Università degli Studi di Milano Bicocca

DIPARTIMENTO DI INFORMATICA

CORSO DI STUDIO MAGISTRALE IN INFORMATICA

Relazione di Machine Learning
Breast Cancer

Componenti:
Nicolò Fiorentini 933157
Anoir Kadmiri 933097

Anno accademico:
2024/2025

Sommario

Chapter 1

Introduzione

1.1 Componenti del gruppo

Il team è composto dalle seguenti persone:

- Fiorentini Nicolò 933157
- Anoir Kadmiri 933097

1.2 Descrizione del progetto

In questo progetto analizziamo un dataset applicando tecniche di machine learning per valutarne la predittività. L'obiettivo è confrontare le prestazioni di diversi modelli dopo un'adeguata fase di pre-elaborazione.

Inizialmente selezioniamo un dataset e ne eseguiamo un'analisi esplorativa per individuare relazioni tra variabili e target. Durante questa fase, ci occupiamo della pulizia dei dati: conversione dei formati, gestione dei valori nulli, rimozione di elementi irrilevanti e normalizzazione delle variabili.

Successivamente standardizziamo le caratteristiche e applichiamo la PCA per la riduzione della dimensionalità. Infine, addestriamo diversi modelli sia sui dati trasformati che su quelli originali, confrontandone le prestazioni.

Infine, valuteremo le loro prestazioni utilizzando metriche di accuratezza e altre misure pertinenti, traendo delle conclusioni dall'analisi dei risultati ottenuti.

1.3 Selezione del dataset

Per questa progetto, dopo un'attenta analisi e ricerca, è stato adottato il dataset *Breast Cancer Data Set*, reperibile su Kaggle al seguente indirizzo: `https://www.kaggle.com/datasets/erdemtaha/cancer-data/data`.
Questo dataset è stato scelto in quanto fornisce dettagliati indicatori di salute legati al diabete, consentendo di esaminare le connessioni tra le diverse variabili e la presenza della malattia.

Il dataset è costituito da 569 campioni, ciascuno dei quali rappresenta un tumore di tipo benigno o maligno. Ogni campione è caratterizzato da 32 attributi, suddivisi in tre categorie principali:

1. **Caratteristiche generali**: includono l'ID del paziente, lo spessore del tumore e le dimensioni delle cellule.
2. **Caratteristiche derivate**: comprendono 10 attributi che descrivono proprietà del nucleo cellulare, come raggio, texture e perimetro.
3. **Caratteristiche generali**: sono 13 attributi che catturano i valori più elevati delle misurazioni del nucleo cellulare.

L'ultima colonna del dataset rappresenta la classificazione del tumore, che può essere benigno (B) o maligno (M). Questo dataset costituisce un'ottima base per l'applicazione di tecniche di apprendimento automatico e analisi dei dati, con l'obiettivo di sviluppare modelli in grado di distinguere con precisione tra tumori benigni e maligni in base alle loro caratteristiche cliniche.

1.4 Obiettivi previsti

L'obiettivo di questo progetto, è quello di sviluppare un modello in grado di determinare la natura del tumore al seno, utilizzando due classi di classificazione:

- **1 = Maligno** - Diagnosi positiva (canceroso)
- **0 = Benigno** - Diagnosi negativa (assente)

L'obiettivo è classificare correttamente la natura del tumore al seno, distinguendo tra benigno e maligno.

Chapter 2

Analisi esplorativa del Dataset

La prima fase è stata dedicata alla comprensione del significato delle colonne e alla valutazione dei tipi di dati con cui avremmo dovuto operare. In seguito, si è passati a un'analisi esplorativa del dataset che ci ha permesso di esaminare la distribuzione delle singole variabili e la correlazione che tali variabili hanno con la variabile target. È stata inoltre valutata la qualità delle informazioni contenute nell'insieme di dati, verificando la presenza di eventuali:

- Valori mancanti
- Sbilanciamento del dataset
- Valori nulli

2.1 Componenti del dataset

Di seguito è riportato l'elenco completo degli attributi presenti nel dataset, accompagnato dalla spiegazione del loro significato.

1. **id**: Codice numerico che identifica il paziente
2. **diagnosis**: Indica se il soggetto è affetto da una condizione maligna o benigna.
3. **radius mean**: La media dei raggi delle cellule tumorali.
4. **texture mean**: La media delle variazioni dei livelli di grigio nei pixel delle immagini digitali delle cellule tumorali.
5. **perimeter mean**: La media dei perimetri delle cellule tumorali.
6. **area mean**: L'area media delle cellule tumorali.
7. **smoothness mean**: La media della variazione locale nella lunghezza dei raggi delle cellule tumorali.
8. **compactness mean**: Misura quanto il perimetro della cellula è compatto rispetto a quanto è stretto.
9. **concavity mean**: La media delle concavità (aree cave) delle cellule tumorali.
10. **concave points mean**: La media dei punti concavi delle cellule tumorali.

11. **symmetry mean**: La media della simmetria delle cellule tumorali.
12. **fractal dimension mean**: La media della dimensione frattale, che misura la complessità della forma della cellula tumorale.
13. **radius se**: L'errore standard del raggio delle cellule tumorali.
14. **texture se**: L'errore standard della texture delle cellule tumorali.
15. **perimeter se**: L'errore standard del perimetro delle cellule tumorali.
16. **area se**: L'errore standard dell'area delle cellule tumorali.
17. **smoothness se**: L'errore standard della liscezza delle cellule tumorali.
18. **compactness se**: L'errore standard della compattezza delle cellule tumorali.
19. **concavity se**: L'errore standard della concavità delle cellule tumorali.
20. **concave points se**: L'errore standard dei punti concavi delle cellule tumorali.
21. **symmetry se**: L'errore standard della simmetria delle cellule tumorali.
22. **fractal dimension se**: L'errore standard della dimensione frattale delle cellule tumorali.
23. **radius worst**: Il peggior valore del raggio delle cellule tumorali.
24. **texture worst**: Il peggior valore della texture delle cellule tumorali.
25. **perimeter worst**: Il peggior valore del perimetro delle cellule tumorali.
26. **area worst**: Il peggior valore dell'area delle cellule tumorali.
27. **smoothness worst**: Il peggior valore della liscezza delle cellule tumorali.
28. **compactness worst**: Il peggior valore della compattezza delle cellule tumorali.
29. **concavity worst**: Il peggior valore della concavità delle cellule tumorali.
30. **concave points worst**: Il peggior valore dei punti concavi delle cellule tumorali.
31. **symmetry worst**: Il peggior valore della simmetria delle cellule tumorali.
32. **fractal dimension worst**: Il peggior valore della dimensione frattale delle cellule tumorali.

Queste variabili forniscono un quadro dettagliato dello stato di salute e degli stili di vita dei soggetti, consentendo un'analisi approfondita delle correlazioni con la diagnosi.

2.1.1 Tipi di dati

Dopo aver compreso il significato di ogni colonna del nostro dataset siamo andati a comprendere la natura di ogni elemento e delle caratteristiche numeriche di ognuno. Tramite i metodi "df.describe()" e "df.dtypes".

Il metodo df.describe() fornisce statistiche riassuntive delle variabili numeriche, come media, deviazione standard, valori minimi e massimi e altro, aiutandoci a comprendere la distribuzione dei dati. df.dtypes mostra invece i tipi di dato di ciascuna colonna.

Dal risultato ottenuto eseguendo il codice per visualizzare il tipo dei dati, abbiamo notato che tutti i dati con cui stiamo lavorando sono di tipo float64(numerici), ad eccezione dell'unico dato categorico, il target. Di conseguenza, la prima operazione di manipolazione del dataset sarà la conversione della variabile categorica in una variabile intera (0,1), poiché i modelli di addestramento che utilizzeremo richiedono dati numerici o binari. Successivamente, verificheremo se è necessario manipolare ulteriormente i dati per altri casi.

2.1.2 Gestione elementi

Data la presenza di elementi non rilevanti per il nostro studio e di una colonna 'errore' nel dataset, abbiamo deciso di rimuovere la colonna 'id', in quanto superflua per il nostro progetto di analisi e potenzialmente in grado di alterare i risultati dei modelli. Inoltre abbiamo eliminato la colonna 'Unnamed: 32', poiché contiene solo valori nulli.

Successivamente, abbiamo verificato la presenza di valori mancanti, nulli e valori duplicati nel dataset, non riscontrandone nessuno.

2.1.3 Variabile target

Infine, ci siamo concentrati sulla gestione della colonna 'Diagnosis', che conteneva i valori 'M' (maligno) e 'B' (benigno). Poiché questi valori categorici non possono essere utilizzati direttamente nei modelli di machine learning, abbiamo applicato un encoding, trasformando 'M' in '1' e 'B' in '0', rendendo così la variabile adatta per l'analisi.

Come ultima operazione esplorativa, abbiamo rappresentato graficamente la distribuzione della variabile target, che è la colonna 'Diagnosis'. La variabile target, è la variabile che i modelli di machine learning cercano di predire, ossia l'output atteso basato sulle caratteristiche in ingresso. Il risultato è il seguente:

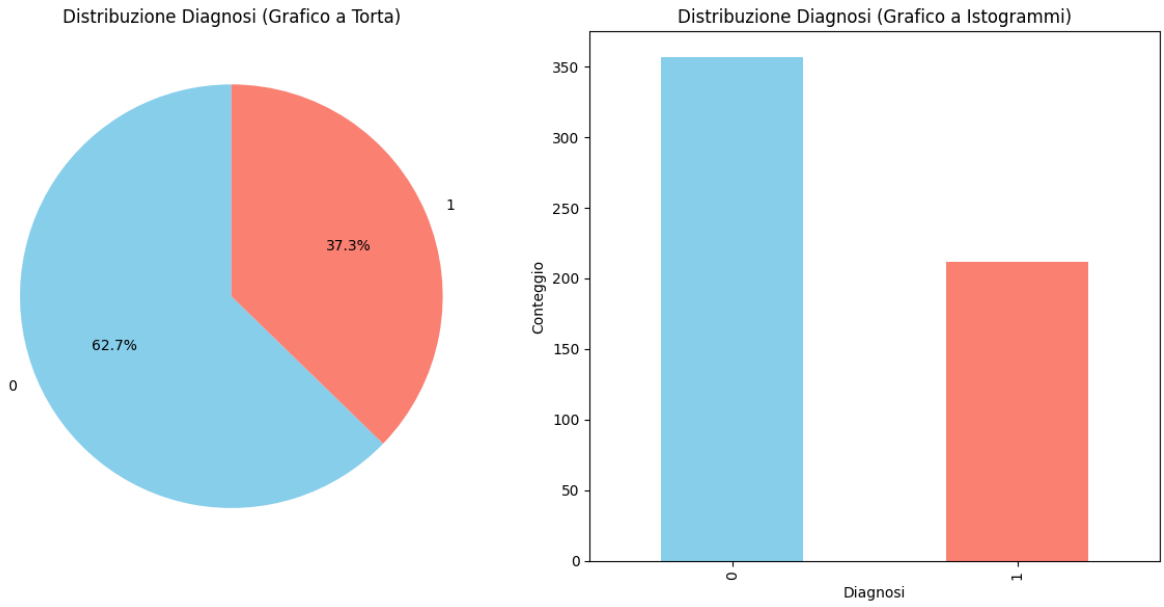


Figure 2.1: Graifico a torta e plot della distribuzione della variabile Diagnosis.

Il dataset risulta sbilanciato: il 62,7% delle istanze è benigno, il 37,3% maligno. Questo squilibrio può influenzare negativamente le prestazioni dei modelli, che tendono a favorire la classe più rappresentata. In questi casi, metriche standard come l'accuratezza risultano poco informative. Per una valutazione più affidabile, adotteremo ulteriori metriche come precision, F1-score e AUC-ROC, più adatte a misurare le prestazioni su entrambe le classi.

