CASO DI STUDIO ICON 2023

***Breast Cancer***

Prof. Nicola Fanizzi

AUTORI

Corsini Carmine [738708], c.corsini2@studenti.uniba.it

Del Core Angelo [737750], a.delcore6@studenti.uniba.it

Lopedota Francesco [735971], f.lopedota1@studenti.uniba.it

# INDICE

1. [Introduzione](#_Toc39047413) 3

2. Organizzazione del Dataset 3

3. Analisi e pre-elaborazione dei dati 4

4. Osservazione grafica dei dati 5

5. Apprendimento non supervisionato 8

6. Apprendimento supervisionato 13

7. Rete Bayesiana 22

**INTRODUZIONE**

~INFORMAZIONI TEORICHE SUL PROGETTO:

Il nostro progetto mira a fornire un supporto medico completo per la diagnosi preventiva del tumore al seno. Inoltre, desideriamo aiutare gli specialisti a distinguere se i sintomi presentati dai pazienti sono correlati a disturbi cronici o a condizioni diverse. Inoltre, stiamo cercando di identificare le caratteristiche comuni tra i pazienti affetti da tali disturbi per comprendere meglio le somiglianze tra di loro. Attraverso l'analisi dei dati, stiamo cercando di individuare gruppi distinti di persone all'interno del campione, che possano reagire in modo simile alla somministrazione di determinati farmaci. Questo ci aiuta a personalizzare i trattamenti in base alle caratteristiche specifiche di ciascun gruppo. Nel corso del progetto, abbiamo utilizzato sia algoritmi di apprendimento supervisionato che di apprendimento non supervisionato. Questi algoritmi ci hanno permesso di identificare modelli di dati e di fare previsioni accurate sulla base delle informazioni disponibili.

* ***In grassetto e corsivo verranno riportate le nostre valutazioni, su ciascuna delle nostre scelte progettuali.***

~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.

**ORGANIZZAZIONE DEL DATASET**

***Link:*** [Breast Cancer Dataset | Kaggle](https://www.kaggle.com/datasets/yasserh/breast-cancer-dataset/code?datasetId=1829286&sortBy=voteCount)

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamenteLl cancro al seno è comune tra le donne, con il 25% dei casi di cancro. La sfida principale è classificare i tumori in maligni o benigni. Questo aiuta nella diagnosi precoce e nella scelta del trattamento. Qui di seguito riportiamo delle informazioni sulle righe e colonne del Dataset. Come si puo’ vedere ci sono 569 righe e 32 colonne.

**ANALISI E PRE-ELABORAZIONE DEI DATI**

Per prima cosa abbiamo eliminato la colonna ID, inutile per il fine della nostra analisi e abbiamo distinto le colonne con valori continui e colonne con valori discreti:

Immagine che contiene testo, schermata

Descrizione generata automaticamente

Successivamente abbiamo verificato il numero di valori NULL per ogni colonna e abbiamo utilizzato un grafico a torta per capire come fosse distribuito il nostro dataset in relazione alla nostra variabile target “diagnosis”, con due valori: B che indica un tumore Benigno, M un tumore Maligno.

Immagine che contiene testo, schermata, cerchio, diagramma

Descrizione generata automaticamenteImmagine che contiene testo, menu, schermata, Carattere

Descrizione generata automaticamente

**OSSERVAZIONE GRAFICA DEI DATI**

Per comprendere al meglio i dati, abbiamo svolto delle osservazioni grafiche che ci permettessero di valutare la correlazione tra i dati nel Dataset e soprattutto in che modo la diagnosi fosse influenzata da essi. Qui di seguito riportiamo dei grafici che abbiamo realizzato:

**Immagine che contiene testo, diagramma, schermata, Diagramma

Descrizione generata automaticamente**

Da questo grafico si evince che un perimetro che va da 50 a 75, mediamente è riconducibile a un tumore benigno, al contrario, allontandoci da questo range, il tumore diventa mediamente malgino.

**Immagine che contiene testo, schermata, diagramma, Diagramma

Descrizione generata automaticamente**

In questo grafico invece, abbiamo relazionato l’aria media del tumore con la diagnosi ed è risultato che ad un area media di 500 è molto probabile che il tumore sia benigno. La distribuzione invece dei tumori maligni incomincia da un area di 500 in poi.

**Immagine che contiene testo, diagramma, schermata, Diagramma

Descrizione generata automaticamente**

L’ultima relazione che abbiamo deciso di analizzare è quella della concavità media del tumore. Come sempre è risultato che a valori minori corrisponde un tumore benigno, a uno maggiore un tumore maligno.

