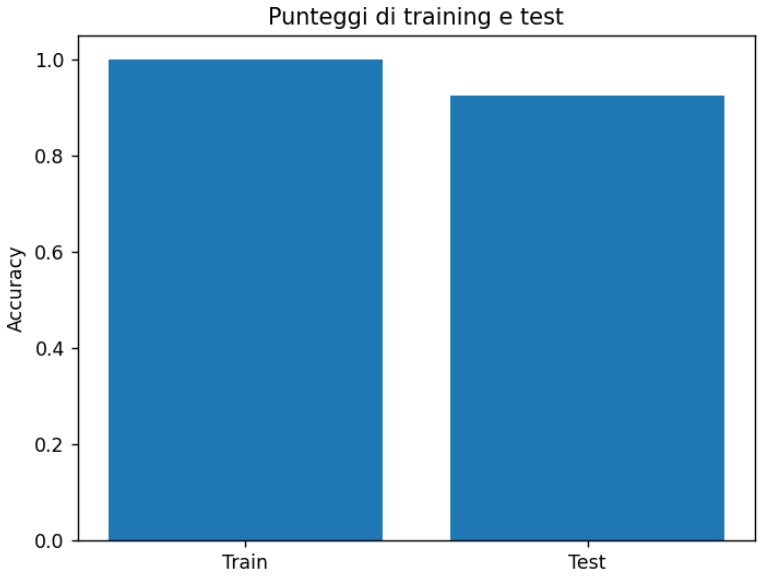
- **C = 1**
- **Gamma = 10**

Questa combinazione è risultata la più efficace, portando il modello a ottenere buoni risultati in termini di precision e recall. I valori selezionati hanno permesso di trovare un equilibrio ottimale tra la capacità di classificazione del modello e la sua generalizzazione ai nuovi dati, minimizzando il rischio di overfitting.

+ Risultati test per SVC(C=1, gamma=10) :

	precision	recall	f1-score	support
B	0.97	0.95	0.96	107
M	0.92	0.95	0.94	64
accuracy			0.95	171
macro avg	0.95	0.95	0.95	171
weighted avg	0.95	0.95	0.95	171

Visualizzazione dell'overfitting:



~REGRESSIONE LOGISTICA

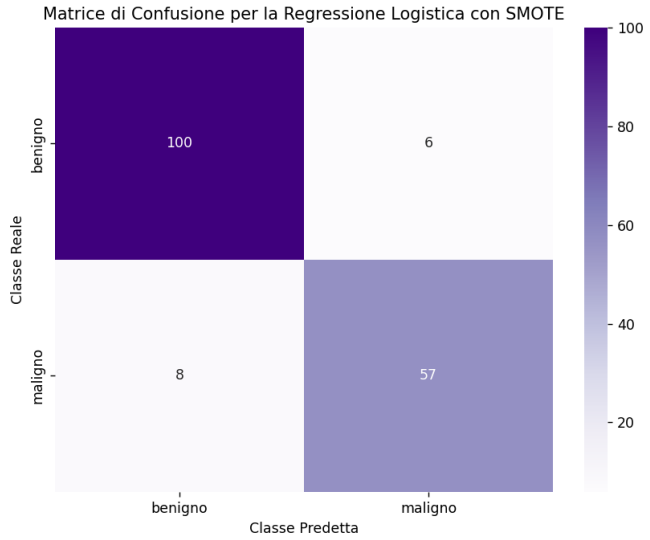
Infine, per eseguire un'analisi diversa, abbiamo optato per l'utilizzo della regressione logistica con SMOTE.
Nel dataset utilizzato, le classi risultano un po' squilibrate (la classe B è più presente della classe M). La combinazione di SMOTE e regressione logistica mira a migliorare le prestazioni del modello.

Risultati con SMOTE:
L'accuratezza complessiva del modello è del 91,8%.

Matrice di Confusione:
Il modello ha fatto bene nel classificare la maggior parte dei casi di "benigno", ma ha commesso 8 errori nel non riconoscere il cancro in pazienti che lo avevano.