Chapter 3

Analisi approfndita e PCA

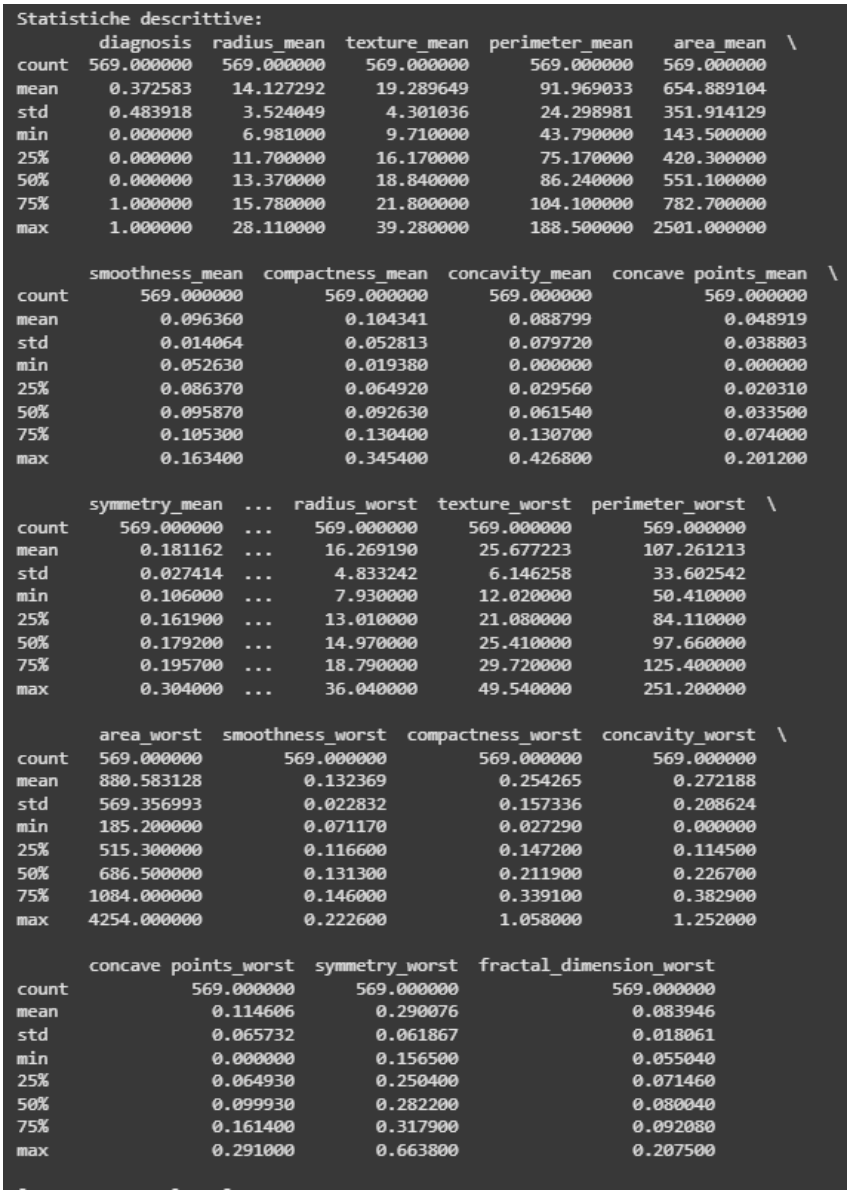
L'analisi univariante e multivariante, sono due approcci fondamentali nell'analisi esplorativa dei dati che ci permettono di comprendere meglio le caratteristiche di un dataset e le relazioni tra le variabili.

L'analisi univariata esamina una variabile alla volta, descrivendone la distribuzione tramite indicatori come media, varianza, mediana e la presenza di outlier. Serve a comprendere il comportamento individuale di ciascuna variabile.

L'analisi multivariata, invece, studia le relazioni tra più variabili simultaneamente, permettendo di individuare correlazioni, interazioni e pattern complessi. È essenziale per capire come più variabili influenzano congiuntamente il target o interagiscono tra loro.

3.1 Uni-Variante

Come prima cosa, tramite la funzione `numeric_data.describe()` abbiamo ottenuto una panoramica statistica delle variabili numeriche del dataset. Questa funzione fornisce informazioni utili come la media, la deviazione standard, i valori minimi e massimi, e i percentili (25%, 50% e 75%) per ogni colonna numerica del dataset. È uno strumento fondamentale per comprendere rapidamente la distribuzione e la variabilità dei dati.



Statistiche descrittive:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
count	569.000000	569.000000	569.000000	569.000000	569.000000	
mean	0.372583	14.127292	19.289649	91.969033	654.889104	
std	0.483918	3.524049	4.301036	24.298981	351.914129	
min	0.000000	6.981000	9.710000	43.790000	143.500000	
25%	0.000000	11.700000	16.170000	75.170000	420.300000	
50%	0.000000	13.370000	18.840000	86.240000	551.100000	
75%	1.000000	15.780000	21.800000	104.100000	782.700000	
max	1.000000	28.110000	39.280000	188.500000	2501.000000	

	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	\
count	569.000000	569.000000	569.000000	569.000000	
mean	0.096360	0.104341	0.088799	0.048919	
std	0.014064	0.052813	0.079720	0.038803	
min	0.052630	0.019380	0.000000	0.000000	
25%	0.086370	0.064920	0.029560	0.020310	
50%	0.095870	0.092630	0.061540	0.033500	
75%	0.105300	0.130400	0.130700	0.074000	
max	0.163400	0.345400	0.426800	0.201200	

	symmetry_mean	...	radius_worst	texture_worst	perimeter_worst	\
count	569.000000	...	569.000000	569.000000	569.000000	
mean	0.181162	...	16.269190	25.677223	107.261213	
std	0.027414	...	4.833242	6.146258	33.602542	
min	0.106000	...	7.930000	12.020000	50.410000	
25%	0.161900	...	13.010000	21.080000	84.110000	
50%	0.179200	...	14.970000	25.410000	97.660000	
75%	0.195700	...	18.790000	29.720000	125.400000	
max	0.304000	...	36.040000	49.540000	251.200000	

	area_worst	smoothness_worst	compactness_worst	concavity_worst	\
count	569.000000	569.000000	569.000000	569.000000	
mean	880.583128	0.132369	0.254265	0.272188	
std	569.356993	0.022832	0.157336	0.208624	
min	185.200000	0.071170	0.027290	0.000000	
25%	515.300000	0.116600	0.147200	0.114500	
50%	686.500000	0.131300	0.211900	0.226700	
75%	1084.000000	0.146000	0.339100	0.382900	
max	4254.000000	0.222600	1.058000	1.252000	

	concave points_worst	symmetry_worst	fractal_dimension_worst
count	569.000000	569.000000	569.000000
mean	0.114606	0.290076	0.083946
std	0.065732	0.061867	0.018061
min	0.000000	0.156500	0.055040
25%	0.064930	0.250400	0.071460
50%	0.099930	0.282200	0.080040
75%	0.161400	0.317900	0.092080
max	0.291000	0.663800	0.207500

Figure 3.1: Statistiche descrittive dei componenti del dataset

Successivamente, abbiamo utilizzato tre tipi di grafici per analizzare la distribuzione delle variabili:

- **Istogrammi**: mostrano la frequenza dei valori suddivisi in intervalli, utili per identificare outlier, tendenza centrale e dispersione.
- **Grafici a densità**: rappresentano la distribuzione con una curva continua, permettendo di analizzarne la forma, le modalità e le code.
- **Box plot**: evidenziano quartili, mediana, estremi e outlier, facilitando il confronto tra variabili o gruppi.

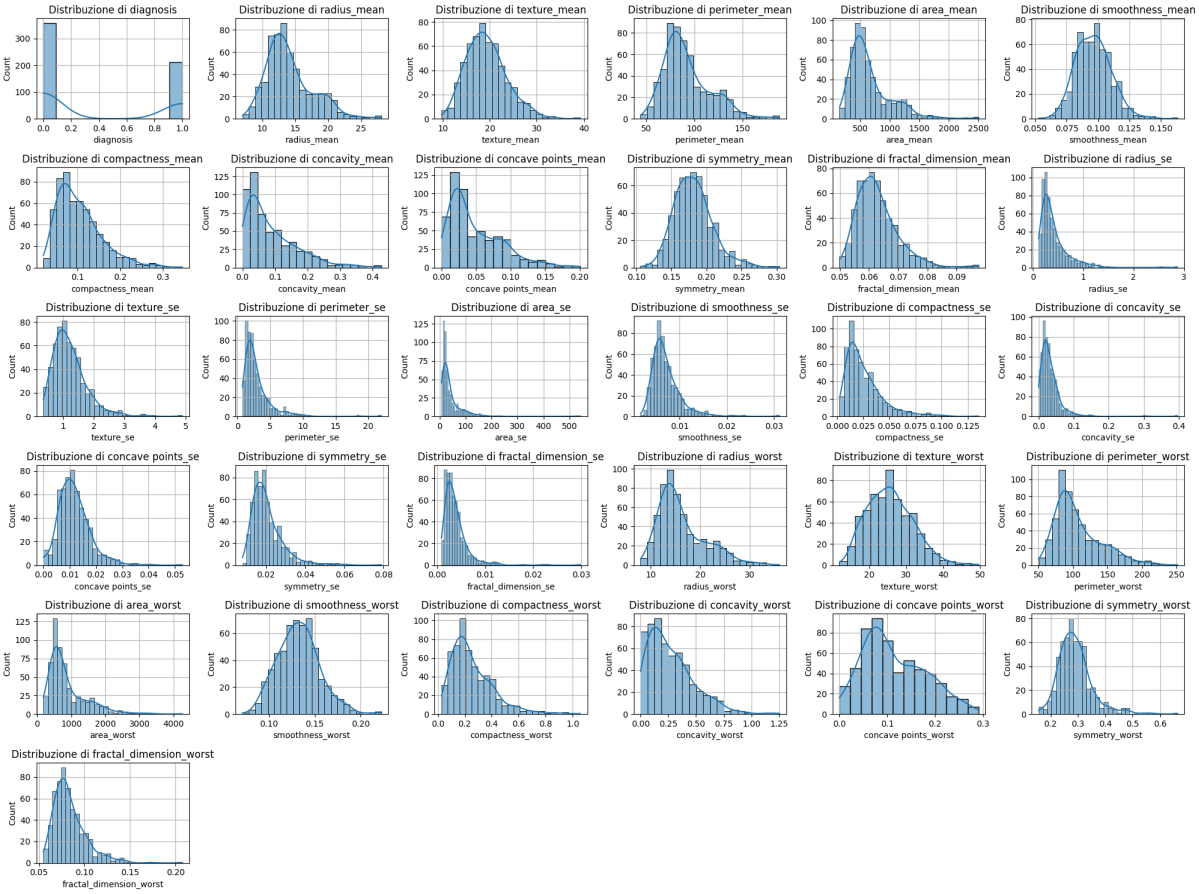


Figure 3.2: Plot e sinusoidi delle variabili del dataset

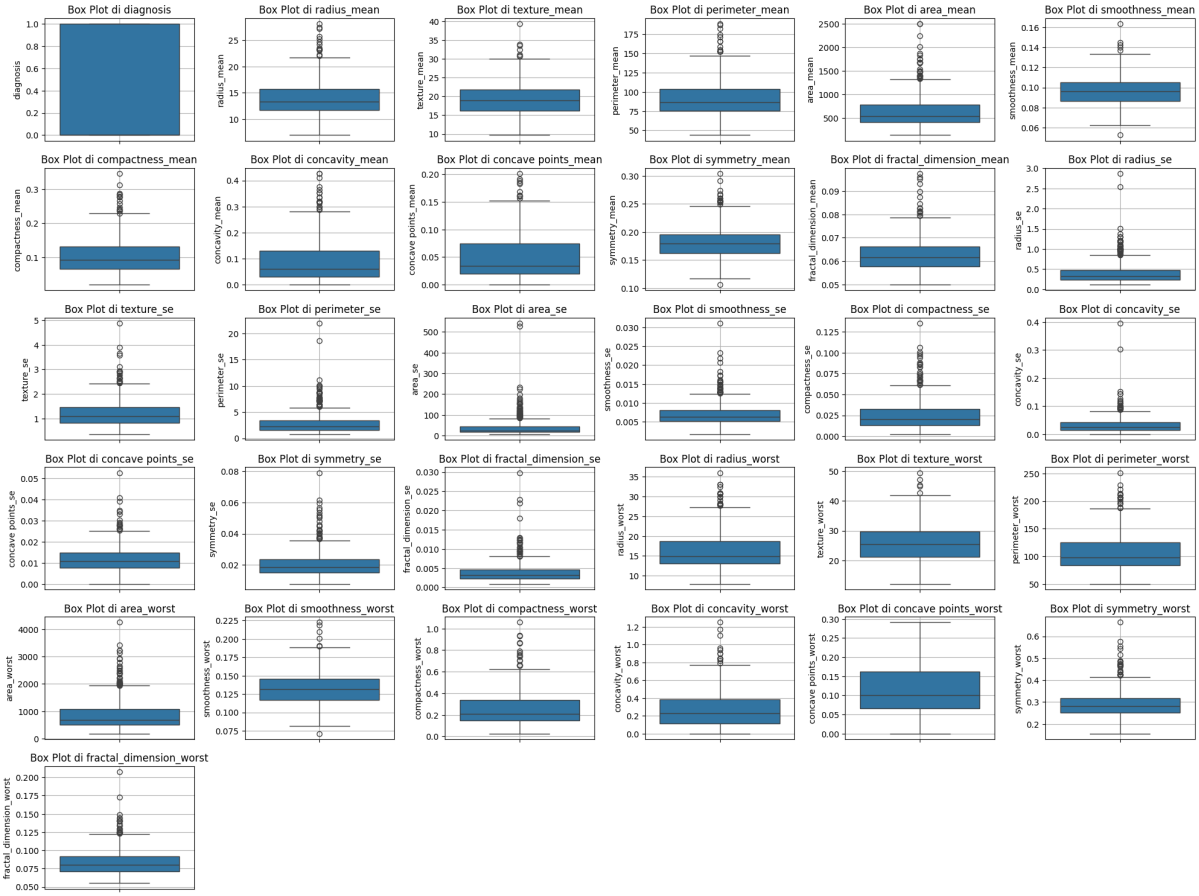


Figure 3.3: BoxPlot sinusoidi delle variabili del dataset

3.2 Multi-Variante

L'analisi univariata non basta a evidenziare le relazioni tra variabili. Per questo, abbiamo utilizzato una heatmap della matrice di correlazione, che offre una panoramica visiva delle correlazioni lineari tra le variabili. I colori caldi indicano correlazioni positive, quelli freddi negative. Questo strumento consente di individuare rapidamente variabili fortemente correlate e possibili ridondanze, facilitando la comprensione delle interazioni tra le variabili. Di seguito il grafico ottenuto:

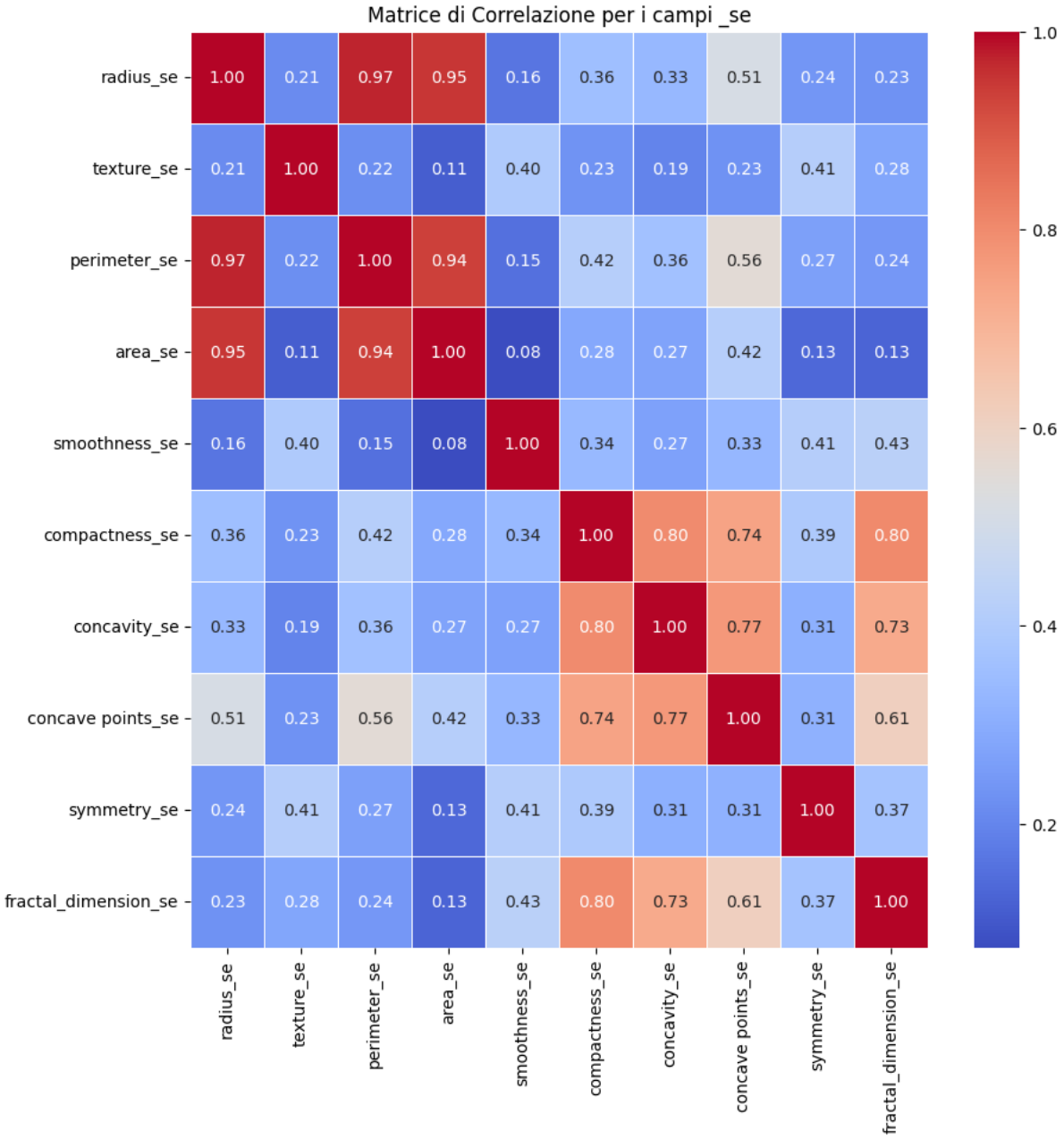


Figure 3.4: Matrice di confusione con i campi "se"

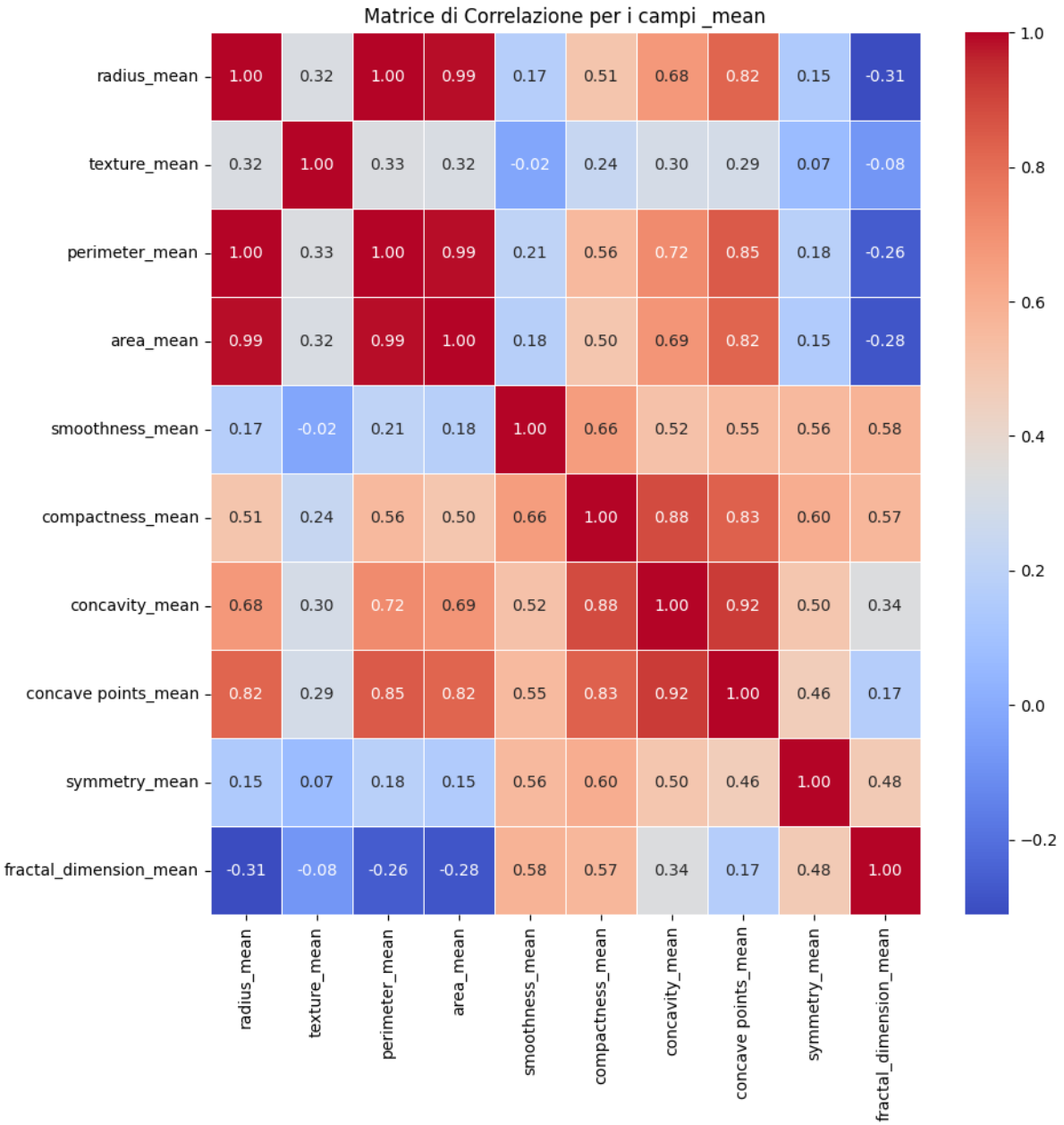


Figure 3.5: Matrice di confusione con i campi "mean"

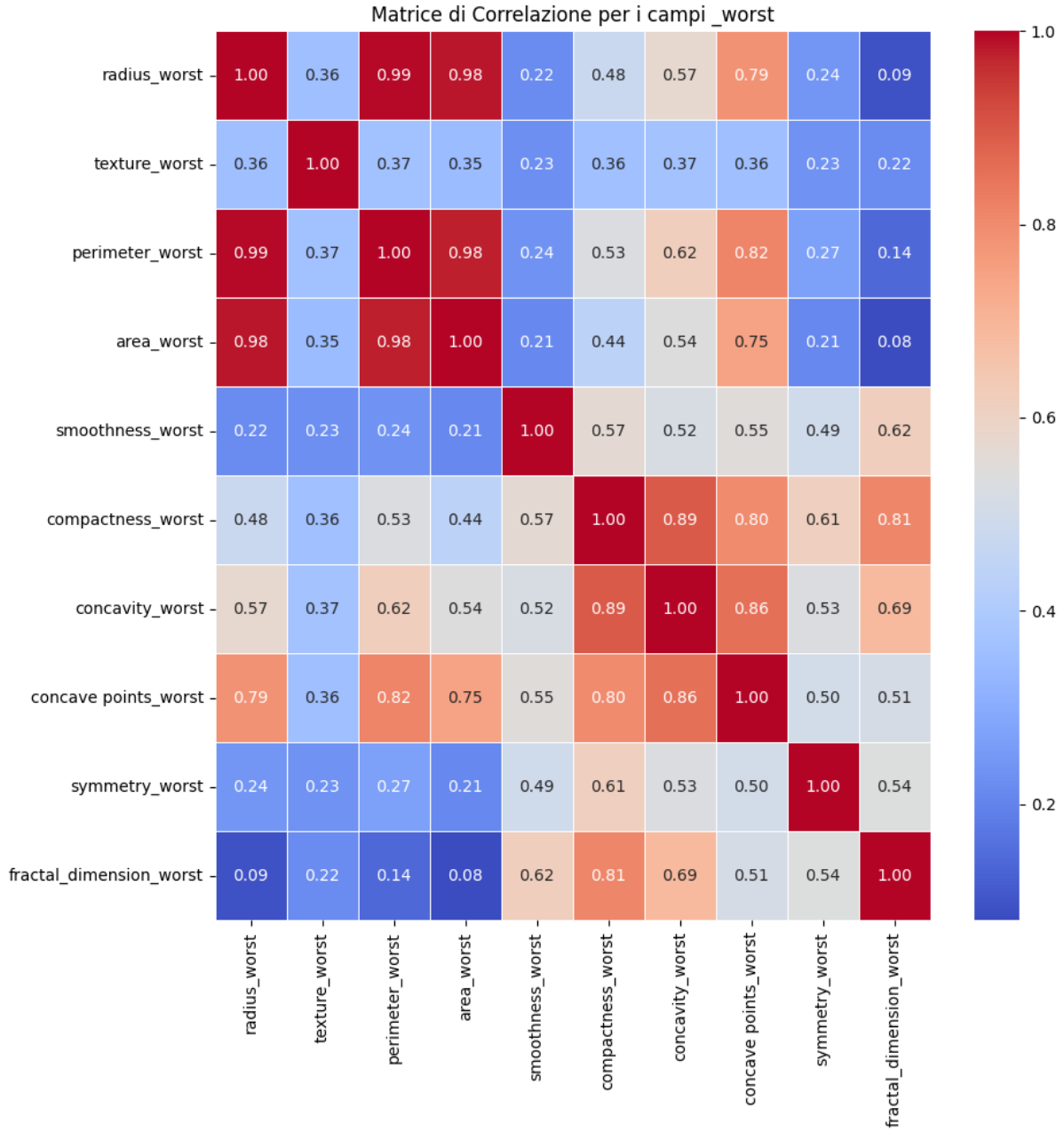


Figure 3.6: Matrice di confusione con i campi "worst"

3.3 PCA

Nella fase conclusiva dell'analisi, abbiamo applicato la PCA (Principal Component Analysis), una tecnica multivariata usata per ridurre la dimensionalità del dataset, mantenendo le informazioni più rilevanti. La PCA trasforma le variabili originali in nuove variabili, dette componenti principali, che riassumono la struttura dei dati. Poiché opera solo su dati numerici, è stato necessario escludere le variabili non numeriche e standardizzare le restanti.