**Immagine che contiene schermata, modello, punto

Descrizione generata automaticamente**Per finire la nostra analisi grafica, abbiamo creato un Heatmap che ci permettesse di inquadrare la relazione tra i dati in maniera completa e chiara. Riportiamo lo screen:

Prima di passare alle tecniche di apprendimento non supervisionato e supervisionato, abbiamo creato una versione del Dataset standardizzata. Per prima cosa abbiamo estrapolato le colonne con valori continui e dopo abbiamo utilizzato la funzione Standard Scaler per poterli correttamente standardizzare. Di seguito, un breve cenno teorico.

***StandardScaler ( )*** - il nuovo valore standard z di un campione x è calcolato come:

***z = (x - u) / s***

dove u è la media dei campioni di addestramento ed s è la deviazione standard dei campioni di addestramento.

**APPRENDIMENTO NON SUPERVISIONATO**

~CLUSTERING:

Il clustering, noto anche come classificazione non supervisionata, è un approccio analitico che raggruppa elementi di un dataset in classi omogenee. A differenza della classificazione, non assegna etichette di classe o categorie predefinite. Queste classi, chiamate cluster, sono gruppi di oggetti che presentano somiglianze tra di loro, ma con caratteristiche diverse rispetto agli oggetti in altri cluster. L'obiettivo è esplorare la distribuzione delle persone all'interno del dataset senza influenze esterne o preconcetti predefiniti.

~PRINCIPAL COMPONENT ANALYSIS:

Inizialmente, abbiamo applicato la PCA ai dati standardizzati. La Principal Component Analysis è una tecnica che semplifica i dati attraverso una combinazione lineare dei coefficienti ottenuti dagli autovettori della matrice di correlazione delle variabili originali. Le componenti principali risultanti sono non correlate e catturano la massima varianza dei dati. Utilizzando la PCA, è possibile ridurre la dimensionalità dei dati mantenendo una percentuale elevata di informazione.

Abbiamo scelto di utilizzare la PCA data l'alta dimensionalità del dataset, la scarsa informatività delle feature e per ridurre il rumore presente nei dati. Abbiamo optato per l'utilizzo del dataset standardizzato in quanto i test e le metriche hanno mostrato che questi dati supportavano meglio l'algoritmo di clustering impiegato.

Per determinare il numero di componenti principali da includere, abbiamo utilizzato due tecniche:

* la Regola di Kaiser: secondo questa regola, si includono nel modello finale tutte le componenti con un autovalore maggiore o uguale a 1.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Risultano quindi 6 valori da comprendere nella nostra PCA.

* Inoltre, per verificare i dati e avere un confronto è stato utilizzato il metodo dello Scree Plot. Questo metodo prevede la creazione di un grafico in cui gli autovalori sono rappresentati sull'asse verticale e il numero di componenti potenziali da estrarre è rappresentato sull'asse orizzontale. La linea risultante presenta una curva che può essere concava o convessa in diverse parti. Secondo questo approccio, il numero di componenti da selezionare corrisponde al punto in cui si osserva un cambio significativo nella pendenza della curva.

Immagine che contiene testo, schermata, linea, diagramma

Descrizione generata automaticamente

* ***I risultati si sono dimostrati ovviamente coincidenti e quindi abbiamo scelto di prendere come principali componenti le prime 6.***

~KMEDOIDS:

Abbiamo impiegato l'algoritmo K-medoids utilizzando la funzione KMedoids() per il clustering. Questo metodo rappresenta un'alternativa al K-means, in quanto si basa sulla minimizzazione della somma delle differenze tra i punti appartenenti a un cluster e un punto designato come medoide di quel cluster. A differenza del K-means, il K-medoids può essere utilizzato per dati categorici. In particolare, abbiamo utilizzato l'algoritmo Partitioning Around Medoids (PAM) come metodo specifico per il K-medoids. Questo algoritmo si sviluppa in diverse 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 finché non vi è alcuna variazione nelle assegnazioni.

Immagine che contiene linea, diagramma, Diagramma, testo

Descrizione generata automaticamentePer determinare il numero di cluster (k), abbiamo utilizzato metodi come la curva del gomito. Questo metodo cerca il punto in cui un certo score, dopo un certo valore di k, non mostra una variazione significativa. Ad esempio, abbiamo utilizzato l'inertia\_, che rappresenta la somma delle distanze quadratiche dei dati rispetto al loro cluster più vicino in ogni assegnazione.