Classification Report per la Regressione Logistica con SMOTE:

	precision	recall	f1-score	support
benigno	0.93	0.94	0.93	106
maligno	0.90	0.88	0.89	65
accuracy			0.92	171
macro avg	0.92	0.91	0.91	171
weighted avg	0.92	0.92	0.92	171

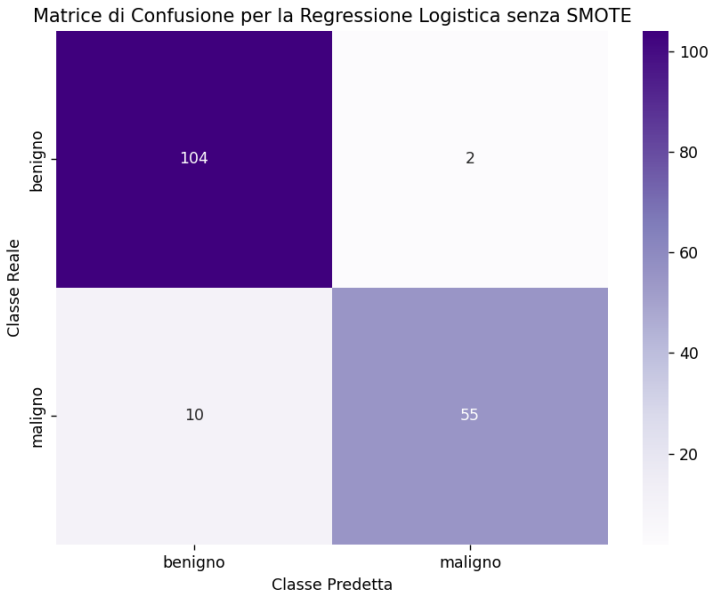


Risultati senza SMOTE:
L'accuratezza del modello senza SMOTE è del 93%. Questo valore è leggermente superiore rispetto a quello ottenuto con SMOTE.

Matrice di Confusione:
Il modello ha erroneamente classificato 2 esempi benigni come maligni (falsi positivi). Ha erroneamente classificato 10 esempi maligni come benigni (falsi negativi).

Classification Report per la Regressione Logistica senza SMOTE:

	precision	recall	f1-score	support
benigno	0.91	0.98	0.95	106
maligno	0.96	0.85	0.90	65
accuracy			0.93	171
macro avg	0.94	0.91	0.92	171
weighted avg	0.93	0.93	0.93	171



~RISULTATI

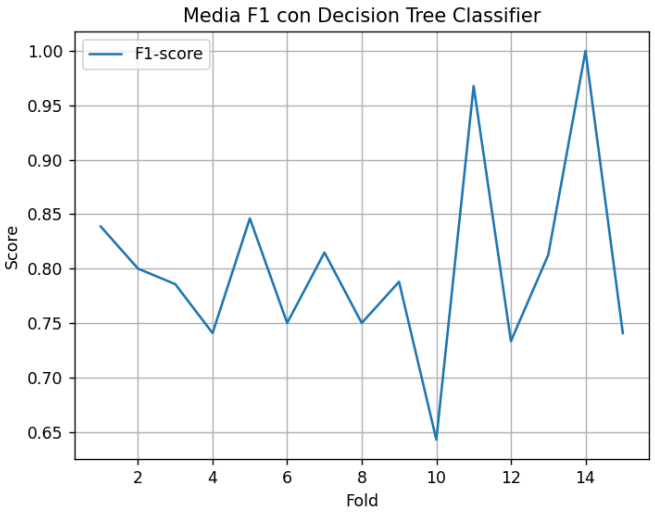
Successivamente per effettuare una valutazione più approfondita sulle prestazioni dei modelli è stata applicata una stratified k-fold cross validation per tre volte, ognuna con diversi valori di k.