Abbiamo quindi utilizzato la funzione `StandardScaler()` per normalizzare le variabili, garantendo la stessa scala di misura. Successivamente, abbiamo applicato la PCA sui dati standardizzati mediante le apposite funzioni della libreria.

Infine, abbiamo analizzato graficamente la varianza spiegata da ciascuna componente principale. La varianza misura la dispersione dei dati e consente di valutare quanta informazione viene conservata da ogni componente.

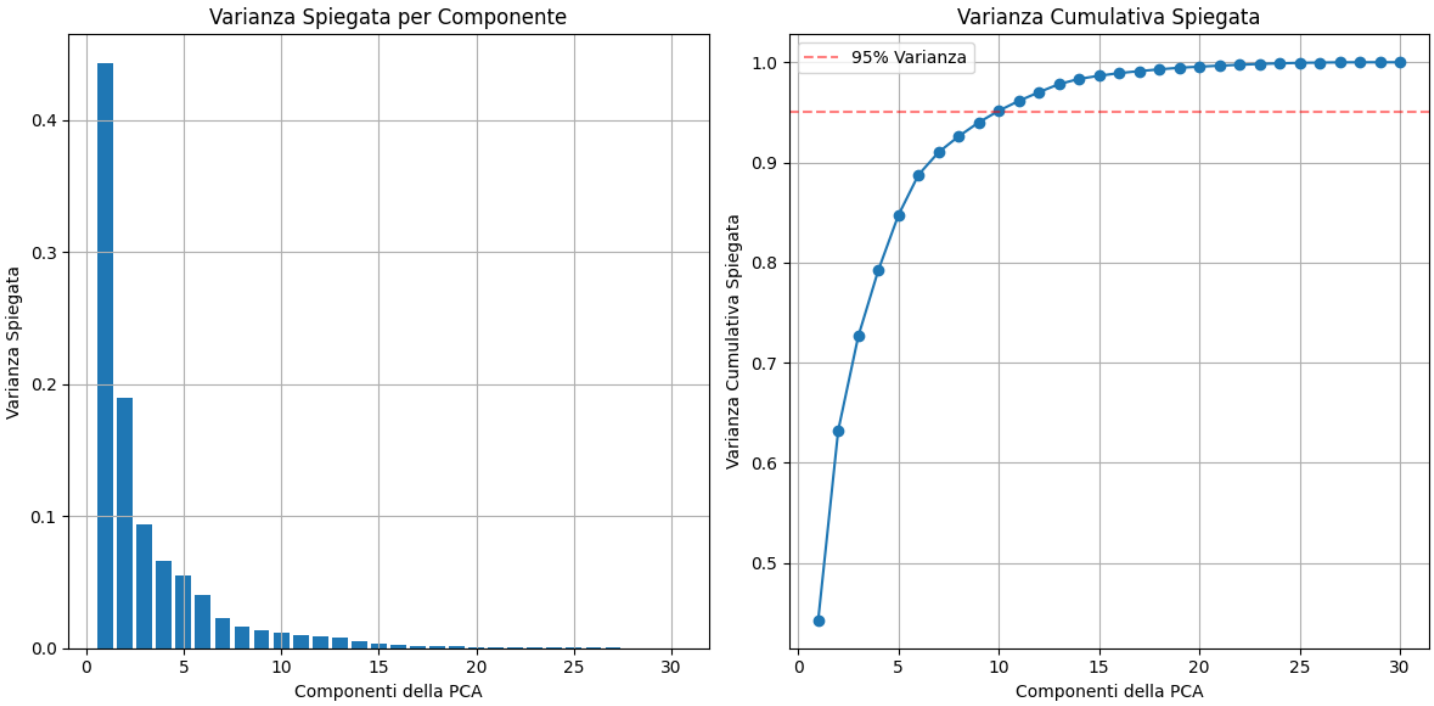


Figure 3.7: Grafici che rappresentano la varianza

Dal grafico ottenuto, nella sezione di destra (varianza cumulativa), è evidente che le prime 10-15 componenti catturano circa il 95% della varianza totale dei dati, raggiungendo la soglia indicata dalla linea rossa tratteggiata. Questo suggerisce che si può ridurre significativamente la dimensionalità del dataset mantenendo queste prime componenti senza perdere informazioni rilevanti.

Il grafico di sinistra mostra la varianza spiegata da ciascuna componente individualmente. La prima componente è dominante, spiegando circa il 45% della varianza totale, seguita dalla seconda (19%) e dalla terza (9%). Le componenti successive contribuiscono progressivamente meno, con un netto calo dopo la quinta componente, confermando che la maggior parte dell'informazione è concentrata nelle prime componenti.

Chapter 4

Applicazione modelli

In questa fase si applicano i modelli di machine learning, analizzandone le prestazioni rispetto agli obiettivi, con particolare attenzione alla capacità predittiva e di generalizzazione.

4.1 Preparazione del Dataset

Per prima cosa, abbiamo preparato due dataset distinti:

- **Dataset originale**: ottenuto con il preprocessing descritto nel Capitolo 2, che include la rimozione della variabile target nelle fasi di training, l'eliminazione di colonne ridondanti o non informative, la gestione dei valori mancanti e altre trasformazioni per ottimizzare l'analisi.
- **Dataset trasformato**: derivato dal precedente, ma ulteriormente processato con PCA e standardizzazione, per ridurre la dimensionalità e migliorare la gestione delle feature da parte dei modelli, soprattutto quelli sensibili alla scala.

Una volta ottenuti i dataset, si è proceduto alla suddivisione in due insiemi distinti:

- Training set (70%)
- Validation set (30%)

Questa suddivisione è essenziale per valutare correttamente i modelli. Il training set serve per l'addestramento, mentre il validation set permette di testare le performance su dati nuovi, simulando la capacità di generalizzazione. Così si individuano eventuali overfitting e si stima con maggiore affidabilità la reale efficacia del modello.

Chapter 5

Rete Neurale

Il primo modello selezionato per l'implementazione e l'addestramento all'interno di questo progetto è rappresentato dalle Reti Neurali. Si tratta di modelli di apprendimento supervisionato, ovvero sistemi che vengono addestrati su un dataset contenente esempi etichettati, in cui sono noti sia gli input che i corrispondenti output attesi. Durante il processo di addestramento, la rete apprende a correlare gli input con gli output attraverso un meccanismo iterativo di ottimizzazione, che consiste nell'aggiornamento dei pesi associati alle connessioni tra i nodi. L'obiettivo è ridurre al minimo l'errore tra l'output previsto dalla rete e quello reale.

5.1 Motivazione dell'uso della Rete Neurale (MLP)

Si è scelto di utilizzare una rete neurale artificiale (MLP) per questo dataset in quanto si tratta di un modello particolarmente adatto alla classificazione binaria su dati complessi e multidimensionali, come nel caso della diagnosi di tumore al seno.

Nel dettaglio, le motivazioni principali sono:

- **Capacità di modellare relazioni non lineari**:
  Le 30 variabili numeriche del dataset descrivono caratteristiche morfologiche delle cellule tumorali. Queste variabili non sono necessariamente in relazione lineare con la diagnosi (benigno o maligno). La rete neurale, grazie ai layer nascosti e alle funzioni di attivazione non lineari, è in grado di catturare pattern complessi nei dati che altri modelli più semplici potrebbero non cogliere.

- **Adattabilità al problema medico**:
  Nella diagnosi medica è importante minimizzare gli errori, soprattutto i falsi negativi. Le reti neurali, se ben addestrate, hanno dimostrato di avere ottime prestazioni in termini di accuratezza, recall e AUC, rendendole ideali per contesti in cui è necessario identificare con precisione i casi positivi (in questo caso, tumori maligni).

- **Esperienza pratica e sperimentazione controllata**:
  Il modello MLP è relativamente semplice da implementare con le moderne librerie di machine learning (es. `sklearn`, `keras`). Questo mi ha permesso di sperimentare diversi parametri (numero di neuroni, epoche, ecc.) e ottimizzare le prestazioni tramite validazione incrociata, rendendo il modello robusto.

- **Ottimo compromesso tra performance e flessibilità**:
  Rispetto a modelli più interpretabili ma limitati (come la regressione logistica), la rete

neurale offre una maggiore capacità predittiva senza essere troppo sensibile al rumore o ai dati sbilanciati. Questo la rende una scelta efficace per un dataset diagnostico ben strutturato e privo di valori mancanti come questo.

5.2 Obiettivo

L'obiettivo principale delle reti neurali è quello di apprendere e modellare relazioni complesse tra variabili, emulando in parte il funzionamento del cervello umano. Le reti neurali artificiali sono costituite da unità elementari chiamate nodi o neuroni artificiali, organizzati in tre tipologie di livelli:

1. uno o più livello di input
2. uno o più livelli nascosti
3. uno o più livello di output

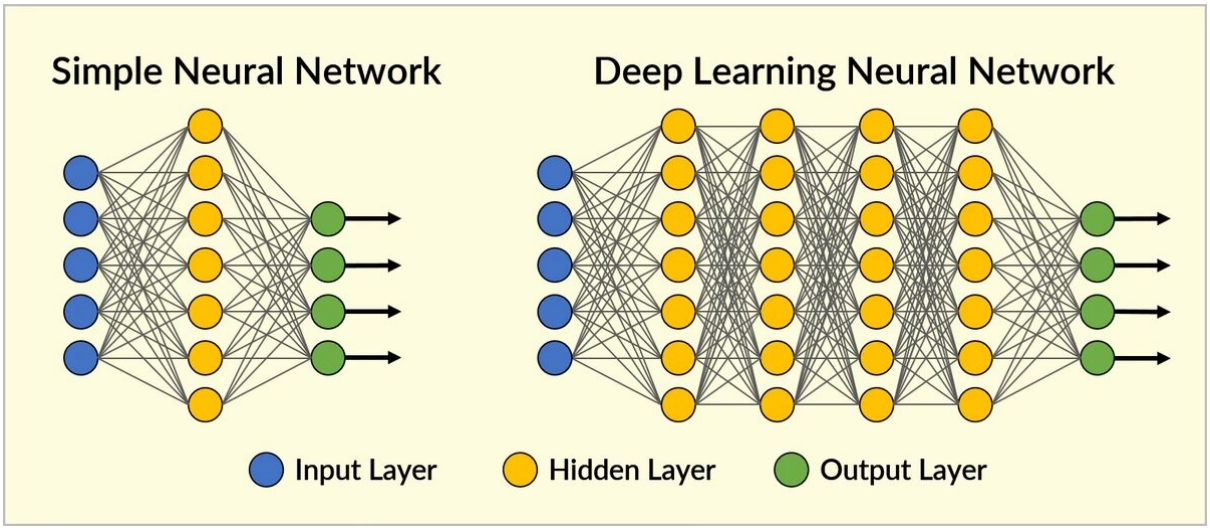


Figure 5.1: Esempio di una struttura di una rete neurale

Ogni connessione tra neuroni è caratterizzata da un peso, che rappresenta l'importanza del segnale trasmesso. L'addestramento della rete si basa su due fasi principali: una fase di propagazione, in cui gli input attraversano i vari strati fino a generare un output, e una fase di retro-propagazione dell'errore, in cui i pesi vengono aggiornati per migliorare progressivamente l'accuratezza del modello.

5.3 Impostazione del Modello

Procediamo con la definizione dell'architettura del modello di rete neurale utilizzando la libreria Keras. Iniziamo creando un modello sequenziale tramite la funzione Sequential(). Questo modello è composto da un primo layer di input, che contiene 30 neuroni, corrispondenti alle 30 feature presenti nel nostro dataset. Successivamente, decidiamo di impostare il layer a 20 neuroni il secondo layer, in seguito a una serie di esperimenti che hanno prodotto risultati ottimali. Per questo strato, utilizziamo la funzione di attivazione ReLU (Rectified Linear Unit),

che ha mostrato buone performance. Infine, il layer di output è composto da un singolo neurone e utilizziamo la funzione di attivazione sigmoide, poiché ci troviamo di fronte a un problema di classificazione binaria. ReLU (Rectified Linear Unit) è una funzione di attivazione che restituisce zero per input negativi e mantiene inalterato il valore dell'input per valori positivi, permettendo reti più profonde grazie alla sua semplicità computazionale e alla riduzione del problema del gradiente evanescente.

Funzione Sigmoide

La funzione di attivazione sigmoide è comunemente impiegata nei problemi di classificazione binaria. Essa comprime l'output in un intervallo compreso tra 0 e 1, rendendola particolarmente adatta per rappresentare una probabilità di appartenenza a una delle due classi. In altre parole, la sigmoide trasforma l'output grezzo della rete neurale in una probabilità, permettendo così di interpretare l'output come la probabilità che un dato input appartenga alla classe positiva.

Struttura del modello

Per quanto riguarda la configurazione e la compilazione dell'addestramento del modello, è stata utilizzata la funzione model.compile, in cui è stata specificata la funzione di loss da minimizzare durante il processo di training.

In particolare, è stata scelta la funzione "binary_crossentropy", ideale per i problemi di classificazione binaria. Inoltre, è stato definito l'ottimizzatore da impiegare per aggiornare i pesi del modello durante l'addestramento: in questo caso, si utilizza l'algoritmo Adam, che regola dinamicamente il tasso di apprendimento per ciascun parametro del modello.

Inoltre, sono state specificate le metriche da monitorare durante l'addestramento. In particolare, è stata scelta l'accuratezza del modello, che rappresenta la percentuale di previsioni corrette rispetto al totale delle previsioni effettuate.

```python
def create_and_train_model(X_train, y_train, X_test, y_test):
    # Usa l'approccio funzionale di Keras con Input layer
    inputs = Input(shape=(X_train.shape[1],))
    x = Dense(20, activation="relu")(inputs)
    outputs = Dense(1, activation="sigmoid")(x)

    # Crea il modello
    model = Model(inputs=inputs, outputs=outputs)

    # Compila il modello
    model.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])
    model.summary()
    print("\n")

    # Addestramento del modello
    history = model.fit(
        X_train, y_train,
        epochs=50,
        batch_size=10,
        validation_data=(X_test, y_test),
        verbose=2
    )
    return model, history
```

Figure 5.2: Codice della rete neurale

Infine, il modello è stato addestrato per 50 epoche con un batch size di 10. Un'epoca corrisponde a un ciclo completo in cui il modello viene allenato su tutto il dataset di training. In ogni epoca, il modello esegue il calcolo delle previsioni sui dati di addestramento e aggiorna i pesi per ridurre l'errore rispetto ai target desiderati.

Il batch size di 10 indica che i dati di training sono suddivisi in piccoli gruppi di 10 campioni per volta. Durante ogni ciclo di addestramento, il modello aggiorna i pesi utilizzando solo il batch corrente. Utilizzare un batch size inferiore, come 10, permette una migliore generalizzazione, poiché i pesi vengono aggiornati più frequentemente rispetto a un batch size più grande, riducendo la possibilità di rimanere bloccati in ottimi locali.

Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 30)	0
dense (Dense)	(None, 20)	620
dense_1 (Dense)	(None, 1)	21

Total params: 641 (2.50 KB)
Trainable params: 641 (2.50 KB)
Non-trainable params: 0 (0.00 B)

Figure 5.3: Struttura finale della rete neurale

5.4 Confronto modelli

Per valutare l'efficacia dei modelli, abbiamo applicato i due approcci sui due dataset precedentemente preparati: uno con i dati originali e l'altro con i dati trasformati mediante PCA. In entrambi i casi, il modello di rete neurale è stato addestrato e testato per confrontare le performance relative.

Successivamente siamo andati ad utilizzare i due modelli per effettuare una previsione sui due set di test e calcolare le metriche di accuratezza. I risultati ottenuti sono i seguenti:

1. Accuracy with original data: 0.7953
2. Accuracy with PCA-reduced data: 0.9766

Inoltre, siamo andati a visionare a livello grafico la differenza tra i due modelli, con i seguenti grafici:

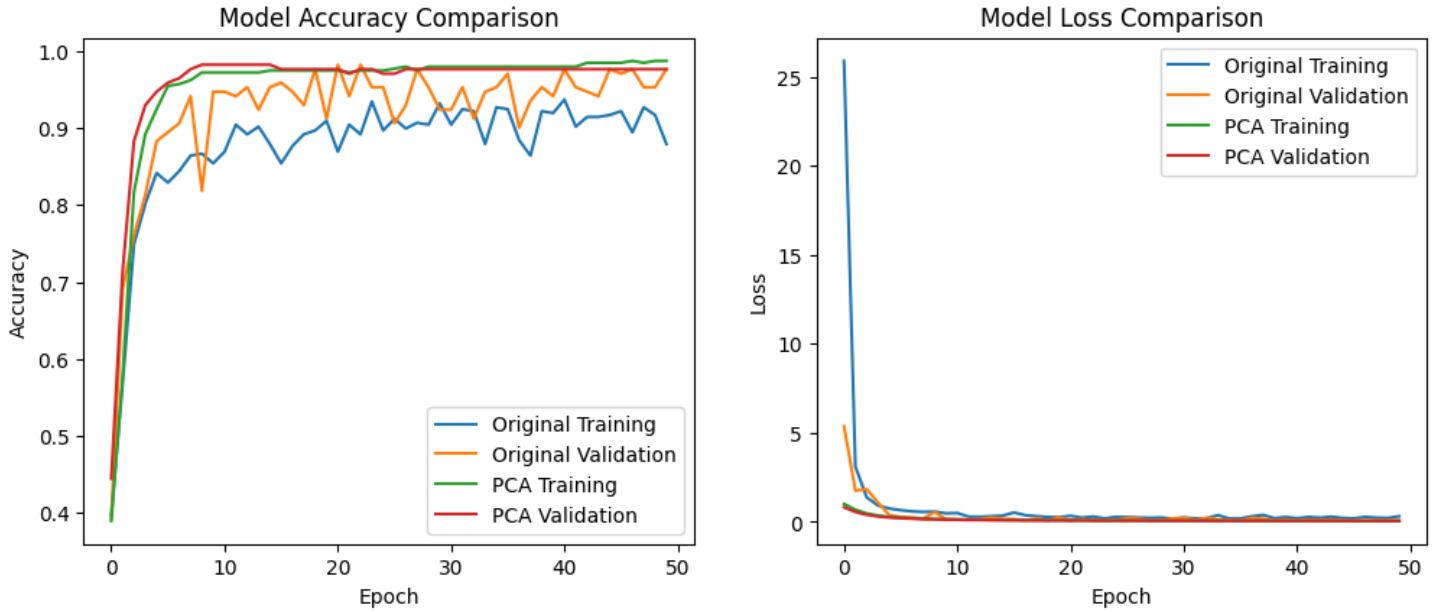


Figure 5.4: Confronto tra accuratezza e perdita dei modelli applicati su due dataset differenti

Analisi dei risultati

Dall'analisi dei grafici delle performance e delle accuracy sui dati di test, fanno emergere una differenza significativa tra i due modelli. Il modello originale ha raggiunto un'accuratezza di circa 90-91% durante l'addestramento, ma solo 79.53% quando valutato sul set di test. In contrasto, il modello addestrato con PCA ha mantenuto performance eccellenti e coerenti sia durante l'addestramento (98-99%) che in fase di test (97.66%).

Questa differenza di circa 18 punti percentuali sul test set merita un'analisi approfondita. Tra le possibili spiegazioni:

1. **Rimozione delle variabili irrilevanti tramite PCA:**La riduzione della dimensionalità ha permesso al modello di concentrarsi sulle informazioni più utili, migliorando l'accuratezza e la generalizzazione.

2. **Miglior convergenza e stabilità:**Il grafico della loss mostra che il modello PCA raggiunge una convergenza più stabile con valori di loss inferiori.

3. **Minore overfitting:**Il modello originale mostra una significativa discrepanza tra performance di addestramento (90%) e test (79.53%), indicando overfitting. Il modello PCA mantiene performance coerenti in entrambe le fasi.

4. **Maggiore stabilità:**Le curve di accuratezza del modello PCA sono più lisce e stabili rispetto alle oscillazioni evidenti nel modello originale.

5. **Effetto della standardizzazione:**La standardizzazione applicata tramite PCA ha reso i dati più uniformi, facilitando una migliore performance del modello.

In conclusione, il modello con PCA non solo ha ottenuto un'accuratezza significativamente superiore, ma ha anche dimostrato una migliore capacità di generalizzazione a dati non visti in precedenza, come evidenziato dalla coerenza tra le performance di addestramento e test.

5.5 Analisi del modello

Analizzando il grafico (figura 5.4), possiamo affermare che il modello addestrato ha ottenuto ottime prestazioni. È evidente come sia riuscito a minimizzare l'errore e a massimizzare l'accuratezza sia sui dati di addestramento che su quelli di validazione, evitando fenomeni di overfitting.

Il Grafico 5.4 (sezione di destra) illustra l'andamento della loss function, ovvero la funzione di costo che il modello cerca di minimizzare durante il processo di apprendimento, in funzione del numero di epoche. La curva blu rappresenta la loss sul training set, mentre quella arancione mostra la loss sul validation set. Inizialmente, la loss sul training è piuttosto elevata ma cala rapidamente nelle prime epoche, segno che il modello sta effettivamente apprendendo dai dati. La loss sul validation set si riduce più gradualmente, il che è del tutto normale poiché questi dati non vengono utilizzati direttamente durante l'addestramento. Dopo circa 15 epoche, entrambe le curve si stabilizzano su valori contenuti, indicando che il modello ha acquisito una buona capacità di generalizzazione.

Nello stesso grafico(sezione di sinistra), invece, abbiamo l'evoluzione dell'accuratezza, ovvero la percentuale di previsioni corrette, sia sul training set che sul validation set. Anche qui, la curva blu rappresenta l'accuratezza sul training set, mentre quella rossa quella sul validation set. Come atteso, l'accuratezza sul training cresce rapidamente nelle prime epoche, raggiungendo valori prossimi al 90%. L'accuratezza sul validation set presenta maggiori oscillazioni, ma segue un trend di crescita analogo, attestandosi anch'essa intorno al 90% nelle ultime epoche. Questo comportamento suggerisce che il modello non soffre di sovra-addestramento e mantiene una buona affidabilità su dati non visti.

Adesso andremo a valutare il modello con altre metriche, per visionare in maniera accurata le effettive prestazioni e performance del modello.

5.5.1 Report di classificazione

Questi rapporti forniscono metriche di valutazione dettagliate, come precision, recall e F1-score, permettendo un'analisi completa delle capacità predittive del modello sia su dati già noti (training set), sia su dati non visti durante l'addestramento (test set).

Report di Classificazione - Set di Training:

	precision	recall	f1-score	support
Benigno	0.99	0.99	0.99	249
Maligno	0.99	0.98	0.98	149
accuracy			0.99	398
macro avg	0.99	0.99	0.99	398
weighted avg	0.99	0.99	0.99	398

Report di Classificazione - Set di Test:

	precision	recall	f1-score	support
Benigno	0.99	0.97	0.98	108
Maligno	0.95	0.98	0.97	63
accuracy			0.98	171
macro avg	0.97	0.98	0.98	171
weighted avg	0.98	0.98	0.98	171

Figure 5.5: Report di classificazione del modello

I risultati mostrati nell'immagine 5.5 evidenziano le ottime prestazioni del modello di classificazione, con un'elevata accuratezza sia sul set di addestramento che su quello di test. In particolare, le metriche di precision, recall e F1-score risultano tutte prossime a 0,99 per entrambe le classi, a conferma dell'elevata capacità del modello nel riconoscere correttamente sia i casi positivi che quelli negativi.

Significato delle metriche:

- **Accuracy:** rappresenta la percentuale di istanze correttamente classificate sul totale. Un'accuracy elevata indica che il modello commette pochi errori complessivi.
- **Precision:** misura la proporzione di veri positivi sul totale dei casi predetti come positivi. Valori alti indicano che il modello commette pochi falsi positivi.
- **Recall (Sensibilità):** indica la proporzione di veri positivi sul totale dei reali positivi. Valori alti suggeriscono che il modello rileva quasi tutti i casi positivi, con pochi falsi negativi.
- **F1-score:** è la media armonica tra precision e recall. Fornisce una valutazione bilanciata, particolarmente utile quando si vuole evitare di favorire una metrica a discapito dell'altra.

Per concludere, è interessante notare che, nel nostro caso, le performance sul set di test risultano leggermente superiori a quelle sul set di addestramento, un comportamento che solitamente indica buona generalizzazione e assenza di overfitting significativo. Ciò conferma che il modello non solo è stato addestrato efficacemente, ma è anche capace di mantenere prestazioni elevate su dati mai visti prima.

5.5.2 Matrice confusione

La matrice di confusione è uno degli strumenti che si possono utilizzare per valutare le prestazioni del modello:

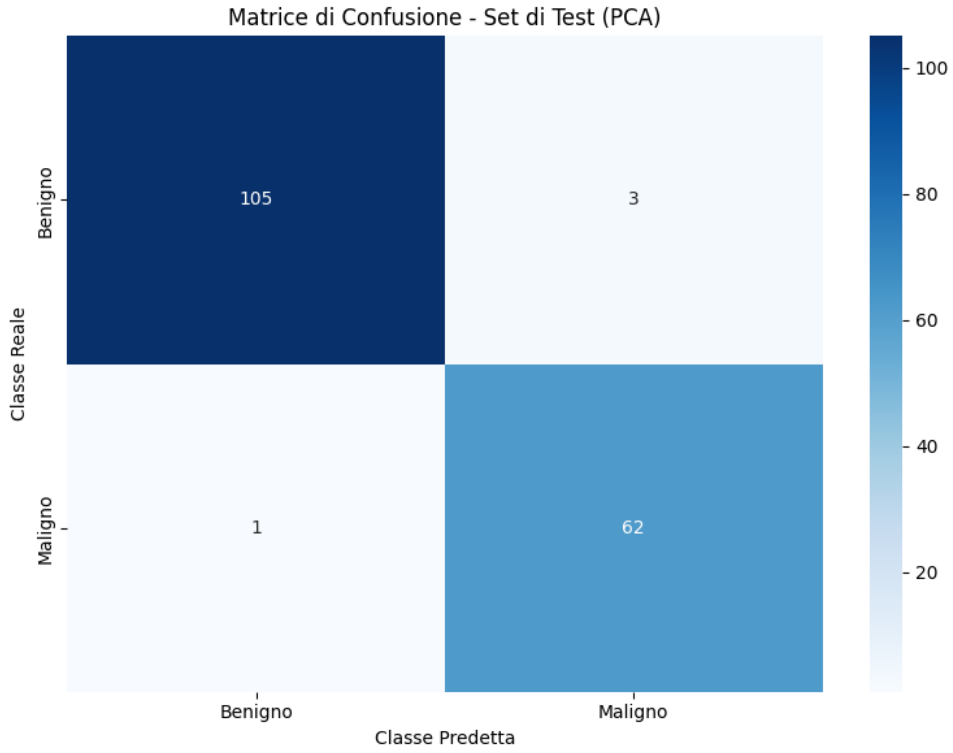


Figure 5.6: Matrice di confusione

Dall'analisi della matrice di confusione ottenuta sui dati di test, emergono le seguenti considerazioni:

- **Veri Positivi (TP):** sono state correttamente identificate 105 istanze positive.
- **Falsi negativi (FN):** sono stati identificati 3 casi in cui le istanze positive sono state erroneamente classificate come negative
- **Falsi positivi (FP):** è stato identificato un solo caso in cui l'istanza negativa è state erroneamente classificata come positiva
- **Veri negativi (TN):** sono state correttamente identificate 62 istanze negative

5.5.3 Curva ROC

Inoltre, è stata valutata anche la curva ROC per analizzare più nel dettaglio le capacità del modello nella distinzione tra le classi. La curva ROC, infatti, è uno strumento fondamentale per misurare l'efficacia di un classificatore binario, in quanto permette di osservare il comportamento del modello nel distinguere tra esempi positivi e negativi, come tumori benigni e maligni nel nostro caso. Un andamento della curva che tende verso l'angolo in alto a sinistra del grafico indica che il modello è in grado di identificare correttamente la maggior parte dei veri positivi, riducendo al minimo i falsi positivi.

A supporto di questa analisi, è stato considerato anche il valore dell'AUC (Area Under the Curve), che rappresenta sinteticamente la capacità discriminativa del modello. Un valore prossimo a 1 suggerisce un'eccellente capacità predittiva. Nel nostro caso, l'AUC pari a circa 0,9988 conferma che il modello è altamente affidabile e preciso nella classificazione.

Pertanto, sia la curva ROC che l'alto valore dell'AUC dimostrano che il modello presenta prestazioni elevate e un'ottima capacità di generalizzazione, riuscendo a distinguere in modo accurato tra le due classi oggetto di studio.

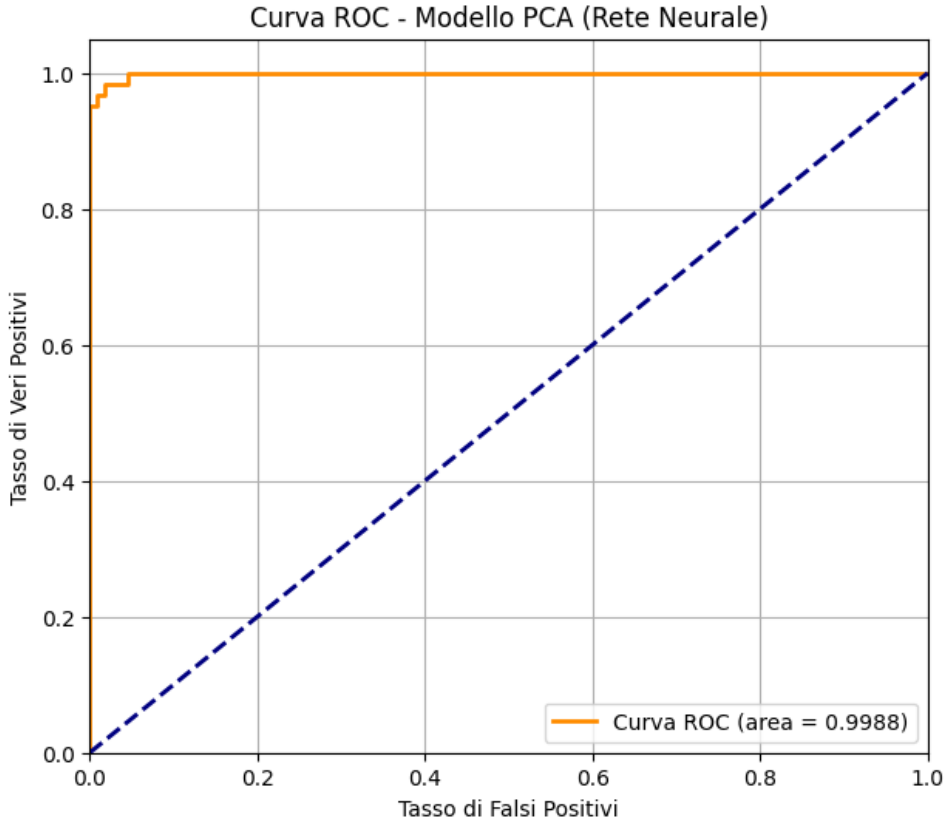


Figure 5.7: Matrice di confusione

5.5.4 10-KFOLDS

Un'ulteriore metrica utilizzata per valutare la robustezza del nostro modello è la cross-validation, una tecnica che consente di stimare le prestazioni di un modello su dati non osservati durante l'addestramento, fornendo una valutazione più realistica della sua capacità di generalizzazione, specialmente con dataset di dimensioni limitate.

Il processo di cross-validation prevede la suddivisione del dataset in più sottoinsiemi (folds). Il modello viene addestrato su una combinazione di questi e testato sul fold rimanente, ripetendo il processo con diverse combinazioni. Abbiamo utilizzato l'approccio a 10-fold StratifiedKFold, in cui il dataset è diviso in 10 parti uguali, con la distribuzione delle classi mantenuta proporzionale in ogni fold. Questo approccio è particolarmente utile quando il dataset è sbilanciato, migliorando la valutazione delle prestazioni del modello.

L'uso di 10-fold StratifiedKFold garantisce una rappresentazione proporzionale delle classi in ogni fold, riducendo il rischio di distorsioni nei risultati dovuti a una distribuzione sbilanciata, e migliorando la stima delle prestazioni su dati non visti.

```
Esecuzione StratifiedKFold con modello PCA esistente...
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
Fold 1/10: Accuracy = 0.9825
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 94ms/step
Fold 2/10: Accuracy = 1.0000
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 74ms/step
Fold 3/10: Accuracy = 1.0000
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
Fold 4/10: Accuracy = 0.9649
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
Fold 5/10: Accuracy = 0.9825
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 54ms/step
Fold 6/10: Accuracy = 0.9825
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step
Fold 7/10: Accuracy = 1.0000
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
Fold 8/10: Accuracy = 0.9649
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 63ms/step
Fold 9/10: Accuracy = 0.9825
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 87ms/step
Fold 10/10: Accuracy = 1.0000

Risultati finali (Modello PCA):
StratifiedKFold Accuracy Media: 0.9860
Deviazione Standard dei punteggi di accuratezza: 0.0131
Intervallo di confidenza al 90%: [0.9779, 0.9940]
Tempo di esecuzione del 10-fold StratifiedKFold: 2.6611 secondi
```

Figure 5.8: Valori strafied 10-kfolds modello rete neurale

I risultati ottenuti applicando la cross-validation mostrano che il modello SVM ha un'accuratezza media del 98.60%, con una deviazione standard di 1.31% (la deviazione standard misura quanto i punteggi di accuratezza variano tra i fold. In altre parole, indica la dispersione dei risultati attorno alla media). Questo evidenzia che il modello ha ottime prestazioni su dati non visti, con una variazione contenuta nelle performance tra i diversi fold della cross-validation.

Il tempo complessivo di esecuzione della cross-validation è stato di circa 2.66 secondi, che riflette il costo computazionale associato alla valutazione del modello su più sottoinsiemi del dataset.

Infine, l'intervallo di confidenza al 90% del modello è stato calcolato e il risultato è compreso tra 0.9779 e 0.9940, suggerendo che il modello ha una stima robusta delle sue prestazioni, con una probabilità del 90% che la vera accuratezza rientri in questo intervallo.

5.5.5 Considerazioni finali

Nel complesso, il modello sviluppato adotta un approccio solido per l'addestramento e la valutazione delle prestazioni su un dataset mirato alla diagnosi del tumore al seno. I risultati ottenuti sono molto promettenti, con un'accuratezza del 98% sia sul set di addestramento che su quello di test. Le metriche di precisione confermano la capacità del modello di distinguere efficacemente tra casi positivi e negativi, con una bassa incidenza di falsi positivi e falsi negativi, come evidenziato anche dalle matrici di confusione.

Chapter 6

SVM

Il modello selezionato per l'implementazione e l'addestramento è SVM (Support Vector Machine). È un modello di apprendimento supervisionato e viene usato per problemi di classificazione. L'obiettivo è trovare l'iperpiano che separa al meglio due (Benigno e Maligno) classi di dati.

La SVM funziona cercando la linea (o iperpiano, se i dati hanno più dimensioni) che divide i dati nel modo più netto possibile, lasciando la maggiore distanza tra le classi. Se i dati non sono facilmente separabili, la SVM può trasformarli per trovare una separazione più chiara.

6.1 Motivazione dell'uso della SVM (Support Vector Machine)

Ho scelto di utilizzare un modello SVM (Support Vector Machine) perché è particolarmente adatto a compiti di classificazione binaria su dataset con molte feature numeriche, come quello utilizzato per la diagnosi del tumore al seno.

Le motivazioni principali sono:

- **Efficacia in spazi ad alta dimensionalità**:
  Il dataset include 30 variabili numeriche, derivate da misurazioni morfologiche delle cellule. Le SVM sono note per funzionare molto bene in spazi con molte feature, mantenendo alte prestazioni anche quando alcune variabili sono ridondanti o correlate.

- **Buona capacità di generalizzazione**:
  Le SVM cercano di massimizzare il margine tra le classi, cioè la distanza tra i punti più vicini alla frontiera di decisione. Questo approccio tende a produrre modelli che generalizzano meglio su dati mai visti, riducendo il rischio di overfitting, soprattutto in contesti dove i dati non sono estremamente numerosi.

- **Possibilità di modellare relazioni non lineari**:
  Grazie all'uso del *kernel trick* (in particolare il kernel RBF), la SVM può modellare confini decisionali curvi e non lineari, permettendo di distinguere in modo accurato tra tumori benigni e maligni anche quando la separazione tra le classi non è lineare nello spazio originale.

- **Stabilità e robustezza**:
  La SVM è meno sensibile alla scelta dei parametri e funziona bene anche in presenza di dati rumorosi. Questo la rende particolarmente adatta a contesti clinici, dove è importante avere modelli stabili e affidabili anche in presenza di variazioni nei dati.

```python
def create_train_SVM(X_train, y_train, dataset_name):
    # Creazione e addestramento del modello SVM
    print(f"\nCreazione e addestramento SVM su {dataset_name}:")
    start_time = time()
    svm_model = SVC(kernel="linear", probability=True, random_state=54)
    svm_model.fit(X_train, y_train)
    end_time = time()
    training_time = end_time - start_time

    print(f"Tempo di training: {training_time:.4f} secondi")

    return svm_model, training_time
```

Figure 6.1: Codice di SVM

6.2 Impostazione modello

Nel progetto viene utilizzata l'implementazione del modello della libreria sklearn, per inizializzare SVM sono stati modificati 3 parametri, rispetto a quelli di default:

1. **"Kernel=linear"**: specifica che il modello deve usare un iperpiano lineare per separare le classi e si utilizza quando si suppone che siano separabili in modo lineare;

2. **"probability=true"**: attiva il calcolo delle probabilità di appartenenza alle classi tramite stima statistica, è utile se vogliamo usare metodi che richiedono probabilità , ma rallenta l'addestramento;

3. **"random_state=54"**: serve per rendere riproducibili i risultati: fissando un seme casuale, l'addestramento fornirà sempre lo stesso modello a parità di dati e impostazioni.

6.3 Confronto modelli

Per mostrare l'efficacia dei modelli abbiamo applicato i due approcci sui due dataset presentati precedentemente: uno con i dati originali e l'altro con i dati trasformati mediante PCA.

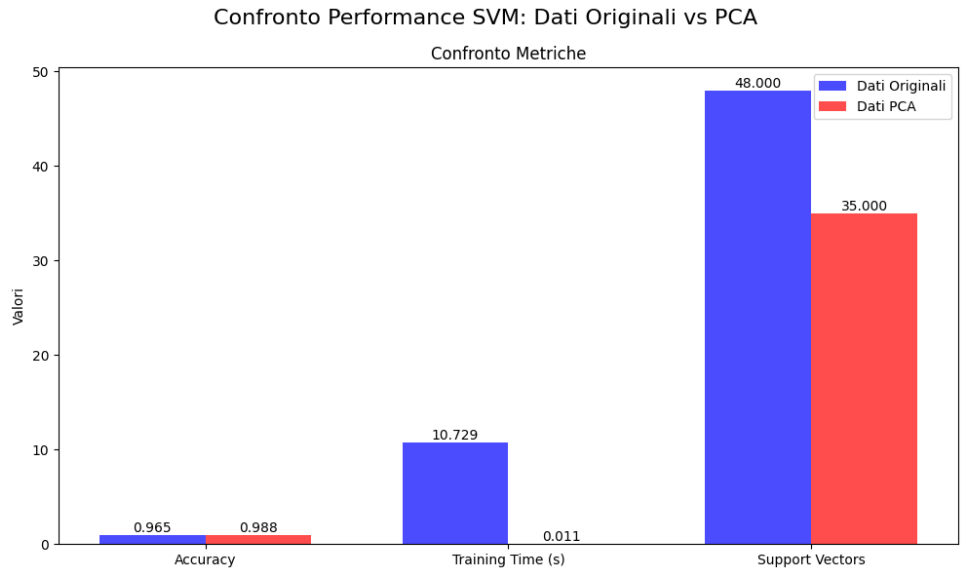


Figure 6.2: Confronto sui modelli che sono stati addestrati su dataset differenti

L'applicazione della PCA ha portato ad un miglioramento significativo delle performance del modello SVM:

- L'**accuracy** complessiva è aumentata dal 96% al 99%;
- Si nota un **notevole incremento del recall sulla classe "Maligno"**, passata da 0.94 a 0.98: questo significa meno falsi negativi;
- Le **metriche di precisione e F1-score** migliorano per entrambe le classi, segnalando una maggiore affidabilità complessiva del modello.

Il confronto evidenzia come l'utilizzo della **PCA** abbia contribuito a:

- Ridurre la dimensionalità del dataset e il rumore presente nei dati;
- Rendere il problema più facilmente separabile per un classificatore lineare;
- Aumentare la velocità di addestramento mantenendo (e migliorando) la qualità predittiva;

Alla luce di questi risultati, risulta evidente come l'integrazione della PCA abbia avuto un impatto positivo sulle prestazioni del classificatore SVM. La significativa riduzione del rumore, unita a una maggiore separabilità delle classi e a un miglioramento generale delle metriche, rende questo approccio particolarmente promettente. Pertanto, nei prossimi step dell'analisi ci concentreremo sul modello SVM addestrato sui dati trasformati mediante PCA, approfondendone il comportamento e valutandone la robustezza in scenari più complessi.

6.4 Analisi del modello

6.4.1 Matrice confusione

La matrice di confusione è uno degli strumenti che si possono utilizzare per valutare le prestazioni del modello:

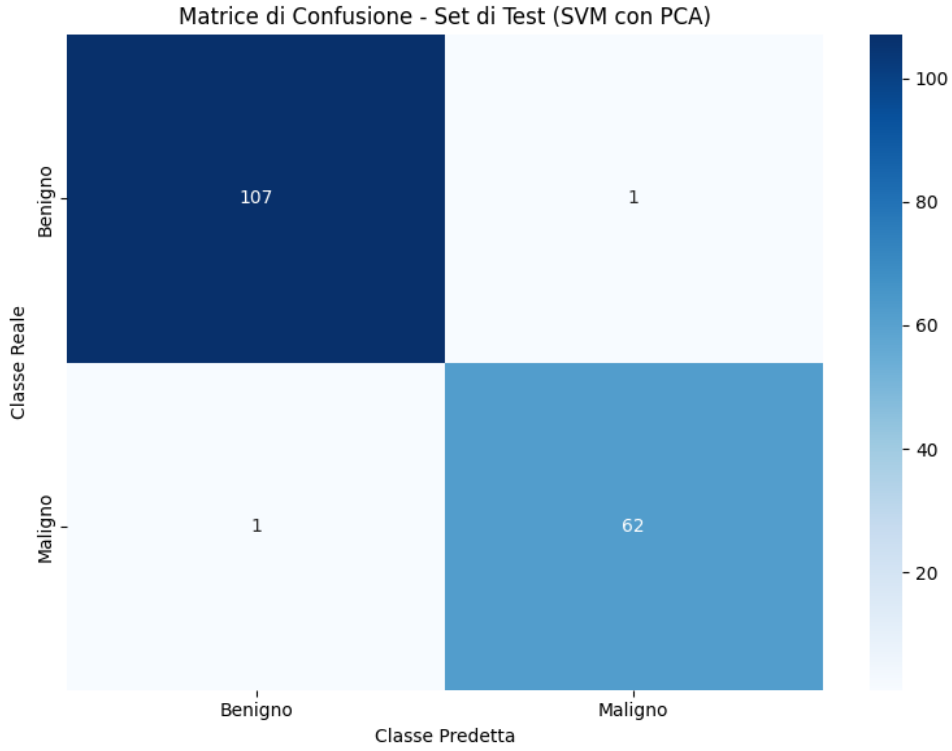


Figure 6.3: Matrice di confusione

Dall'analisi della matrice di confusione ottenuta sui dati di test emergono le seguenti considerazioni:

- **Veri Positivi (TP):** sono state correttamente identificate 107 istanze positive;
- **Falsi Negativi (FN):** sono stati identificati 1 casi in cui le istanze positive sono state erroneamente classificate come negative;
- **Falsi positivi (FP):** sono stati definiti 1 casi in cui l'istanza negativa è state erroneamente classificata come positiva;
- **Veri negativi (TN):** sono state correttamente identificate 62 istanze negative.

6.4.2 Curva ROC

Per analizzare nel dettaglio le capacità del modello nella distinzione tra le classi è stata utilizzata anche per SVM la curva ROC. Un andamento della curva che tende verso l'angolo in alto a sinistra del grafico indica che il modello è in grado di identificare correttamente la maggior parte dei veri positivi,riducendo al minimo i falsi positivi.

Un altro valore da analizzare e da associare è l'AUC (Arena under the Curve), che rappresenta la capacità discriminativa del modello. (Un valore vicino a 1 suggerisce un'eccellente capacità predittiva) In questo caso, l'AUC è pari a 0.9984 e conferma che il modello è affidabile e preciso nella classificazione.

Pertanto, sia la curva ROC che l'alto valore dell'AUC dimostrano che il modello presenta prestazioni elevate e un'ottima capacità di generalizzazione, riuscendo a distinguere in modo accurato tra le due classi oggetto di studio.

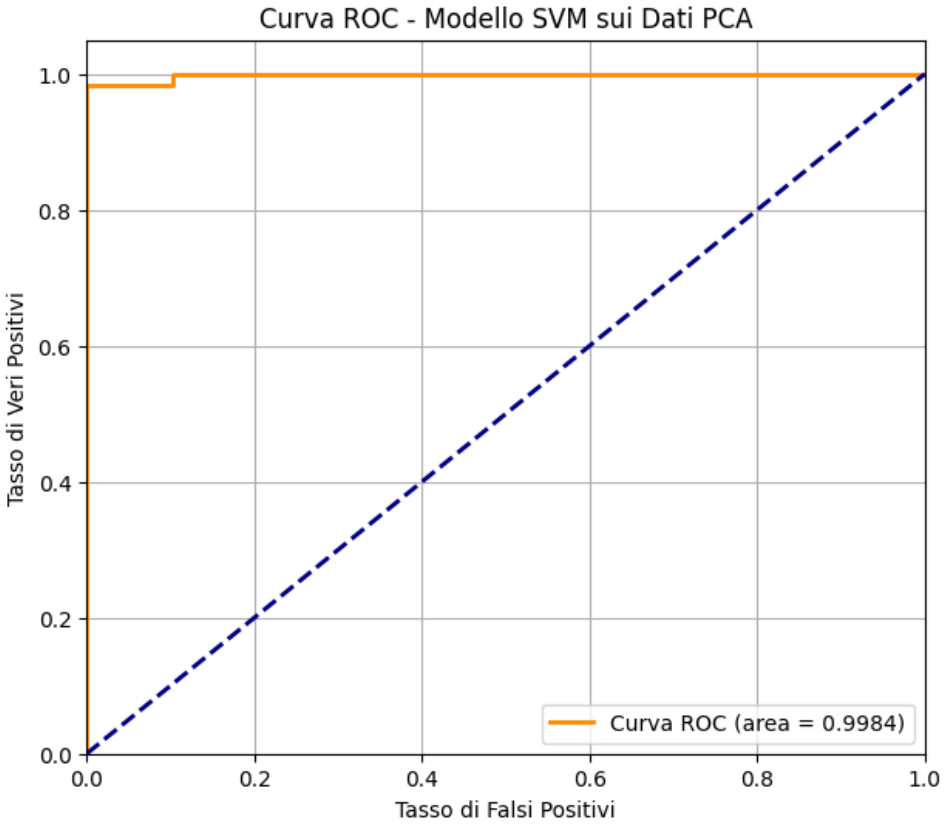


Figure 6.4: Curva ROC

6.4.3 10-KFOLDS

Abbiamo utilizzato l'approccio a 10-fold StratifiedKFold per SVM su dati preprocessati con PCA, in cui il dataset è diviso in 10 parti uguali, con la distribuzione delle classi mantenuta proporzionale in ogni fold.

I risultati ottenuti applicando la cross-validation mostrano che il modello SVM ha un'accuratezza media del 97.89%, con una deviazione standard di 0.0161 (intervallo di confidenza al 90%: 0.9696-0.9881). L'accuratezza nei singoli fold varia da un minimo di 0.9649 a un massimo di

1.0000, con due fold che raggiungono il 100% di accuratezza. Il tempo di esecuzione dell'intero processo di 10-fold StratifiedKFold è stato di soli 0.0188 secondi, dimostrando l'efficienza computazionale di questo approccio.

```
Esecuzione StratifiedKFold con modello SVM su dati PCA...
Fold 1/10: Accuracy = 0.9825
Fold 2/10: Accuracy = 0.9825
Fold 3/10: Accuracy = 1.0000
Fold 4/10: Accuracy = 0.9474
Fold 5/10: Accuracy = 0.9825
Fold 6/10: Accuracy = 0.9649
Fold 7/10: Accuracy = 1.0000
Fold 8/10: Accuracy = 0.9649
Fold 9/10: Accuracy = 0.9825
Fold 10/10: Accuracy = 0.9821

Risultati finali (Modello SVM con PCA):
StratifiedKFold Accuracy Media: 0.9789
Deviazione Standard dei punteggi di accuratezza: 0.0161
Intervallo di confidenza al 90%: [0.9696, 0.9883]
Tempo di esecuzione del 10-fold StratifiedKFold: 0.0188 secondi
```

Figure 6.5: Valori stratified 10-kfolds modello SVM

6.4.4 Considerazioni finali

In generale, il modello creato sembra implementare un approccio solido per addestrare e valutare le prestazioni di classificazione SVM sul dataset selezionato che riguarda la diagnosi del cancro al seno. Le prestazioni del modello sono eccellenti, con un'accuratezza del 98% sia sui dati di addestramento che sui dati di test. Inoltre, i valori di precisione indicando che il modello è in grado di classificare correttamente sia i casi positivi che negativi. Le matrici di confusione mostrano anche un numero ridotto di falsi positivi e falsi negativi, il che è fondamentale in un contesto medico in cui è cruciale minimizzare gli errori di classificazione. Questi risultati suggeriscono che il modello SVM addestrato ha un'elevata capacità di discriminazione e potrebbe essere uno strumento prezioso per supportare la diagnosi del cancro al seno.

Chapter 7

Conclusione

Nel contesto di questo progetto, il gruppo ha affrontato la problematica della predizione del cancro al seno, sviluppando modelli di machine learning mirati all'ottimizzazione e alla prevenzione di tale patologia in base ai valori dei soggetti presi in considerazione. Come illustrato nei capitoli successivi, sono stati selezionati e confrontati i seguenti modelli: Reti Neurali e Support Vector Machine (SVM).

Per valutare e confrontare le prestazioni di questi modelli, sono state impiegate diverse tecniche, tra cui l'analisi degli intervalli di confidenza, la misurazione dei tempi di training e l'utilizzo delle curve ROC(Receiver Operating Characteristic). Nello specifico, le curve ROC sono state ampiamente sfruttate come indicatore diagnostico per valutare l'accuratezza dei modelli predittivi, mentre l'analisi dei tempi di elaborazione ha fornito informazioni cruciali sulla fattibilità e l'efficienza implementativa dei diversi approcci.

Infine, l'inclusione degli intervalli di confidenza ha conferito una maggiore robustezza statistica alle valutazioni, consentendo una stima più approfondita delle differenze prestazionali tra i modelli.

7.1 Confronto Curve ROC

Questa immagine rappresenta le curve ROC dei due modelli di machine learning confrontati: SVM e Rete Neurale, entrambi applicati su dati pre-trattati con PCA. Dal grafico si denota che entrambi i modelli hanno ottenuto prestazioni eccellenti, con AUC: 0.9984 per SVM con PCA e 0.9988 per la Rete Neurale con PCA. La differenza minima (0.0004) a favore della Rete Neurale indica un vantaggio pressoché trascurabile. Le curve di entrambi i modelli si avvicinano significativamente all'angolo superiore sinistro, testimoniando un ottima sensibilità, nettamente superiori rispetto alla linea di riferimento casuale (random) con AUC=0.5.

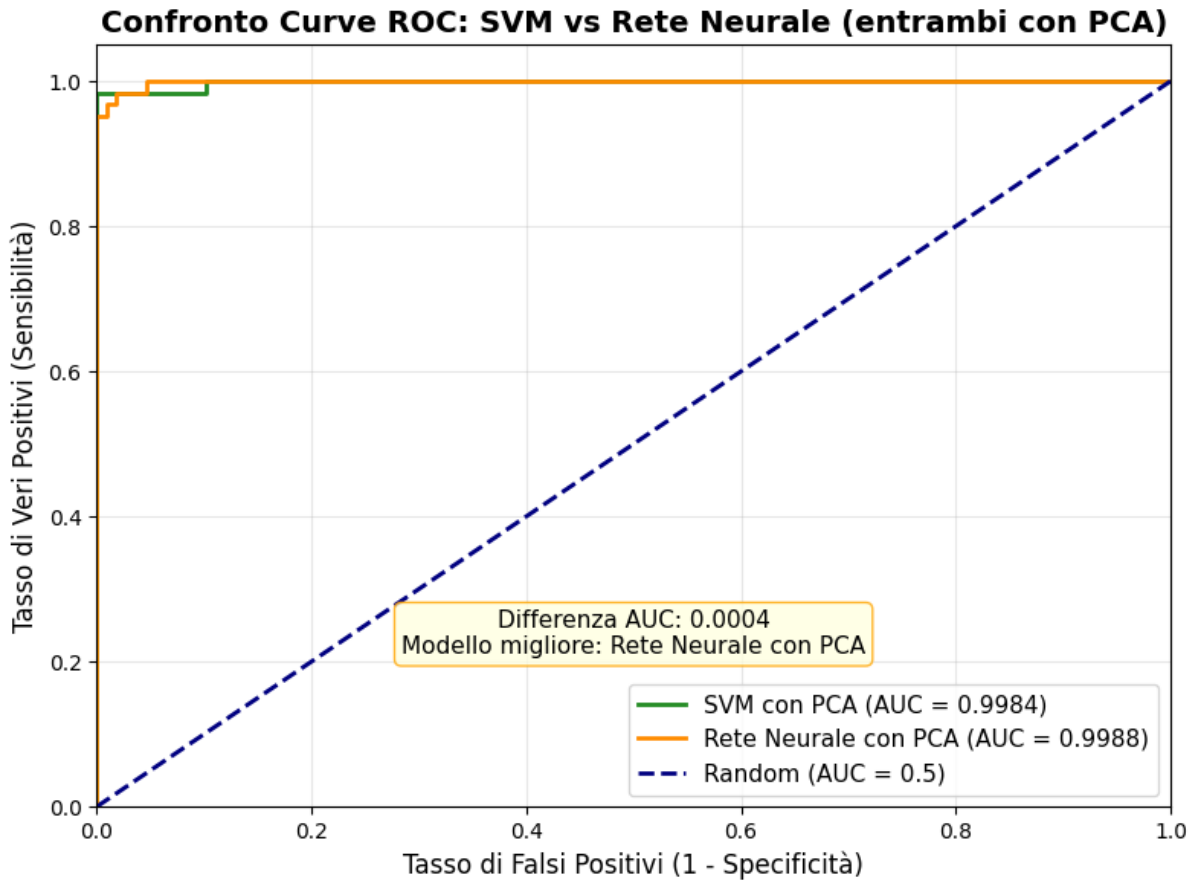


Figure 7.1: Confronto curve roc dei modelli applicati

7.2 Confronto Intervalli di confidenza

L'immagine illustra il confronto degli intervalli di confidenza al 90% per l'accuratezza dei due modelli. La Rete Neurale con PCA ha ottenuto un'accuratezza media leggermente superiore (0.9807) rispetto a SVM con PCA (0.9789), con una differenza di 0.0018. È importante notare la sovrapposizione degli intervalli di confidenza: 0.9732-0.9882 per la Rete Neurale e 0.9696-0.9883 per SVM. Questa sovrapposizione suggerisce che la differenza di performance tra i due modelli probabilmente non è significativa, rendendo entrambi validi dal punto di vista dell'accuratezza predittiva.

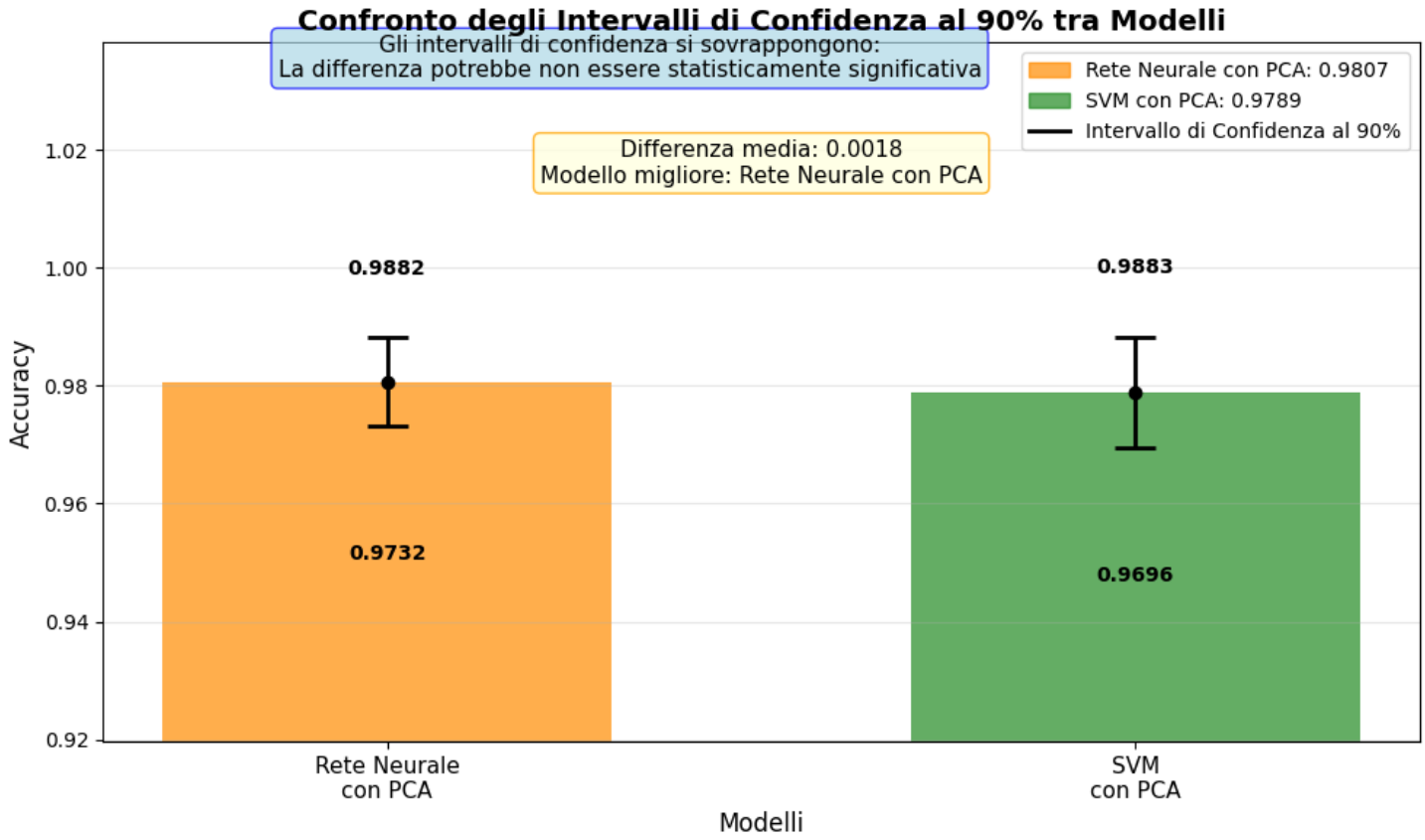


Figure 7.2: Confronto intervalli di confidenza dei modelli applicati

7.3 Confronto tempi di addestramento

L'analisi comparativa dei tempi di addestramento, illustrata nell'immagine, rivela la differenza più sostanziale tra i due modelli. SVM con PCA dimostra un'efficienza computazionale superiore, richiedendo solamente 0.01 secondi per completare l'addestramento, contro i 16.44 secondi necessari alla Rete Neurale con PCA. Questo rappresenta una differenza prestazionale del 99.9% a favore di SVM, come evidenziato dall'annotazione nel grafico.

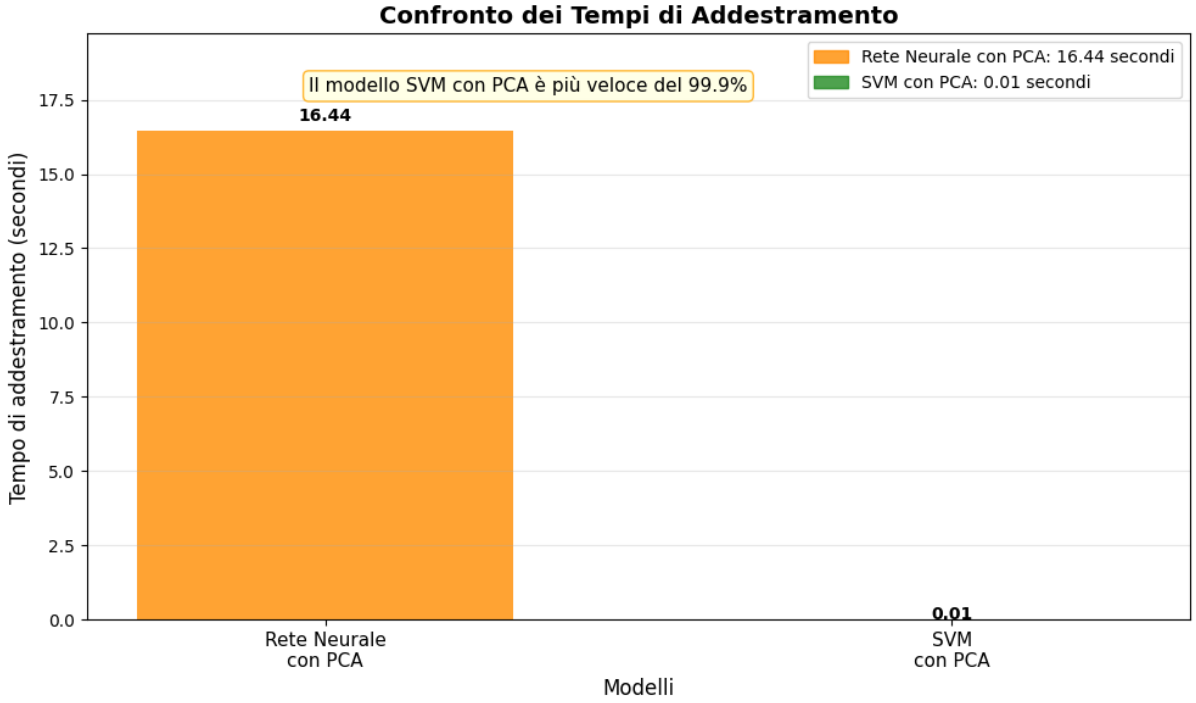


Figure 7.3: Confronto tempi di addestramento dei modelli applicati

7.4 Conclusione Finale

Entrambi i modelli mostrano performance eccezionali sul dataset analizzato, con valori di accuratezza e AUC prossimi alla perfezione. Questa prestazione è probabilmente dovuta a diverse caratteristiche favorevoli del dataset: una chiara separabilità delle classi, l'assenza di variabili categoriche che richiederebbero codifica speciale, e l'efficacia della riduzione dimensionale tramite PCA che ha isolato le componenti più informative. Inoltre, il dataset potrebbe essere privo di outlier significativi. È particolarmente notevole come entrambi i modelli abbiano raggiunto queste prestazioni elevate nonostante lo sbilanciamento della variabile target, dove circa il 62,7% delle tuple appartiene alla classe "Benigno" e il restante 37,3% alla classe "Maligno". Questo suggerisce che sia SVM che la Rete Neurale, combinati con PCA, sono stati in grado di gestire efficacemente lo sbilanciamento delle classi senza necessità di tecniche aggiuntive di bilanciamento. Considerando il compromesso tra accuratezza e efficienza, SVM con PCA emerge come la scelta ottimale, offrendo prestazioni praticamente identiche alla Rete Neurale ma con un tempo di addestramento drasticamente inferiore.