Immagine che contiene linea, Diagramma, diagramma, Parallelo

Descrizione generata automaticamente

Si evince dalla curva, che il gomito è a 2. Prima di continuare, pero’, abbiamo deciso di verificare questo, utilizzando un altro metodo, utilizzando il coefficiente di Silhouette.

~COEFFICIENTE DI SILHOUETTE:

Abbiamo utilizzato il coefficiente di silhouette come metrica per valutare la qualità dei risultati ottenuti dal clustering. Questo coefficiente varia da -1 a 1 e fornisce un'indicazione della distinzione dei cluster:

* Un punteggio di 1 indica che i cluster sono ben separati e chiaramente distinti.
* Un punteggio di 0 indica che i cluster sono sovrapposti o che la distanza tra i cluster non è significativa.

Un punteggio di -1 indica che i punti sono stati assegnati in modo errato ai cluster.

Il coefficiente di silhouette è calcolato come segue:

***silhouette\_score = (p - q) / max(p, q),*** dove:

* p è la distanza media dei punti nel cluster più vicino (a cui il punto non appartiene)
* q è la distanza media intra-cluster da tutti i punti nel proprio cluster.

Un valore più alto del coefficiente di silhouette indica una migliore separazione dei cluster.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

* ***Poiché sia il Silhouette score che la regola del gomito danno come risultato ottimale 2, è stato deciso di addestrare l’algoritmo partizionando il dataset in 2 cluster.***

Di seguito riportiamo il clustering eseguito con KMedoids:

Immagine che contiene schermata, diagramma

Descrizione generata automaticamente

Metriche di valutazione del clustering:

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

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), di seguito riportiamo dei cenni teorici:

* L'omogeneità misura se i cluster contengono solo campioni appartenenti a una singola classe. È calcolata come la differenza tra l'entropia condizionale delle etichette vere dato l'assegnamento dei cluster e l'entropia delle etichette vere. Il valore dell'omogeneità è compreso tra 0 e 1, dove un valore più vicino a zero indica un cluster meno omogeneo.
* ***Un valore di omogeneità di 0.8 indica che i campioni all'interno di ciascun cluster sono molto simili tra loro dal punto di vista delle etichette. Pertanto, i cluster creati dall'algoritmo k-medoids sembrano essere coerenti in termini di etichettatura.***
* La completezza verifica se tutti i punti dati appartenenti a una determinata classe sono raggruppati nello stesso cluster. È calcolata come la differenza tra l'entropia condizionale delle etichette previste dato l'assegnamento dei cluster e l'entropia delle etichette previste. Anche il valore della completezza è compreso tra 0 e 1, dove un valore più vicino a zero indica una completezza inferiore.
* ***Un valore di completezza di 0.18 indica che l'algoritmo ha recuperato solo una piccola parte dei campioni appartenenti alla stessa categoria. Quindi, i cluster creati potrebbero non essere completi in termini di coprire tutti i campioni corrispondenti a una particolare categoria.***
* La misura V, che rappresenta la media armonica tra omogeneità e completezza, è calcolata con un parametro β pari a 1. Questa misura fornisce un'indicazione complessiva della qualità del clustering, considerando sia l'omogeneità che la completezza. Un valore più alto della misura V indica una migliore combinazione di omogeneità e completezza.
* ***Un valore di V\_measure di 0.29 indica che i cluster creati hanno una coerenza moderata e un recupero parziale dei campioni corrispondenti alla stessa categoria.***

**APPRENDIMENTO SUPERVISIONATO**

L'apprendimento supervisionato è una branca dell'intelligenza artificiale che comprende la classificazione come una delle sue tecniche. In questa modalità di apprendimento, gli algoritmi operano su dataset che contengono etichette o target predefiniti. L'obiettivo è quello di predire o classificare nuovi dati utilizzando le informazioni apprese dai dati di addestramento. Gli algoritmi di classificazione lavorano su dataset etichettati e producono output che possono essere variabili qualitative, come binarie, nominali o ordinate. Questi algoritmi sfruttano i dati di addestramento per costruire un modello che possa generalizzare e fare previsioni accurate sui dati non visti in precedenza. La fase di testing, che riguarda la valutazione delle prestazioni del modello su dati separati dal training, sarà approfondita successivamente nella descrizione degli algoritmi specifici.

~KNEIGHBOURS CLASSIFIER (DA QUI IN POI BISOGNA RIGENERARE CON CHAT GPT PERCHE’ HO SPUDORATAMENTE COPIATO E INCOLLATO DA MIA CUGINA)

**KNeighborsClassifier() :** Il k-nearest neighbors parte dal presupposto che punti simili (e quindi oggetti) possono essere trovati l’uno vicino all’altro. ‘k’ indica appunto il numero di punti più vicini a quello da classificare, ed è un intero positivo.