STRATIFIED K-FOLD CROSS VALIDATION: Tecnica molto simile alla k-fold cross validation, ma con in più un proporzionamento dei vari esempi per classe target nei fold che si generano evitando di creare sbilanciamenti sulle istanze prese in esame. Questo approccio garantisce che ogni fold sia rappresentativo della distribuzione delle classi nel dataset originale.
Nel nostro caso, abbiamo eseguito la Stratified K-Fold Cross Validation tre volte, variando il valore di **k** in ciascuna esecuzione per confrontare l'effetto di differenti suddivisioni del dataset:

- Prima validazione con **k=5**
- Seconda validazione con **k=10**
- Terza validazione con **k=15**

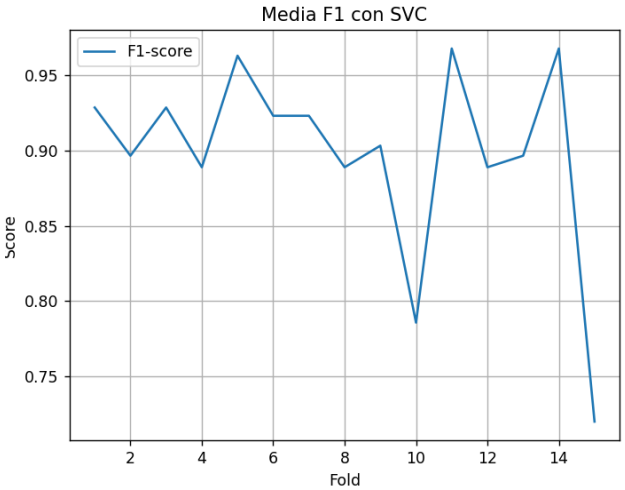
Per ogni suddivisione, abbiamo calcolato le metriche di performance, tra cui F1-score e accuracy, e osservato il comportamento del modello tramite grafici che mostravano le variazioni nelle prestazioni.
Dopo aver confrontato i risultati, abbiamo scelto di adottare **Stratified K-Fold con k=15**. Questa scelta è stata motivata dal fatto che, sebbene i valori di **F1-score** e **accuracy** fossero già buoni per k=5 e k=10 (entrambi vicini a 0.85), con **k=15** il modello ha ottenuto risultati leggermente superiori, con performance più stabili e valori finali più elevati rispetto alle altre suddivisioni.

➔ **Decision Tree Classifier:**



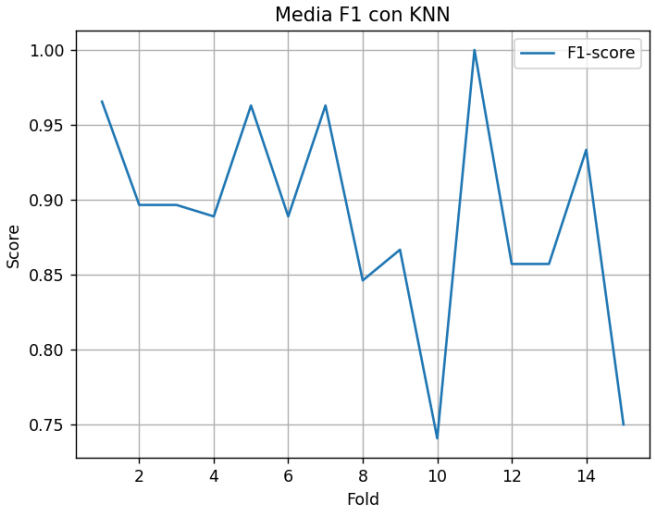

```
Valutazioni del Decision Tree Classifier, con stratified K-Cross validation pari a 15
Average Accuracy: 0.8522996680891417
Average Precision: 0.8167847566376978
Average Recall: 0.7949206349206348
Average F1-score: 0.8007456870091278
```

➔ **SVM:**



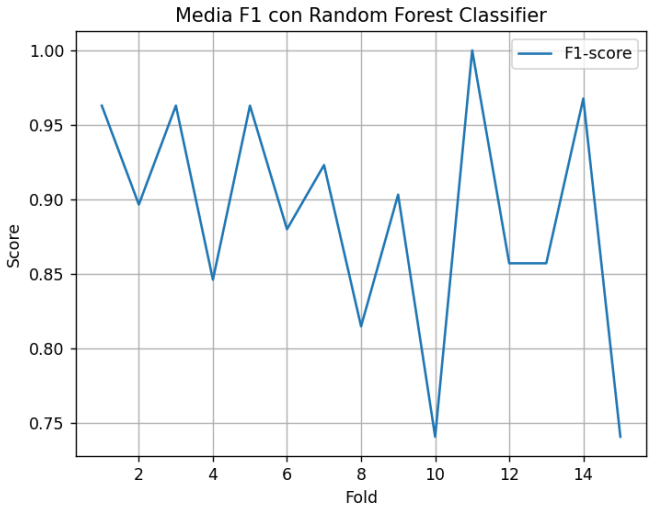

```
+-----------------------------------------------------------------------------------------+
Valutazioni del SVC, con stratified K-Cross validation pari a 15
Average Accuracy: 0.9259838786154572
Average Precision: 0.9275246975246977
Average Recall: 0.8746031746031746
Average F1-score: 0.8980302496223891
+-----------------------------------------------------------------------------------------+
```

➔ **KNN:**



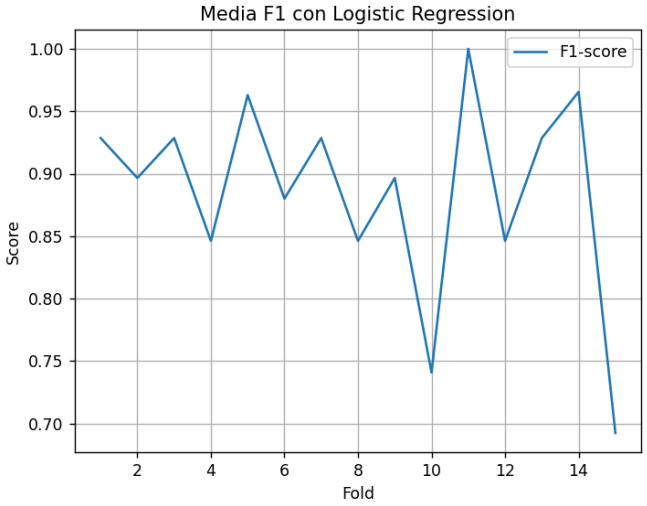

```
+-----------------------------------------------------------------------------------------+
Valutazioni del KNN, con stratified K-Cross validation pari a 15
Average Accuracy: 0.9190137505926982
Average Precision: 0.9205982905982907
Average Recall: 0.8615873015873016
Average F1-score: 0.8875669796359452
+-----------------------------------------------------------------------------------------+
```

➔ **Random Forest Classifier:**



```
+-----------------------------------------------------------------------------------------------+
Valutazioni del Random Forest Classifier, con stratified K-Cross validation pari a 15
Average Accuracy: 0.9154575628259838
Average Precision: 0.925397233485469
Average Recall: 0.8463492063492063
Average F1-score: 0.8816697716859009
+-----------------------------------------------------------------------------------------------+
```

➔ Regressione Logistica




```
Valutazioni della Regressione Logistica con stratified K-Cross validation pari a 15
Average Accuracy: 0.918918918918919
Average Precision: 0.9331746031746032
Average Recall: 0.8473015873015872
Average F1-score: 0.885825289227588
+-----------------------------------------------------------------------------------------------+
```

RETE BAYESIANA

È stata implementata una rete bayesiana per capire, date tutte le caratteristiche (di quelle specificate nel dataset) di un soggetto, quanto questo abbia la probabilità di un tumore benigno o maligno. I nodi rappresentano le variabili, mentre gli archi rappresentano le relazioni di dipendenza statistica tra le variabili e le distribuzioni locali di probabilità dei nodi figlio rispetto ai valori dei nodi genitori.

***Oss**: Durante lo sviluppo del progetto, per visualizzare la rete bayesiana, è stato necessario utilizzare un ambiente virtuale conda. Questo perché l'installazione di Graphviz (libreria utilizzata per visualizzare alberi decisionali e grafici), ha causato conflitti di dipendenze con le versioni predefinite dei pacchetti nel sistema principale. Utilizzando un ambiente Conda, è stato possibile creare un'installazione isolata e gestire le dipendenze in modo più flessibile.*

~PRE-PROCESSAMENTO DEI DATI

Dal momento che l'algoritmo utilizzato richiede che i valori siano discreti (e non continui), è stato necessario trasformare i dati continui in valori discreti. Per semplicità, abbiamo effettuato una conversione diretta dei valori continui in numeri interi, riducendo così il rischio di incorrere in problematiche legate alla gestione di dati continui in un modello che funziona meglio con variabili discrete. Questo approccio ci ha permesso di sfruttare al meglio le capacità della rete bayesiana.

Predisposizione della struttura della rete:

La predisposizione della struttura della rete bayesiana è stata effettuata utilizzando la **local hill climb search** per stimare un grafo aciclico diretto (DAG) che massimizza il punteggio secondo il metodo di scoring scelto. Il processo inizia con un modello di partenza, denominato 'start_dag', e procede apportando modifiche alla rete in maniera iterativa, fino a raggiungere un massimo locale.
In questo caso, il punteggio è calcolato tramite il metodo **K2Score()**, che utilizza la distribuzione di Dirichlet con iper-parametri impostati a 1. Questo punteggio consente di quantificare l'influenza di una variabile sulla propria lista di potenziali genitori, favorendo la costruzione di una rete che rappresenti efficacemente le relazioni tra le variabili.

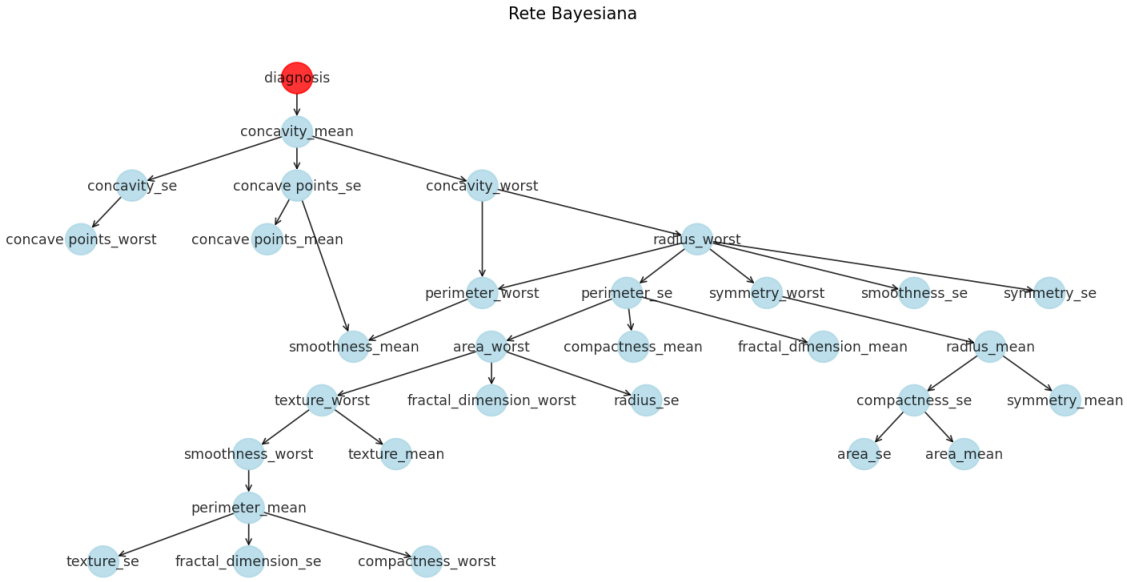
- **Nodi**: diagnosis, concavity_mean ,radius_mean, compactness_se, symmetry_mean, perimeter_mean, texture_se, fractal_dimension_se ,compactness_worst, concavity_se, concave points_se, concavity_worst, perimeter_se ,area_worst ,compactness_mean, fractal_dimension_mean, area_se, area_mean, concave points_worst, concave points_mean, smoothness_mean, radius_worst, symmetry_worst, smoothness_se, symmetry_se, perimeter_worst, texture_worst, smoothness_worst, texture_mean, fractal_dimension_worst, radius_se

- **Archi**: ('diagnosis', 'concavity_mean'), ('concavity_mean', 'concavity_se'), ('concavity_mean', 'concave points_se'), ('concavity_mean', 'concavity_worst'), ('radius_mean', 'compactness_se'), ('radius_mean', 'symmetry_mean'), ('compactness_se', 'area_se'), ('compactness_se', 'area_mean'), ('perimeter_mean', 'texture_se'), ('perimeter_mean', 'fractal_dimension_se'), ('perimeter_mean', 'compactness_worst'), ('concavity_se', 'concave points_worst'), ('concave points_se', 'concave points_mean'), ('concave points_se', 'smoothness_mean'), ('concavity_worst', 'perimeter_worst'), ('concavity_worst', 'radius_worst'), ('perimeter_se', 'area_worst'), ('perimeter_se', 'compactness_mean'), ('perimeter_se', 'fractal_dimension_mean'), ('area_worst', 'fractal_dimension_worst'), ('area_worst', 'texture_worst'), ('area_worst', 'radius_se'), ('radius_worst', 'symmetry_worst'), ('radius_worst', 'perimeter_se'), ('radius_worst', 'smoothness_se'), ('radius_worst', 'symmetry_se'), ('radius_worst', 'perimeter_worst'), ('symmetry_worst', 'radius_mean'), ('perimeter_worst', 'smoothness_mean'), ('texture_worst', 'smoothness_worst'), ('texture_worst', 'texture_mean'), ('smoothness_worst', 'perimeter_mean')

Creazione della rete:

La rete è stata prima creata con **BayesianNetwork()**, dando come argomento gli archi del modello stimato precedentemente e poi addestrata. Lo stimatore utilizzato è il **Bayesian Estimator**. Il BE combina la verosimiglianza dei dati con una distribuzione a priori sui parametri per ottenere una distribuzione a posteriori dei parametri. Invece di fornire un singolo valore stimato per i parametri, il BE fornisce una distribuzione di probabilità sui possibili valori dei parametri. L'implementazione della rete bayesiana tramite il Bayesian Estimator ci consente di ottenere una rappresentazione più ricca e informata delle relazioni tra le variabili, favorendo l'analisi di dati complessi come quelli utilizzati nel contesto della diagnosi di tumori.
Questa distribuzione tiene conto sia dei dati osservati che delle conoscenze a priori sui parametri. L'uso di una distribuzione a priori permette di incorporare conoscenze o ipotesi precedenti sulle CPD dei nodi.

Rappresentazione grafica della rete:




```
### Valutazione Rete Bayesiana ###
Accuracy: 0.9123
Precision: 0.9001
Recall: 0.9050
F1 Score: 0.9025
```

Si è calcolata, infine, la probabilità che una certa persona, con determinati valori, abbia un tumore al seno benigno o maligno.

Caso maligno:

```
Probabilità per una donna di avere un tumore maligno al seno:
+-------------+------------------+
| diagnosis   |   phi(diagnosis) |
+=============+==================+
| diagnosis(0) |           0.0076 |
+-------------+------------------+
| diagnosis(1) |           0.9924 |
+-------------+------------------+
```

Caso benigno:

```
Probabilità per una donna di avere un tumore benigno al seno:
+--------------+------------------+
| diagnosis    |   phi(diagnosis) |
+==============+==================+
| diagnosis(0) |           0.9976 |
+--------------+------------------+
| diagnosis(1) |           0.0024 |
+--------------+------------------+
```