In questo algoritmo, lo spazio viene partizionato in regioni in base alle posizioni e alle caratteristiche degli oggetti di apprendimento. Per calcolare la distanza tra essi, si è usata la distanza euclidea (=default metrica di minkowski con p=2). Infine, l’oggetto è assegnato alla classe che è la più frequente fra i suoi k vicini.

È stato utilizzato il dataset standardizzato, poiché più indicato per gli algoritmi distance-based e inoltre, per i test condotti, è quello che porta l’algoritmo a risultati migliori.

Per prima cosa bisogna scegliere quanto vale k. Per decidere ciò, l’algoritmo è stato addestrato a partire da un vicino, fino ad arrivare a venti. Ad ogni ciclo, è stata applicata la Cross Validation, essa consiste nel sezionare il set di esempi più volte, con porzioni di train e test sempre diverse, calcolandone per ognuna lo score. Nel nostro caso viene utilizzata come metrica l’accuracy. Con la cross-validation si evita l’overfitting e si ha una visione più completa del funzionamento dell’algoritmo. Il risultato finale è dato dalla media dei punteggi di tutte le combinazioni.

Per scegliere il numero di vicini ideale è stata scelta una 5-fold cross validation e applicata al variare di k. Quando l’algoritmo ha raggiunto la F1 maggiore, è stato preso il corrispondente numero di vicini.

* ***Abbiamo scelto una cross validation di 5 perché è il giusto compromesso tra tempo di computazione, valutazione e dimensione del dataset. Diminuendo codesto numero, il risultato della metrica è troppo ridotta, aumentandolo invece si rarefaziona troppo.***

Immagine che contiene diagramma, linea, Diagramma, origami

Descrizione generata automaticamente

Stabilito k=3, il modello è stato addestrato.

* ***Abbiamo scelto k=3 perché è il primo picco della curva.***

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

~DECISION TREE CLASSIFIER

***DecisionTreeClassifier()*** : L’obiettivo di questo algoritmo è dividere la popolazione iniziale per il valore di una variabile che permette di creare due gruppi che sono il più omogenei possibile internamente e il più disomogenei possibile tra loro. Può lavorare su dati numerici e categorici contemporaneamente. Durante l’esecuzione l’algoritmo divide continuamente i dati di input in base a determinate metriche (Default gini - L'impurità Gini rappresenta la probabilità di classificare in modo errato un punto dati casuale nel set). Lo scopo è quello di trovare la migliore divisione.

Nell’albero, i nodi sono i luoghi in cui i dati vengono splittati e le foglie i risultati intermedi o finali. È stato utilizzato il dataset standardizzato, poiché non vincolato da misure di distanza, per la migliore gestione degli outliers e inoltre è risultato, dai test eseguiti, quello in grado di portare l’algoritmo ad una performance migliore.

Per questo e per i successivi algoritmi impiegati, al fine di selezionare i migliori iper-parametri, si è optato di usare RandomizedSearchCV. Quest’ultima è una funzione che dato un insieme di parametri, ne sceglie un sottoinsieme e testa il modello preso in considerazione al variare di questi, mediante una 5-fold cross validation e lo valuta con la F1-measure.

Così facendo si possono adottare i parametri che portano allo score migliore. Tra: parametri = {'criterion': ['gini', 'entropy', 'log\_loss']} 'entropy' è il criterio che ha portato a risultati migliori:

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

* ***Abbiamo scelto il criterio Entropy, perché è il criterio che ci permette di avere delle valutazioni di precision e recall decisamente piu’ alte rispetto a Gini e log\_loss.***

~RANDOM FOREST CLASSIFIER

***RandomForestClassifier()*** :Il metodo che usa questo algoritmo fa parte delle tecniche che utilizzano un gruppo di modelli imprecisi, come gli alberi di decisione, al fine di crearne uno più preciso. RandomForestClassifier() quindi, costruisce un gran numero di alberi decisionali individuali, su diversi sottoinsiemi del dataset originale, che operano come un insieme. Ogni singolo albero nel Random Forest fa una previsione di classe e la classe che è stata predetta maggiormente diventa il risultato che dà il modello. È stato utilizzato il dataset standardizzato, poiché non vincolato da misure di distanza, per la migliore gestione degli outliers e inoltre è risultato, dai test eseguiti, quello in grado di portare l’algoritmo ad una performance migliore.

Tramite RandomizedSearchCV la scelta tra i parametri = {'n\_estimators': [25, 50, 75, 100, 150, 200, 250]}, è risultata 25.

* ***Il codice eseguito ci permetteva di stampare il numero di estimators migliore e tra i 7 che avevamo selezionato per l’iterazione, è risultato il 25.***

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

~SVM

***SVC()*** : Traccia ogni dato come un punto nello spazio n-dimensionale (dove n è un numero di caratteristiche disponibili) con il valore di ciascuna caratteristica che è il valore di una particolare coordinata. Successivamente esegue la classificazione trovando l'iperpiano, ovvero la superficie di decisione ottimale che divide positivi da negativi in uno spazio di embedding, utilizzando vettori di supporto.

Rispetto agli algoritmi presenta due vantaggi principali: maggiore velocità e migliori prestazioni con un numero limitato di campioni (nell'ordine delle migliaia).

SVM può essere di due tipi:

* SVM lineare - quello appena descritto. Viene utilizzato per dati separabili linearmente, il che significa che se un set di dati può essere classificato in due classi utilizzando una singola linea retta. Tali dati vengono definiti come dati separabili linearmente e il classificatore che viene utilizzato è chiamato classificatore SVM lineare.
* SVM non lineare - viene utilizzato per dati separabili non linearmente, ovvero il set di dati non può essere classificato utilizzando una linea retta. Tali dati vengono definiti come dati non lineari e il classificatore utilizzato viene chiamato Classificatore SVM non lineare.

È stato utilizzato il dataset normalizzato, poiché più indicato per gli algoritmi distance-based e inoltre, per i test condotti, è quello che porta l’algoritmo a risultati migliori.

Tramite *RandomizedSearchCV* la scelta tra i parametri = parametri = {'C': [0.001, 0.01, 0.1, 1, 10, 100], 'gamma': [0.001, 0.01, 0.1, 1, 10, 100]}, è risultata C=1, gamma=10.

* ***Abbiamo scelto il valore C=1 e gamma=10, perché dopo svariati cicli di iterazione con i parametri immessi da noi in input, questi due valori ci hanno portato a dei valori di precision e recall molto buoni.***

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

~RISULTATI

Successivamente per effettuare una valutazione più approfondita 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.

Nel nostro caso è stata eseguita una stratified k-fold cross validation per tre volte, in maiera tale da provare diverse suddivisioni del dataset variando anche k, infatti la prima validazione è stata fatta con k=5, la seconda con k=10, la terza con k=15.

Per ognuna di esse sono state calcolate le metriche nominate in precedenza e si è osservato mediante grafici il variare delle prestazioni per ogni test.

* ***Abbiamo scelto una Stratified K-Fold Cross Validation di 15 perché con le run con 5,10 i valori di F1 e accuracy seppur essendo sempre vicini al 0.85, risultavano comunque inferiori a quelli riportati di seguito.***

**Immagine che contiene diagramma, linea, Diagramma, testo

Descrizione generata automaticamenteDecision Tree Classifier:**

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

**SVC:**

Immagine che contiene testo, linea, Diagramma, diagramma

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

**KNN:**

Immagine che contiene testo, linea, Diagramma, diagramma

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

**Random Forest Classifier:**

**Immagine che contiene testo, linea, diagramma, Diagramma

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

**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.

Una rete bayesiana è un grafico aciclico diretto, dove:

• i nodi, rappresentano le variabili,

• 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.

PRE-PROCESSAMENTO DEI DATI

In questo caso è stato necessario trasformare tutti i valori continui nel dataframe in valori discreti, per evitare di incorrere in eventuali problemi: infatti l’algoritmo utilizzato supporta pienamente solo questo tipo di valori. Ciò è stato fatto, per semplicità, convertendoli in interi.

Predisposizione della struttura della rete:

Attraverso la local hill climb search si è stimata la struttura DAG che ha un punteggio ottimale, secondo il metodo di scoring fornito. Inizia con il modello start\_dag e procede con le modifiche della rete passo dopo passo fino al raggiungimento di un massimo locale. In questo caso specifico si è calcolato il punteggio, che misura quanto una data variabile è "influenzata" da una data lista di potenziali genitori, attraverso K2Score(), metodo che utilizza la distribuzione di Dirichlet con iper parametri impostati ad 1.

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. 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:

Immagine che contiene diagramma, linea, origami

Descrizione generata automaticamente

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

Caso maligno:

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Caso benigno:

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente