POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Biomedica

Tesi di Laurea Magistrale

Machine Learning e Microwave Imaging per l'ictus cerebrale: un caso di studio

Relatore
Prof.ssa Francesca VIPIANA
Correlatori
Prof. Mario R. CASU
Prof. Jorge A. TOBON VASQUEZ

Candidato
Riccardo SANSIVERI

Anno Accademico 2019/20

Dicembre 2020

A me stesso,
che con grande determinazione ed orgoglio non ho mai mollato,
arrivando ad oggi a realizzare questo grande obiettivo.

Ai miei genitori,
che mi hanno sempre sostenuto in questo percorso
non facendomi mancare mai nulla.

A mia sorella, mio cognato ed Elia,
che sono la mia famiglia.

A mio figlioccio Emanuele,
per avermi supportato e sopportato.

Ai miei nonni che non ci sono più, Angela ed Anna.

Alla mia gattina Emma.

E infine, non per ordine di importanza, alla mia ragazza, Federica,
che in questo "funesto" 2020 è stata una luce in fondo al tunnel
facendomi riscoprire l'amore,
incoraggiandomi ed aiutandomi molto
in questa fase finale della mia carriera universitaria.

Sommario

Scopo di questa tesi è quello di valutare se alcuni algoritmi di apprendimento automatico operanti su dati di scattering elettromagnetico possano rilevare e classificare l'ictus cerebrale in funzione della posizione e delle dimensioni che esso assume nello spazio rappresentato dal cervello.
Questo lavoro si focalizza sull'utilizzo di matrici di scattering usate nell'imaging a microonde, un campo della bioimaging che se applicato all'ambito dell'ictus cerebrale ha lo scopo di migliorarne la prognosi dopo l'insorgenza dell'ictus, velocizzare la diagnosi preospedaliera del tipo di ictus (ischemico o emorragico) [1] e monitorarne in modo continuo l'evoluzione nella fase post-acuta.

La Microwave Imaging (MWI), infatti, a differenza delle tecniche per ottenere immagini diagnostiche attualmente in uso, come la Tomografia Computerizzata (TC) basata sui raggi X e l'imaging a risonanza magnetica (MRI)[2]:

- sfrutta il fatto che nella banda delle microonde i tessuti ischemici ed emorragici mostrano proprietà dielettriche diverse rispetto ai tessuti sani;
- non utilizza radiazioni ionizzanti, quindi non sono dannose per il paziente;
- è una tecnica a basso costo e a bassa potenza [3].

L'oggetto di studio sono quindi i segnali di Scattering generati, attraverso la tecnologia a microonde, da un sistema di 24 antenne in modalità Tx/Rx [4], e acquisiti mediante Vector Network Analyzer (VNA).
Tali segnali sono poi usati per l'allenamento di algoritmi di Machine Learning (ML) con lo scopo di classificare la presenza dell'ictus o l'assenza, la posizione e la dimensione. È noto che l'allenamento di questi algoritmi richiede un numero molto elevato di campioni (dataset).
Poiché la costruzione del dataset ideale avrebbe richiesto molto tempo per realizzare un numero elevato di misure sperimentali, per un primo caso di studio volto a validare la metodologia sono stati simulati, tramite Matlab, i segnali di Scattering.

Nel seguente elaborato viene descritto il metodo di simulazione basato sulla procedura inversa dell'algoritmo Truncated Singular Value Decomposition (TSVD) [5], normalmente viene utilizzato come algoritmo di ricostruzione di immagini. Viene inoltre dimostrato come la procedura funziona correttamente, confrontando un caso reale di misura dei parametri S (scena con ictus di forma sferica collocato in una precisa posizione nello spazio), con quella ricreata attraverso Matlab. In particolare, mediante la manipolazione dei dati della regione di interesse (discretizzata con dei tetraedri), è stata creata, mediante una maschera, una variazione di contrasto ΔX tale per cui attraverso la procedura inversa dell'algoritmo TSVD si è ottenuta la corrispondente matrice di scattering ΔS, che è servita per verificarne il confronto.

Con la simulazione di un numero elevato di segnali, relative a varie scene con e senza ictus, si è quindi costruito il dataset per allenare e testare gli algoritmi di ML.
Sono stati utilizzati tre algoritmi supervisionati:

- Support Vector Machine (SVM)
- Multilayer Perceptron (MLP)
- k-Nearest Neighbors (k-NN)

In prima analisi è stato testato un dataset ridotto di 600 features divise nelle loro classi di appartenenza; in particolare, è stato effettuato un preprocessing iniziale con feature standardization e feature selection per rendere migliore il processo di addestramento da parte degli algoritmi di ML, al fine di ottenere una buona accuratezza nella classificazione. Con questa analisi preliminare, è stata ottenuta una buona classificazione con gli algoritmi MLP e SVM:

- SVM con un'accuratezza del 75 % e Micro Average della Curva ROC con un Area Under Curve (AUC) dello 0.95.
- MLP con un'accuratezza del 70.8 % e Micro Average della Curva ROC con un AUC dello 0.97.

Per migliorare il processo di apprendimento, è stato poi costruito un dataset più ampio con 12000 features, ottenendo questa volta un'ottima classificazione per tutt'e tre gli algoritmi.

Keywords: Microwave Imaging, diagnosi preospedaliera, monitoraggio, ictus cerebrale, algoritmo TSVD, classificazione, rilevazione, Machine Learning, MLP, SVM, k-NN.

Indice

Elenco delle tabelle

Elenco delle figure

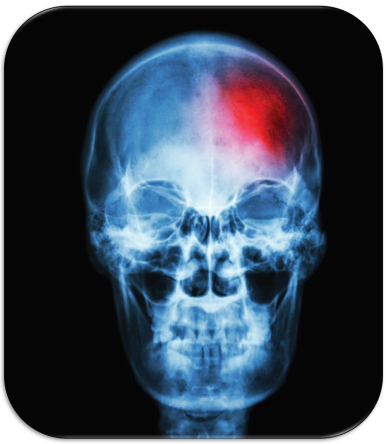
Capitolo 1

Introduzione

L'ictus Cerebrale

L'ictus cerebrale è una delle principali potenziali cause di morte o di lesione permanente nel mondo, con una prevalenza molto alta e un'incidenza che va oltre i 5 Milioni di morti all'anno; negli Stati Uniti ogni anno circa 795000 persone hanno un nuovo o ricorrente attacco di ictus, di cui l'87% è ischemico, mentre il rimanente è emorragico [6].

In Italia, in base ai dati del Ministero della Salute [7], l'ictus sarebbe responsabile del 10-12% delle morti annuali, inoltre, è la prima causa di invalidità. A livello mondiale, l'ictus risulta la seconda causa di morte dopo le malattie di cuore e prima del cancro.
Il verificarsi di un ictus comporta una forte riduzione o un'interruzione dell'afflusso di sangue ossigenato, causandone un danno all'area del tessuto cerebrale interessato o nella peggiore dei casi, la sua necrosi (o morte) [8].
L'ictus si contraddistingue per l'insorgenza improvvisa; tale caratteristica lo rende un'emergenza medica, il che ne comporta la necessità di un intervento terapeutico immediato. Infatti, è possibili limitare le conseguenze dell'evento cerebrovascolare e sperare in un recupero quanto meno parziale se il soccorso è fatto tempestivamente.



I segni e i sintomi di un ictus possono comprendere l'incapacità di muoversi o di percepire un lato del corpo, problemi alla comprensione o all'esprimere parole o la perdita di visione di una parte del campo visivo. Gli ictus emorragici possono essere associati ad un

forte mal di testa. Se i sintomi durano meno di una o due ore, l'episodio viene chiamato attacco ischemico transitorio (TIA), altrimenti, i sintomi possono essere permanenti.

L'avanzare dell'età è un rilevante fattore di rischio di ictus, mentre, il sesso mette in evidenzia che è più diffuso nella popolazione femminile rispetto a quella maschile. Esiste una spiccata tendenza, per alcune popolazioni, di soffrire di diabete, malattie vascolari e cardiovascolari, questo potrebbe suggerire che l'etnia può essere un fattore che influenza l'insorgere di ictus [8].
Ci sono inoltre altri fattori di rischio che ne favoriscono il manifestarsi di ictus come: l'ipertensione, l'aterosclerosi, il fumo di sigaretta e l'abuso di alcol.

L'ictus cerebrale, quindi, può essere ischemico (ostruzione di un vaso sanguigno che porta il sangue al cervello) o emorragico (causato da un sanguinamento e quindi da emorragia cerebrale), e la sua classificazione, tramite le tecniche di imaging convenzionali utilizzate per la diagnostica, è molto importante per stabilirne il tipo di trattamento. In particolare, la maggior parte delle tecniche di imaging per la diagnosi e il trattamento dell'ictus sfruttano le onde elettromagnetiche in una regione specifica dello spettro elettromagnetico, figura (1.1).

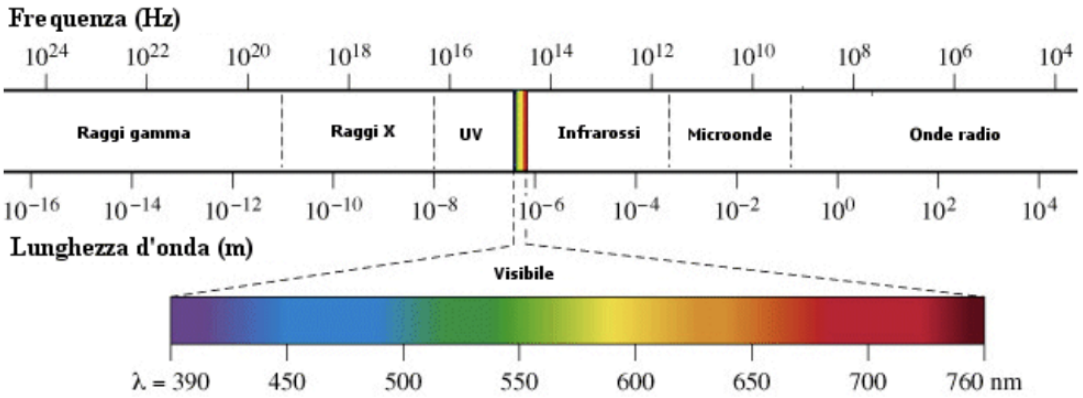


Figura 1.1: Spettro Elettromagnetico [9].

Esistono due importanti tipi di radiazione, ionizzante e non ionizzante.

Le radiazioni ionizzanti sono particolari radiazioni elettromagnetiche o particelle subatomiche dotate di una certa quantità di energia sufficiente per ionizzare la materia che attraversano, generandone particelle elettricamente cariche.
L'esposizione umana a tali radiazioni può comportare a danni (somatici o genetici) o conseguenze per la salute anche gravi nel caso di lunga esposizione.
I raggi gamma, i raggi X e la porzione ad alta frequenza degli ultravioletti dello

spettro EM sono ionizzanti. Le particelle subatomiche ionizzanti più comuni comprendono le particelle alfa, le particelle beta e i neutroni.

La parte più bassa degli ultravioletti dello spettro elettromagnetico, figura (1.1), e anche la parte più bassa dello spettro sotto agli UV, compresa la luce visibile, gli infrarossi, le microonde e le onde radio sono considerate tutte radiazioni non ionizzanti.
Si riferiscono dunque a qualunque tipo di radiazione elettromagnetica che non trasporta sufficiente energia tale da ionizzare atomi o molecole.

1.1 Principali tecniche di imaging per la diagnosi e il trattamento dell'ictus cerebrale

La diagnosi di ictus cerebrale viene generalmente formulata attraverso l'esame clinico. Le tecniche come la Tomografia Computerizzata (TC) e la Risonanza Magnetica (RM), risultano fondamentali per confermare il sospetto clinico, escludere altre patologie, caratterizzare e quantificare le lesioni ischemiche, pianificare il trattamento.

Tomografia Computerizzata (TC)

La TC è una tecnica di indagine diagnostica per immagini che impiega una apparecchiatura a raggi X. Pertanto l'attenuazione di un fascio di raggi X che avviene quando passa attraverso una sezione corporea, consente la riproduzione di immagini tridimensionali e in sezione (tomografia) dell'anatomia, create da un'analisi generata al computer.
Il calcio nelle ossa assorbe maggiormente i raggi X, ne consegue che le ossa appaiono bianche sulla radiografia.
Il grasso e altri tessuti molli assorbono meno raggi X per cui appaiono grigi. L'aria infine, assorbe ancor meno raggi X, pertanto i polmoni sembrano neri.
Dallo spettro elettromagnetico in figura (1.1), sono quelle onde EM con lunghezza d'onda dai 0,01 ai 10 nm, e con una frequenza che va dai 3.10^{16} ai 3.10^{19} Hz, quindi sono radiazioni ionizzanti.
La risoluzione spaziale è un punto di forza di questa tecnica avendo valori molto buoni.
Attraverso la TC è possibile vedere dettagliatamente il cervello, permettendo di capire il tipo di ictus che si ha di fronte.

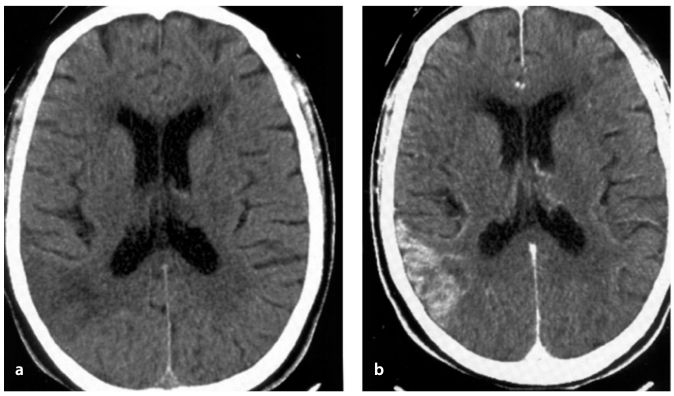


Figura 1.2: Tomografia Computerizzata dell'ictus cerebrale [10].

Se si fa uso di un liquido di contrasto (angio-TAC), si può osservare il flusso sanguigno nei vasi arteriosi e venosi del collo e dell'encefalo.

L'imaging a risonanza magnetica (MRI)

La MRI è una tecnica diagnostica per immagini basata sull'applicazione di un campo magnetico ad elevata intensità sul distretto corporeo da esaminare, che è in grado di riprodurre, attraverso l'elaborazione tomografica computerizzata dei segnali registrati dopo aver applicato un impulso a radiofrequenza, le immagini degli organi interni. Quando un forte campo magnetico viene imposto a un tessuto, i protoni delle sue molecole d'acqua sono allineati secondo la direzione del campo. Viene quindi applicato un campo elettrico di una certa energia per produrre l'assorbimento dei suoi fotoni da parte dei protoni allineati. Quando il campo elettrico viene disattivato, i protoni tornano allo stato di bassa energia ed emettono un fotone che viene catturato dallo scanner.
È possibile costruire un'immagine perché i protoni di diversi tessuti ritornano al loro stato di equilibrio a velocità diverse (risonanza), che è una differenza che può essere rilevata.
La risonanza magnetica utilizza impulsi RF comunemente nella gamma 1-100 MHz, figura (1.1), quindi questa tecnica non rientra nel campo delle radiazioni ionizzanti.
La RMI fornisce un'immagine dettagliata del cervello e individua il tessuto danneggiato da un ictus ischemico ed emorragico.

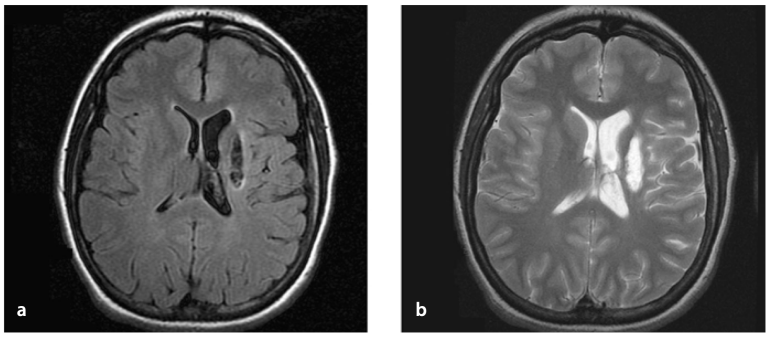


Figura 1.3: Risonanza Magnetica dell'ictus cerebrale [10].

Anche in questo caso, è possibile visualizzare il flusso sanguigno nei vasi arteriosi e venosi, usando un liquido di contrasto (angio-RM).

1.2 Problematiche delle tecniche tradizionali

Negli ultimi anni, con l'obiettivo di aumentare ulteriormente il tasso di recupero e ridurre le conseguenze degli ictus cerebrali, il monitoraggio continuo post-evento dei parametri fisiologici nella fase acuta ha acquisito un'importanza crescente. Il recupero e il trattamento, infatti, dipendono sia dalla diagnosi precoce che dall'attenta osservazione clinica, soprattutto nelle prime ore dopo l'inizio di un ictus.
Pertanto, le modalità di imaging convenzionali utilizzate per la diagnostica, come MRI e TC, sono di grande supporto al clinico nella scelta della terapia e nel seguirne l'efficacia. In tale quadro però, queste tecniche sembrano non essere adatte a causa dell'elevato costo della risonanza magnetica, della natura ionizzante della TC e del fatto che queste tecniche non sono portatili, sono disponibili solo negli ospedali e per essere utilizzate richiedono necessariamente che il paziente sia prelevato dal letto e trasferito in locali appositi dove è presente il dispositivo [11].

Ictus, emorragia intracranica e altre condizioni che influenzano la circolazione e la pressione nel cervello sono forme acute e immediatamente pericolose per la vita. Per i pazienti colpiti da ictus, prima si inizia il trattamento, maggiori sono le possibilità di guarigione; mentre, per i pazienti con trauma emorragico intracranico, il tempo che intercorre tra la lesione e l'intervento chirurgico è il fattore più importante per la sopravvivenza. Una diagnosi preospedaliera potrebbe rappresentare la differenza tra la vita o la morte e / o una vita futura senza disabilità per i pazienti. Solo una minoranza dei pazienti (1-8%) riceve cure all'avanguardia nella fase acuta, una delle ragioni è che ci sono tempi lunghi di attesa che ne ritardano la diagnosi o addirittura impediscono il trattamento [12].

Un trattamento consolidato per l'ictus ischemico è la trombolisi, cioè l'uso di un farmaco che dissolve il coagulo. Tuttavia, la diagnosi con TC o RM è necessaria prima che il farmaco possa essere somministrato perché somministrarlo a pazienti con ictus emorragico potrebbe essere letale[12]. La diagnostica preospedaliera in grado di differenziare ictus ischemico ed emorragico, nonché identificare i pazienti con occlusione di grandi vasi, sarebbe estremamente preziosa per semplificare la gestione dell'ictus.

L'obiettivo della diagnosi preospedaliera è quello di ridurre il tempo che intercorre tra il trauma e il trattamento, prevedendo i pazienti che necessitano di un intervento acuto e i pazienti che potrebbero essere trasportati in sicurezza in un centro non specializzato.

Attualmente, non esiste alcun metodo per l'uso nella pratica clinica standard per il rilevamento preospedaliero di lesioni interne. Sfortunatamente, l'accuratezza nell'identificazione dei pazienti con lesioni gravi è carente e sono necessari nuovi strumenti diagnostici per migliorare questa situazione.
Sono state studiate diverse tecnologie con l'obiettivo di sviluppare un sistema compatto che possa essere utilizzato per diagnosticare pazienti con traumi o ictus in contesti preospedalieri. Alcune importanti alternative promettenti negli studi clinici includono la spettroscopia nel vicino infrarosso NIRS, l'ecografia e l'elettroencefalografia.
Sebbene meno sviluppata nella diagnostica medica, la propagazione delle microonde nei tessuti umani ha un vantaggio sia sull'impedenza che sugli ultrasuoni grazie alla facile penetrazione del cranio umano [1].

1.3 Imaging a microonde

Un potenziale candidato per superare le limitazioni imposte dalle teniche tradizionali è la MicroWave Imaging (MWI), che ha guadagnato una crescente attenzione nelle applicazioni mediche.
Le radiazioni elettromagnetiche nella gamma di frequenza delle microonde, infatti, sono innanzitutto non ionizzanti, la tecnologia su cui si basa è alla base delle telecomunicazioni mobili, beneficiando così degli enormi progressi osservati in questo campo negli ultimi anni. Inoltre, sono particolarmente adatte per effettuare un monitoraggio continuo, poiché non sono dannose per il paziente.
Infine, la tecnologia a microonde consente di progettare dispositivi portatili, economici ed a bassa potenza. Questi vantaggi compensano in una certa misura i limiti intrinseci della MWI in termini di risoluzione spaziale ottenibile rispetto a MRI o TC [2] [11].

Capitolo 2

Microwave Imaging (MWI)

Le circostanze riportate nel precedente capitolo hanno portato ad un maggiore interesse per lo sviluppo di diverse tecniche di diagnostica per immagini. Tra le altre quindi, la MWI è emerso come una tecnica complementare in grado di affrontare le diverse esigenze che emergono nella diagnosi e nella gestione dell'ictus, vale a dire la diagnosi precoce - possibilmente preospedaliera - del tipo di ictus (ischemia o emorragia), l'imaging cerebrale direttamente al letto del paziente e il monitoraggio continuo dell'ictus nella fase post-acuta.

La MWI sfrutta le diverse proprietà elettriche (permettività elettrica e conduttività) che i tessuti umani esibiscono alle frequenze delle microonde a seconda del loro tipo e dello stato patologico. Queste differenze consentono di ottenere una mappa funzionale della regione anatomica ispezionata.
I vantaggi della MWI derivano principalmente dalla natura non ionizzante della radiazione a microonde e dalla ridotta intensità necessaria per ottenere immagini affidabili (ad un'intensità paragonabile a quella attualmente utilizzata per i telefoni cellulari), che lo rendono completamente sicuro e adatto ad applicazioni ripetute. Inoltre, la tecnologia MWI è economica e beneficia di dimensioni ridotte, poiché utilizza componenti miniaturizzati, a basso costo e pronti all'uso, disponibili nella gamma di frequenza delle microonde 0,1–10 GHz fig (1.1), per la generazione e l'acquisizione del segnale e un low-cost degli acceleratori di particelle per velocizzare l'elaborazione [5].

2.1 Principi fisici dell'imaging a microonde

Questo paragrafo fornisce un'introduzione ai principi fisici alla base dell'adozione della tecnologia a microonde come modalità di imaging biomedico per la diagnosi e

il follow-up di malattie e lesioni neurologiche.
Le onde elettromagnetiche, alle frequenze delle microonde, infatti non sono ionizzanti e quindi non dannose per l'uomo.

2.1.1 Proprietà elettriche della materia: materiali dielettrici

In generale, la materia è composta da particelle cariche che se influenzate dall'applicazione di campi esterni (campi elettrici e magnetici prodotti da altre cariche) possono essere conduttori, dielettrici e semiconduttori.
Ai fini dello studio sul cervello e quindi sul corpo umano, ci interessano i materiali dielettrici, poiché mimano bene i tessuti biologici; gli elettroni sono così ben legati che non possono muoversi attraverso il materiale, pertanto il materiale non contiene cariche libere.
Quando viene applicato un campo elettrico esterno $\underline{E}_a$, le cariche legate negative e positive si spostano leggermente di posizione l'una rispetto all'altra.
Appare un Vettore di Polarizzazione elettrica netta $\underline{P}$, parallelo al campo elettrico $\underline{E}_a$:

$$\underline{P} = \chi \varepsilon_0 \underline{E}_a \tag{2.1}$$

dove $\underline{P}$ è il Vettore Polarizzazione espresso in Cm^{-2}, χ è la suscettibilità elettrica, ε_0 è la costante dielettrica nel vuoto ($\frac{1}{36\pi}10^{-9}[\frac{F}{m}]$); si creano dunque, dei momenti di dipolo delle cariche legate.
Il vettore di Induzione elettrica $\underline{D}$ rappresenta il modo in cui un campo elettrico $\underline{E}_a$ influenza la disposizione delle cariche elettriche in un mezzo, tenendo conto della polarizzazione elettrica del materiale.

$$\underline{D} = \underline{D}_0 + \underline{P} = \varepsilon_0(1 + \chi_e)\underline{E}_a = \varepsilon \underline{E}_a \tag{2.2}$$

La Permettività Elettrica ε è una quantità che descrive le proprietà dielettriche che influenzano la riflessione delle onde EM alle interfacce e l'attenuazione dell'energia delle onde all'interno dei materiali [13]. Introducendo la Permettività Complessa ε^* si riescono a descrivere le perdite di cui un materiale è soggetto [14].

$$\varepsilon^* = \varepsilon^{'} - j\varepsilon^{''} \tag{2.3}$$

dove:

- la parte reale $\varepsilon^{'}$ è indicata come costante dielettrica; sarebbe correlata all'energia immagazzinata quando il materiale è esposto a un campo elettrico.
- La parte immaginaria $\varepsilon^{''}$ è indicata come fattore di perdita dielettrica ed è connessa all'assorbimento e all'attenuazione dell'energia.

Nei materiali dielettrici, l'intensità del campo elettrico diminuisce con la distanza z dalla superficie

$$\underline{E}(z) = E_0 e^{-(\alpha+j\beta)z} = E_0 e^{-\alpha z} e^{-j\beta z} \tag{2.4}$$

dove:

- α è la costante di attenuazione $[\frac{Nep}{m}]$
- β è la fase $[\frac{rad}{m}]$

La profondità di penetrazione d_p (molto utile nello studio di come le microonde riescano a penetrare dentro al tessuto cerebrale) è definita come la profondità alla quale la potenza è ridotta a $\frac{1}{e}$ della potenza P che entra in superficie.

$$p = p_0 e^{-2\alpha z} \tag{2.5}$$

Date le proprietà dielettriche fisse, d_p è inversamente proporzionale alla frequenza.

Il Modello di Debye, è utilizzato per spiegare il comportamento delle proprietà dielettriche di liquidi puri, come l'acqua, alle frequenze delle microonde. Il modello di rilassamento di Debye afferma che i materiali dielettrici agiscono come un circuito risonante con perdita. Quando si applica un campo elettrico a un materiale dielettrico, le molecole polari tendono ad allinearsi in base ad esso. Dal momento che, questo effetto viene contrastato dall'agitazione termica, se l'applicazione del campo elettrico viene interrotta, l'allineamento si rilassa in modo esponenziale con una costante di tempo τ tipica del materiale.
Un'approssimazione del primo ordine fornisce la seguente espressione per la Permettività Complessa ε^* del tessuto in funzione della frequenza angolare w [15].

$$\varepsilon^*(w) = \varepsilon^{'}(w) - j\varepsilon^{''}(w) = \varepsilon_\infty + \frac{\varepsilon_s - \varepsilon_\infty}{1 + jw\tau} \tag{2.6}$$

dove:

- ε_∞ è la costante dielettrica nel campo delle frequenze dove $w\tau >> 1$
- ε_s è la la costante dielettrica statica nel campo delle frequenze dove $w\tau << 1$

Poiché la complessità sia della struttura che della composizione del materiale biologico richiede di tenere conto anche della dispersione dovuta alla costante di rilassamento del materiale α, si fornisce così un'alternativa all'equazione (2.6) di Debye nota come equazione di Cole-Cole.

$$\varepsilon^* = \varepsilon_\infty + \frac{\varepsilon_s - \varepsilon_\infty}{(1 + jw\tau)^{(1-\alpha)}} + \frac{\sigma_i}{jw\varepsilon_0} \tag{2.7}$$

Considerato che, le proprietà dielettriche sono altamente correlate al suo contenuto d'acqua, il meccanismo principale per l'interazione tra onde elettromagnetiche e

tessuti biologici è l'oscillazione di molecole di acqua polare e ioni.
I tessuti biologici sono divisi in 2 gruppi principali a seconda del contenuto di acqua [16]:

- Elevato contenuto di acqua: muscoli (73-78%), fegato (75-77%), reni (76-78%), cervello (68-73%), pelle (60-76%), polmone (80-83%);
- Basso contenuto di acqua: grasso (5-10%), ossa (8-16%).

Ci si aspetta, quindi, un alto contrasto risultante dalle proprietà dielettriche dei tessuti, che ci consente lo studio attraverso l'applicazione di onde EM.

2.1.2 Metodi di misurazione delle proprietà dielettriche

Nella misurazione delle proprietà dielettriche, è importante innanzitutto definire quali sono i mezzi che consentono un trasferimento di energia nella maniera ottimale, e quindi una misurazione della stessa.

In generale, l'energia elettromagnetica tende a diffondersi in tutto lo spazio ad una velocità prossima ai 300.000 $\frac{Km}{s}$.

Tale energia, idealmente si desidera che sia trasferita da un luogo all'altro lungo un percorso ben definito senza alcuna diffusione. Una linea di trasmissione è un sistema di conduttori metallici e/o mezzi isolanti dielettrici in grado di "guidare" il trasferimento di energia da un generatore a un carico, indipendentemente dal percorso realizzato come cablaggio. Su una linea di trasmissione, quindi si verifica un fenomeno di propagazione unidimensionale.
Esistono molti tipi di linee di trasmissione, figura (2.1), alcuni dei quali sono utilizzati per diverse applicazioni in specifici intervalli di frequenza. Ad esempio stripline e microstrip sono utilizzati solo all'interno di dispositivi, come amplificatori o filtri, e la loro lunghezza non supera mai alcuni centimetri. Per il cablaggio di un edificio vengono utilizzati doppini intrecciati e cavi coassiali, ma i cavi coassiali possono essere utilizzati anche per le comunicazioni intercontinentali. I tubi cavi metallici, noti come guide d'onda, vengono utilizzati per fornire grandi quantità di potenza a microonde su breve e lunghe distanze.

Le guide d'onda possono anche essere realizzate solo con materiali dielettrici, come nel caso delle fibre ottiche. In generale si considerano solo strutture realizzate da due conduttori metallici, come cavi coassiali, microstrip e stripline. Queste possono essere definite linee di trasmissione in senso stretto, mentre le altre sono più appropriatamente chiamate guide d'onda metalliche o dielettriche.
Le guide d'onda, in generale, hanno una frequenza di funzionamento più bassa, che dipende dalla loro dimensione trasversale.
Viene definita cos'è una modalità trasversale di radiazione elettromagnetica, che è appunto un particolare modello di campo elettromagnetico per il quale sia il

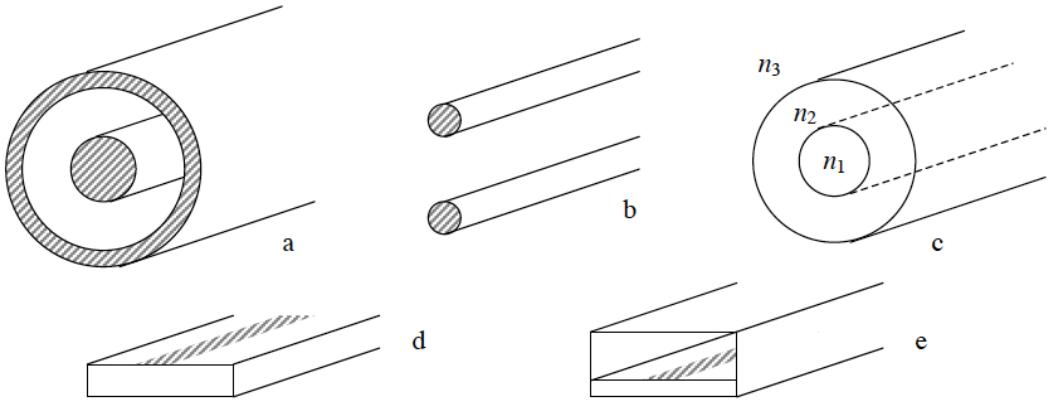


Figura 2.1: Esempi di linee di trasmissione: a) cavo coassiale; b) two wire line; c) fibra ottica; d) microstrip; e) stripline [17].

campo elettrico che quello magnetico viaggiano in direzione perpendicolare (cioè trasversale) rispetto alla direzione di propagazione della radiazione.
In conclusione, le linee di trasmissione sono guide d'onda il cui comportamento, a frequenza sufficientemente bassa, è correlato al solo modo TEM (trasverse electromagnetic) [17].

Prima di introdurre i principali metodi di misurazione delle proprietà dielettriche, è importante sottolineare quali sono i sistemi di misurazione che cambiano in base alla frequenza in gioco.
A bassa frequenza (statica) le misurazioni avvengono con sistemi di misurazione come amperometro e voltmetro, strumenti che riguardano le misure di tensione e corrente; per poi calcolarne semplicemente l'impedenza $Z = \frac{V}{I}$.
Ad alta frequenza cambiano gli strumenti utilizzati, in particolare, nella misurazione nel range delle microonde si utilizza il Vector Network Analyzer (VNA).

Vector Network Analyzer (VNA) e Principi base parametri S

Da un semplice punto di vista, il VNA può essere inteso come un analizzatore di impedenza, ma con più funzionalità. È in grado di misurare a radiofrequenza e microonde, utilizzando metodi più complessi. Per misurare i fattori di impedenza o riflessione, un generatore sinusoidale stimola il campione, mentre le tensioni e le correnti vengono acquisite dai ricevitori. Utilizzando quindi misure della fase incidente e riflessa si ottengono quantità di segnali di magnitudo, rendendo possibile la ricostruzione delle proprietà dielettriche del campione. È importante sottolineare la necessità di un processo di calibrazione per evitare errori nella raccolta dei parametri di scattering.

Trovare i parametri di scattering è l'obiettivo principale del VNA. Questi valori descrivono la relazione tra magnitudine e fase delle onde incidente e riflessa. Le equazioni che collegano i parametri alle onde sono riportate nell'equazione (2.8) (2.9), a seconda della porta considerata.

$$b_1 = S_{11}a_1 + S_{12}a_2 \tag{2.8}$$

$$b_2 = S_{21}a_1 + S_{22}a_2 \tag{2.9}$$

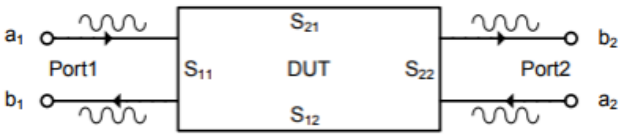


Figura 2.2: Esempio di un Quadripolo e i rispettivi parametri S

Considerando un'onda riflessa da un dispositivo come una combinazione lineare di onde incidenti a_1 o a_2, i parametri di scattering possono essere ottenuti imponendo una terminazione, solitamente intorno a 50 ω, cioè l'impedenza caratteristica Z_0, prima su una porta e poi sull'altra:

$$\begin{cases} S_{11} = \frac{b_1}{a_1}|_{a_2=0} \\ S_{12} = \frac{b_1}{a_2}|_{a_1=0} \\ S_{21} = \frac{b_2}{a_1}|_{a_2=0} \\ S_{22} = \frac{b_2}{a_2}|_{a_1=0} \end{cases} \tag{2.10}$$

I pedici m e n del parametro di scattering S_{mn} indicano rispettivamente il numero della porta del ricevitore e quello in trasmissione. Quindi è possibile scrivere tutti i parametri S all'interno di una matrice dove ogni pedice viene utilizzato come indirizzo di matrice.

$$\begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \tag{2.11}$$

La misurazione delle proprietà elettromagnetiche alle frequenze delle microonde è fondamentale per creare un modello realistico (chiamato phantom) e per definire il miglior materiale accoppiato da utilizzare.
Generalmente le tecniche utilizzate nella misurazione delle proprietà dielettriche sono Transmission line, Resonant cavity e Open-ended coaxial probe.

Open-ended Coaxial Probe (OCP)

É una tecnica utilizzata per la misurazione delle caratteristiche dielettriche dei materiali biologici e non solo.
Il metodo della sonda coassiale consiste nel mettere in contatto la regione della sonda terminale con il campione.
Quindi tramite la sonda un campo elettromagnetico, con parametro di scattering S_{11}, viene acquisito per ottenere la costante dielettrica. Il sistema è composto da un VNA, sonda coassiale e software ad-hoc per il calcolo dei parametri dielettrici.
La sonda deve essere calibrata per evitare errori dovuti alle diverse impedenze.
Le misure non sono sufficientemente accurate e possono essere eseguite solo misure di riflessione.
La misurazione richiede poca preparazione del campione, metodo non distruttivo.

Transimission-line method (TLM)

In questo metodo il campione viene inserito all'interno della linea di trasmissione, generalmente a sezione rettangolare, dove si propaga solo la modalità TEM.
S_{11} e S_{21} vengono quindi misurati e utilizzati per ottenere la costante dielettrica e le perdite. Questo tipo di sistema è caratterizzato da un VNA, linee coassiali o una sezione di guida d'onda, un computer e un software.
Vengono utilizzati 3 tipi principali di linee di trasmissione: guida d'onda rettangolare, linea coassiale e linea a microstrip, figura (2.1). Ha le seguenti caratteristiche:

- Ancora ampiamente utilizzato nell'ingegneria delle misure a microonde;
- Semplicità;
- Più costoso per la stessa gamma di frequenza rispetto all'OCP. Requisiti rigidi su forme e dimensioni del campione;
- Fornisce una buona precisione per materiali ad alta perdita tra quella dell'OCP e quella del metodo della cavità di risonanza;
- Procedura di misura: il campione da analizzare viene posto vicino all'estremità cortocircuitata della linea di trasmissione.

Resonant cavity method

Questo metodo si basa sulla creazione di una perturbazione all'interno della cavità. Il campione influenza la frequenza di risonanza principale e il fattore di qualità della cavità. Da queste quantità è possibile calcolare la permettività o permeabilità complessa per una singola frequenza.

- Ampiamente usato nell'ingegneria di misura a microonde;

- Più accurato dei metodi di guida d'onda. Particolarmente indicato per materiali e sostanze a media e bassa perdita;
- I campioni devono essere modellati con precisione e di piccole dimensioni, metodo distruttivo. Fornisce proprietà dielettriche solo a una frequenza fissa. Più costoso di OCP.

Tutti questi metodi di misura saranno utilizzati per determinare le proprietà dielettriche del cervello.

2.1.3 Proprietà dielettriche del cervello

Il crescente sviluppo dei modelli del corpo umano e degli studi riguardanti le proprietà dielettriche dei tessuti, hanno messo in evidenza la possibilità di sfruttare il contrasto nelle caratteristiche fra i vari tessuti per ottenere immagini in modo non invasivo. Infatti, proprio nel range di frequenze delle microonde [18], la permettività e la conduttività dei tessuti varia considerevolmente dipendendo dal contenuto di acqua, grasso e proteine. Per tale motivo la base teorica dell'imaging a microonde si basa sull'alta differenza di contrasto fra permettività e conduttività dei tessuti sani con quelli maligni o con lesione.

Le proprietà dielettriche del cervello sono sempre state oggetto di studio, si è scoperto che la materia grigia ha una permettività e una conduttività maggiori rispetto alla materia bianca.
Nel sangue, sia la permettività che la conduttività del sangue sono superiori a quelle per la materia grigia e bianca (ad esempio, a 1 GHz, i valori di permettività e conduttività per il sangue sono circa il 20% più alti per la materia grigia e superiore del 60% rispetto alla materia bianca). Invece, Il liquido cerebrospinale (CSF) che si trova nei ventricoli del cervello, tra il cervello e il cranio e che circonda anche il midollo spinale, essendo un liquido comporta delle forti perdite, con una maggiore permettività e conduttività rispetto al sangue [12].
Per riassumere, la permettività dielettrica e la conduttività per la materia grigia e bianca, il liquido cerebrospinale e il sangue riprodotti sono mostrate nella figura (2.3) utilizzando il modello di Cole-Cole (2.7). Malattie e lesioni potrebbero, di per sé, portare a cambiamenti nelle proprietà dielettriche, come accade nel quadro clinico di un ictus emorragico, dove il sangue inizia immediatamente a coagulare dopo l'inizio del sanguinamento. Invece, nel caso clinico di ictus ischemico, le proprietà dielettriche cambieranno a causa della forte perdita di circolazione, favorendone una diminuzione dei parametri dielettrici, in particolare, a 1 GHz la permettività relativa è circa 36 e la conducibilità elettrica è di circa 0.72 $[\frac{S}{m}]$ [11].

L'obbiettivo della MWI è di ricostruire la distribuzione della permettività complessa all'interno del tessuto cerebrale a partire da un set di dati misurati del volume

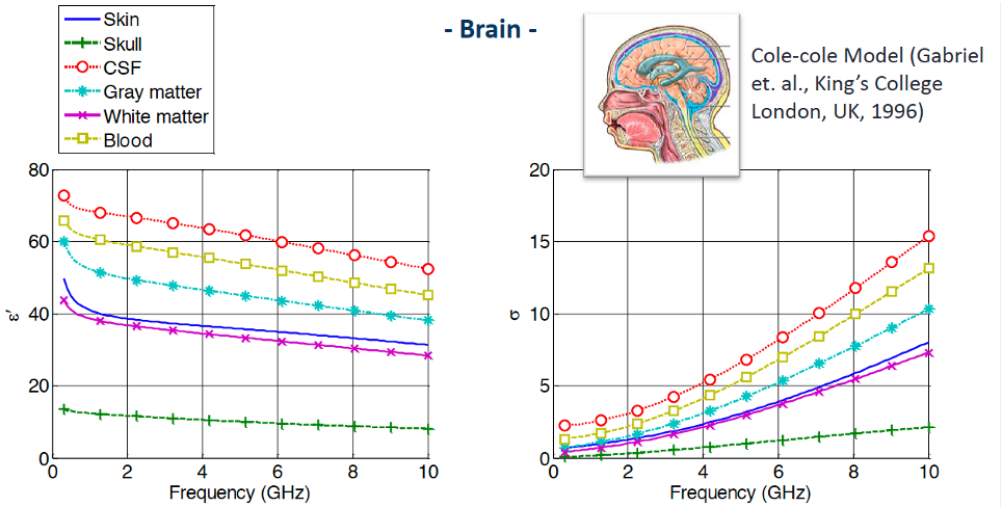


Figura 2.3: Parametri dielettrici del tessuto cerebrale umano, [12]

analizzato. Generalmente i sistemi sono costituiti da un certo numero di antenne disposte in modo da circondare la testa, che possono funzionare sia da ricevitori che da trasmettitori.

La radiazione elettromagnetica viene inviata ai tessuti e il campo diffuso, che dipende dalla morfologia del target e dal contrasto tra i parametri elettrici (permettività e conduttività) del target e il mezzo circostante al target, viene misurato per ottenere immagini della distribuzione spaziale della permettività e della conduttività [11]. La MWI si concentra sul fenomeno elettromagnetico dello scattering, che accade quando un campo elettromagnetico interagisce con il corpo del materiale; il tutto si basa sulla risoluzione di un problema di scattering inverso non lineare e mal-condizionato. La soluzione a tale problema è ottenuta tramite una procedura di ottimizzazione, cercando di minimizzare la differenza fra i dati misurati e quelli calcolati nel modello oppure linearizzando il problema con l'algoritmo TSVD (Born Approximation).

2.2 Premesse sulla fase di progettazione del sistema MWI

Un ruolo cruciale nella progettazione di un sistema MWI è svolto dalla scelta della frequenza di lavoro oltre che dalla scelta del mezzo di accoppiamento. L'adozione di un modello semplice come la guida d'onda negli studi della MWI per il cancro al seno [19], si è basata sull'osservazione che la lunghezza d'onda delle sonde deve essere piccola rispetto alla dimensione del seno, al fine di consentire la

detection di piccoli tumori e dettagli dei tessuti. Si è difatti tratto vantaggio da uno strumento simile anche nel quadro attuale dell'imaging dell'ictus. In effetti, condizioni simili si verificano nel monitoraggio dell'ictus cerebrale, poiché si prevede che l'estensione dell'area danneggiata sia molto più piccola della testa. In base a questi ragionamenti, in precedenti studi, la testa è stata modellata mediante una sequenza di cinque strati planari aventi caratteristiche dielettriche e lunghezza differenti [2][11]. Il primo strato è la pelle (con spessore di 4 mm), il secondo e il terzo strato sono rispettivamente il grasso (4 mm) e l'osso corticale (7 mm). Il quarto strato rappresenta il liquido cerebrospinale (CSF) di spessore 3 mm. Infine, la regione del cervello, cioè la regione in cui si vuole garantire la massima penetrazione del campo in arrivo, è modellata come mezzo spazio. Un modo conveniente per studiare una tale struttura è sfruttare il formalismo della linea di trasmissione (TL), che equivale a considerare una linea di trasmissione equivalente in cui ogni sezione corrisponde a uno strato di tessuto, come in figura (2.3). In un tale modello, $Z_{mm}, Z_s, Z_f, Z_b, Z_{csf}, Z_{br}$ denotano le impedenze caratteristiche del mezzo corrispondente.

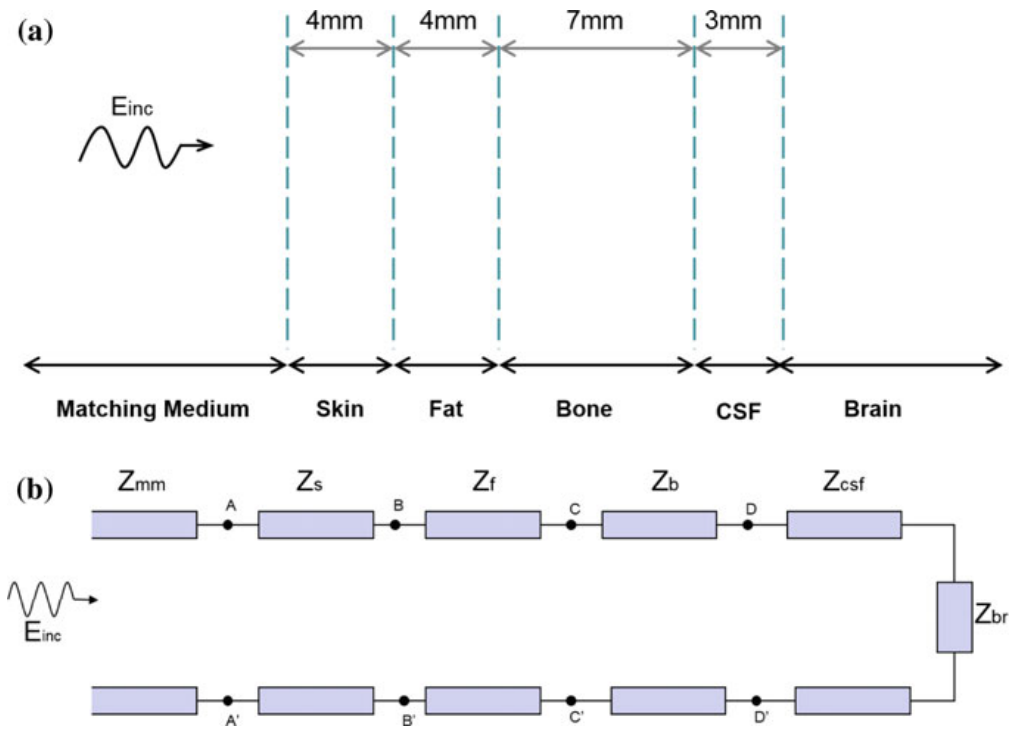


Figura 2.4: (a)Modello planare a strati della testa; (b)corrispondente TL, [2] [11].

Per tenere conto del comportamento dispersivo in frequenza dei tessuti biologici, le loro proprietà elettriche sono state valutate secondo il modello di Cole-Cole

(2.7); sono stati quindi impostati [20] i seguenti parametri riportati nella tabella (2.1). Poiché il cervello è composto principalmente da materia bianca e grigia,

Paramatri Cole-Cole					
Tessuto	ε_∞	$\Delta\varepsilon$	$\tau[s]$	α	$\sigma_i[\frac{S}{m}]$
Dry Skin	4	32	$7.23e^{-12}$	0	0.0002
Fat	2.5	3.0	$7.96e^{-12}$	0.2	0.001
Bone	2.5	10	$13.26e^{-12}$	0.2	0.02
CSF	4	65	$7.96e^{-12}$	0.1	2
Grey Matter	4	45	$7.96e^{-12}$	0.1	0.02
White Matter	4	32	$7.96e^{-12}$	0.1	0.02

Tabella 2.1: Paramatri Cole-Cole dei tessuti del cranio

la permettività complessa dell'ultimo strato è data dalla media delle costante dielettrica complessa di questi due tessuti. Sfruttando questo modello equivalente si è calcolata la trasmittanza T, che permette di stimare la quantità di potenza entrante che penetra nella testa. Questo coefficiente è definito come:

$$T = 1 - \Gamma \tag{2.12}$$

Dove Γ è il coefficiente di riflessione della sezione AA', definito come:

$$\Gamma = \frac{Z_{AA'} - Z_{mm}}{Z_{AA'} + Z_{mm}} \tag{2.13}$$

Con Z_{mm}, l'impedenza caratteristica del mezzo di accoppiamento. Viene quindi calcolato il coefficiente di trasmissione in funzione della frequenza, nella banda di microonde di interesse (da 100 MHz a 10 GHz), e della costante dielettrica del mezzo di accoppiamento, che è stata supposta sia compresa tra aria e acqua ε_{mm} in un range da 1 a 80.

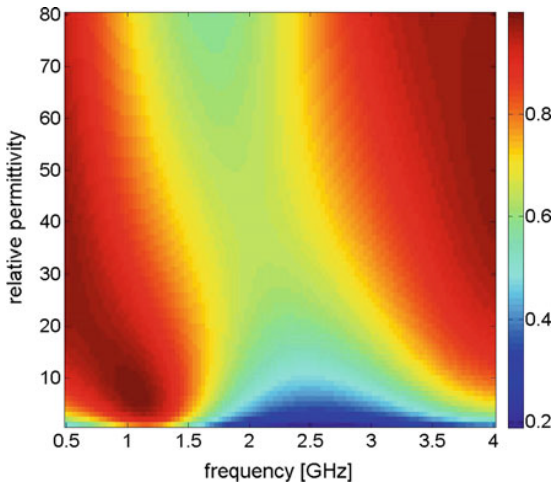


Figura 2.5: Trasmittanza in funzione della frequenza e della costante dielettrica complessa del mezzo di accoppiamento senza perdite, [2] [11].

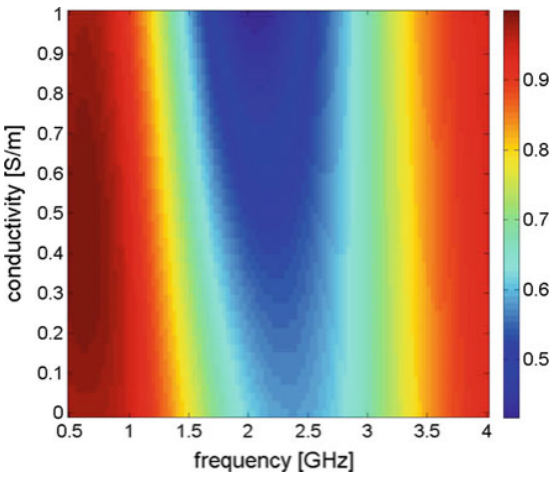


Figura 2.6: Trasmittanza in funzione della frequenza e della conduttività del mezzo di accoppiamento, assumendo $\varepsilon_{mm} = 40$, [2] [11].

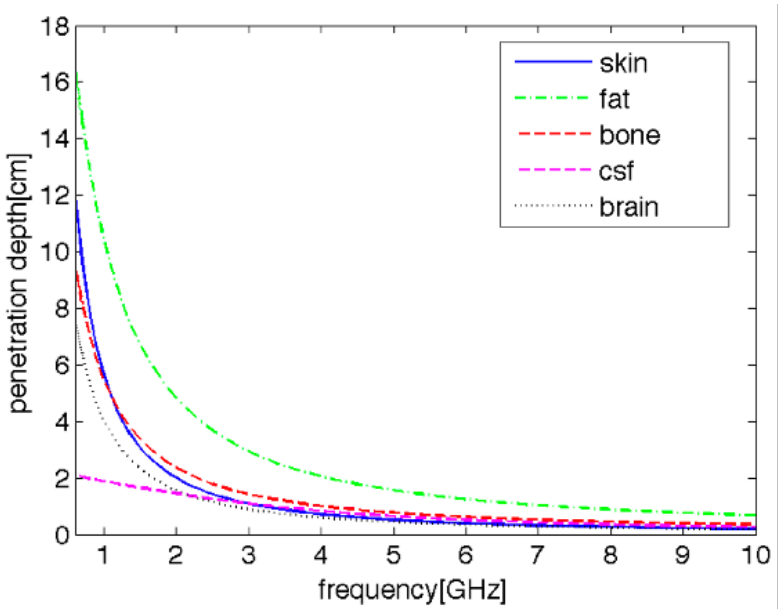


Figura 2.7: Profondità di penetrazione per i tessuti della testa [2] [11].

Tenendo conto delle figure (2.5) e (2.6), è stata identificata la gamma da 0.6 a 1.5 GHz come la banda di frequenza più conveniente. Inoltre, rispetto a tale intervallo

di frequenza, dalla Fig (2.5) possiamo supporre che la costante dielettrica del mezzo di adattamento può essere scelta quasi arbitrariamente come appartenente alla gamma 10 a 40. In figura (2.7) si può notare come dalle frequenze 0.6 a 1.5 GHz la profondità di penetrazione è migliore ai fini dell'imaging.

2.3 Sistemi 3D di imaging a microonde

I primi dispositivi di imaging 2D, adottavano come mezzo di accoppiamento una miscela di Triton X-100 (TX-100, un tensioattivo non ionico) e acqua con percentuale in volume di 70/30, principalmente perché si era dimostrata molto stabile nel tempo e facile da preparare. Inoltre rappresenta un buon compromesso tra la risoluzione dell'immagine e l'attenuazione dei campi elettromagnetici causata dalle perdite. Ogni antenna del sistema di imaging a microonde era a banda larga unipolare, stampata su un substrato dielettrico standard FR4 con spessore pari a 1.6 mm, dove il lato superiore era costituito da una linea di trasmissione con circuito di accoppiamento a doppio stub e terminava con una parte radiante di forma triangolare; il lato inferiore era costituito da un piano di massa che termina vicino all'inizio del triangolo radiante [21].
L'ampiezza del coefficiente di riflessione in ingresso delle antenne era sempre inferiore a $-15dB$ su tutto il range di frequenza (da 1 a 1.75 GHz). Il phantom 2D era realizzato da una cavità cilindrica circolare del diametro di 10 cm, alta 16 cm, realizzata in acrilonitrile butadiene stirene (ABS) e chiusa da un tappo con tre fori dove potevano essere inseriti cilindri in ABS più piccoli. Questo phantom aveva lo scopo di eseguire misurazioni 2D preliminari e, come tale, aveva una dimensione (l'altezza) dominante rispetto all'altra (diametro), non tenendo conto delle dimensioni effettive di una testa umana.

Il sistema proposto invece, fornisce immagini 3D, della testa basandosi sui dati misurati attraverso una serie di 24 antenne organizzate anatomicamente attorno alla testa in modo da imitarne un casco indossabile e adattabile. Ogni antenna è racchiusa in una scatola di materiale grafite-silicio, che funge da mezzo di accoppiamento, e collegata a un VNA a due porte tramite una matrice di commutazione 24 x 24, che consente l'acquisizione dell'intera matrice di scattering differenziale necessaria per l'imaging.
Il phantom è costituito da un guscio di plastica della forma e delle dimensioni di una testa umana, riempito con un materiale omogeneo le cui proprietà dielettriche sono pari al valore medio delle proprietà dei diversi tessuti presenti nel cervello [22]. In particolare la cavità del phantom è riempita con un liquido mixato con Triton X-100 e acqua (a diverse proporzioni) che mima il cervello, modellato da una miscela che ne simula la materia bianca e la materia grigia nel caso di ictus viene riempita una pallina con un'altra miscela che ne rappresenta le caratteristiche dielettriche

del sangue, figura (2.8).

In questo paragrafo, sono presentate le scelte fatte nella progettazione del sistema

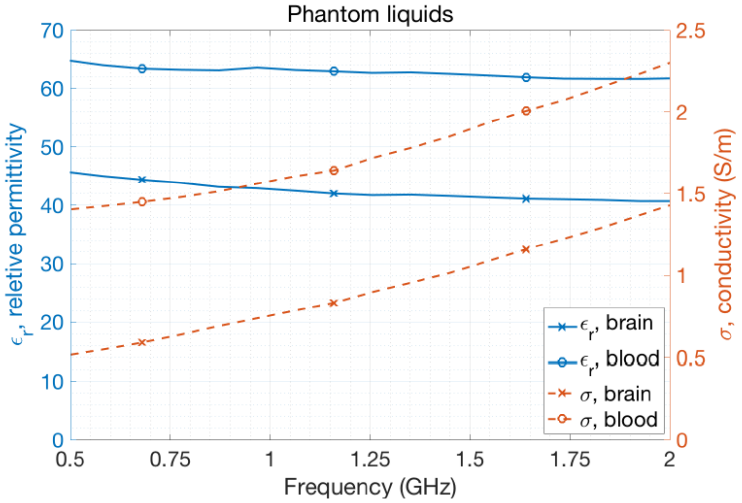


Figura 2.8: Proprietà dielettriche delle miscele che mimano il tessuto cerebrale misurate con VNA nel range di frequenza operativa da 0.5 a 2 GHz [21].

di imaging proposto (cioè, la frequenza operativa, il mezzo di accoppiamento, il numero e la disposizione dell'antenna). La frequenza operativa e le proprietà del mezzo di accoppiamento selezionato sono state impostate in base a precedenti risultati nel paragrafo precedente relativo all'adozione del modello con TL. La scelta della gamma di frequenze di lavoro e le caratteristiche dielettriche del mezzo di accoppiamento sono strettamente correlate all'uso finale del sistema di imaging a microonde realizzato. Il comportamento in frequenza del coefficiente di trasmissione dipende leggermente dalle caratteristiche dielettriche del mezzo di accoppiamento; in particolare, la frequenza di partenza della banda proibita è leggermente superiore se la sua permettività relativa è intorno a 20 o inferiore. Inoltre, lavorare a frequenze superiori a 2.5 GHz potrebbe essere possibile ma non conveniente, a causa della bassa profondità di penetrazione nei tessuti ispezionati. Quindi, la frequenza di lavoro prescelta per il progetto dell'antenna e gli algoritmi di ricostruzione dell'immagine è 1 GHz, con una permettività relativa del mezzo di accoppiamento di circa 20. Il numero degli elementi radianti è stato identificato è pari a 24 elementi come il candidato adatto per eseguire l'attività di imaging desiderata mantenendo la complessità del sistema più bassa possibile [4]. Il prototipo del sistema di imaging a microonde 3D realizzato è mostrato nella figura (2.9). Lo schema a blocchi nella Figura (2.10) mostra le cinque parti principali che compongono il sistema MI per il monitoraggio dell'ictus cerebrale [3]:

Figura 2.9: Prototipo del sistema di imaging a microonde 3D [5].

I. un casco contenente le antenne;

II. una matrice di commutazione incaricata di instradare i segnali RF trasmessi e ricevuti;

III. un VNA per la trasmissione e la ricezione dei segnali;

IV. un hardware dedicato che consente l'elaborazione dei dati acquisiti;

V. un display che mostra l'immagine ricostruita.

Il casco è realizzato in materiale plastico, nel prototipo mostrato in figura (2.11), le 24 antenne sono posizionate su un supporto in plastica stampata in 3D (acrilonitrile butadiene stirene, ABS) con la forma di un casco conforme al phantom della testa. Questa configurazione consente di cambiare o rimuovere facilmente le antenne, se necessario. Il brick dielettrico è stato realizzato con una miscela di gomma uretanica e polvere di grafite ed è stato progettato per raggiungere una permettività dielettrica relativa di $\varepsilon_r = 20$ e per ridurre al minimo le perdite. Le antenne possono funzionare sia come trasmettitori che come ricevitori. La matrice di commutazione, infatti, è stata progettata per instradare il segnale generato dal VNA verso un'unica antenna mantenendo tutte le altre come ricevitori. Poiché il VNA ha due porte

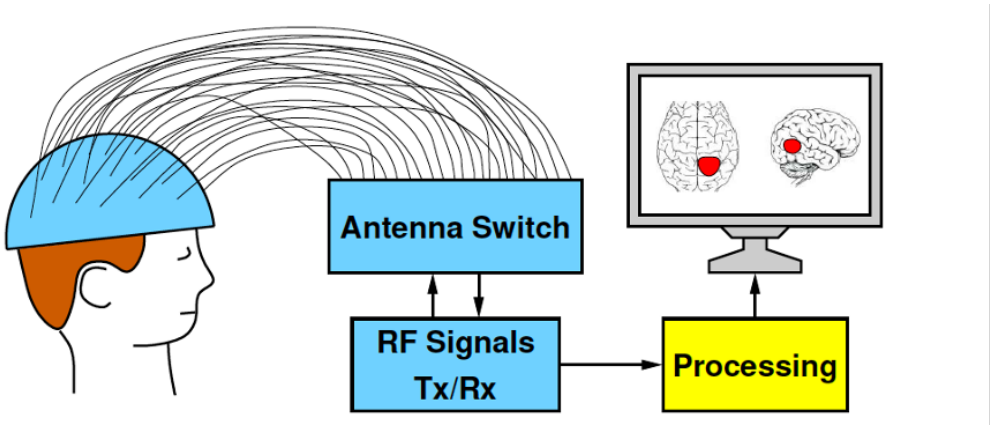


Figura 2.10: Schema imaging a microonde 3D [3].

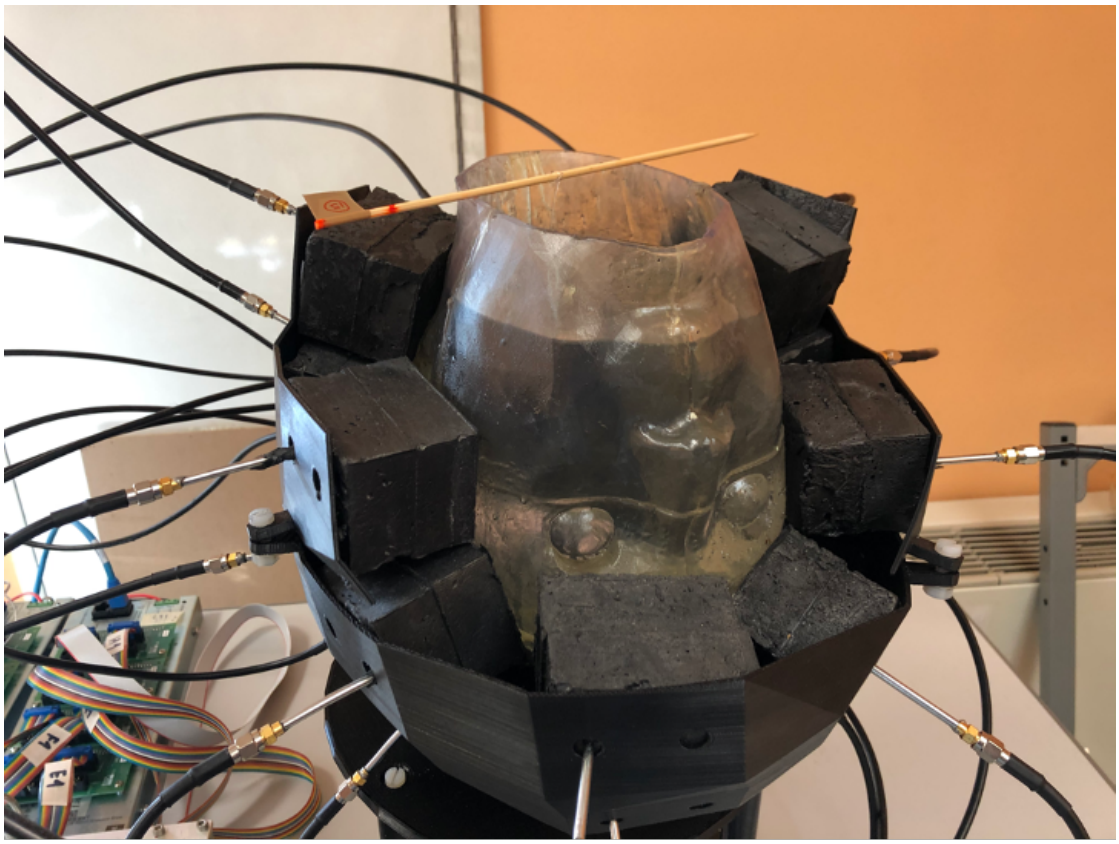


Figura 2.11: Brick di array di Antenne conforme al phantom della testa [5].

(una di trasmissione e una di ricezione), un controller configura la matrice di commutazione in modo tale che solo un trasmettitore e un ricevitore possono

essere contemporaneamente attivi e collegati al VNA. Dal rapporto tra i segnali ricevuti e trasmessi, il VNA ricava i parametri di diffusione in termini di grandezza (cioè attenuazione) e sfasamento, che vengono infine convertiti in numeri complessi con una parte reale e una parte immaginaria. Una scansione completa richiede l'iterazione fino a quando tutte le 24 antenne sono configurate come trasmettitori e, per ciascuna di queste iterazioni, vengono registrati 24 diversi segnali ricevuti (inclusa la riflessione catturata dalla stessa antenna trasmittente). I parametri di scattering ottenuti da una scansione completa vengono memorizzati dal VNA come matrice 24x24 in formato testuale. Questo file rappresenta l'input della fase successiva: l'algoritmo di ricostruzione dell'immagine, che viene eseguito dal blocco di elaborazione giallo nella Figura (2.10).

2.4 Ricostruzione dell'immagine

I dati di scattering raccolti con il sistema vengono elaborati mediante un approccio semplice e ben valutato, basato sulla singular value decomposition (SVD) dell'operatore di scattering che mette in relazione i dati del problema con la funzione di contrasto sconosciuta. L'algoritmo di ricostruzione dell'immagine mira a rilevare le variazioni della massa sanguigna tra due successive acquisizioni di dati.
In particolare, i dati di input del problema di imaging, nel seguito indicato come ΔS, erano rappresentati dalla differenza tra le matrici di scattering misurate in due momenti diversi, mentre l'uscita era un'immagine 3D che mostrava la possibile variazione del contrasto elettrico dei tessuti cerebrali, cioè, $\Delta\chi$, che si verificano tra questi due tempi diversi [5]. La funzione differenziale di contrasto elettrico $\Delta\chi$ è definita come:

$$\Delta\chi = \frac{\Delta\varepsilon}{\varepsilon_b} \tag{2.14}$$

Dove ε_b è la permettività complessa del background non omogeneo all'istante di riferimento (ad esempio, una mappa del cervello alla prima diagnosi) e $\Delta\varepsilon$ è la variazione di permettività complessa tra i due istanti di tempo. Di seguito si farà uso della seguente ipotesi, prendendo come "campo di background" quello irradiato in una testa omogenea avente una permettività relativa pari alla media tra i diversi tessuti del cervello, $\varepsilon_r = 42.5$ e $\sigma = 0.75\frac{S}{m}$ [5].

2.4.1 Applicazione dell'algoritmo TSVD

Le misurazioni dei parametri di scattering vengono effettuate dalle antenne del sistema MWI per ciascuna frequenza di interesse.
Considerando una sola frequenza il campo elettrico può essere scritto come:

$$E_s(r_p, r_q) = \int_\Omega \mathcal{G}_e(r_p, r_q) \cdot E(r, r_q)\chi(r)dr \tag{2.15}$$

dove i parametri sono:

- r_p: coordinate del ricevitore;
- r_q: coordinate del trasmettitore;
- Ω: volume del supporto investigato;
- $\mathcal{G}_e$: funzione verde del modello di riferimento;
- E(r,r_q): campo totale dentro il volume di riferimento;
- χ(r): contrasto elettrico;
- "$\cdot$": prodotto scalare.

Poiché la variazione di contrasto $\Delta\chi$ era localizzata in una piccola porzione del dominio ROI, il problema di imaging può essere linearizzato in modo affidabile sfruttando l'approssimazione di Born distorta [3] [4] [5] [21], in modo da mantenere lineare la relazione tra ΔS e $\Delta\chi$:

$$\Delta S(r_p, r_q) = \mathcal{S}(\Delta\chi) \tag{2.16}$$

L'approssimazione di Born utilizzata in un questo algoritmo è quella al prim'ordine, essa assume che il campo elettrico scatterato possa essere espresso solo in funzione di quello incidente; in più la funzione di Green è la stessa in ogni situazione. Quindi il rispettivo modello di riferimento è sempre lo stesso e a questo punto l'unica incognita che rimane è la distribuzione delle proprietà dielettriche. Dove $\mathcal{S}$ è l'operatore integrale lineare con kernel $-jw\varepsilon_b/4E_b(r_m, r_p) \cdot E_b(r_m, r_q)$, dove r_m mostra le posizioni dei punti in cui il dominio di imaging D è discretizzato, E_b è il campo di background nello scenario imperturbabile. Come metodo affidabile e consolidato per invertire (2.16), sfruttiamo l'algoritmo truncated singular value decomposition (TSVD), che deriva dall'algoritmo SVD, normalmente utilizzato nella ricostruzione delle immagini.

Algoritmo SVD

Nell'algebra lineare la SVD è una fattorizzazione di una matrice reale o complessa rettangolare analoga alla diagonalizzazione di matrici quadrate simmetriche utilizzando una base di autovettori. SVD è un metodo stabile ed efficace per suddividere il sistema in un insieme di componenti linearmente indipendenti, ciascuno dei quali porta il proprio contributo energetico. Un'immagine digitale X di dimensione MxN, con M$\geq$N, può essere rappresentata dal suo SVD come segue [23]:

$$[X] = [U] \cdot [S] \cdot [V]^T \tag{2.17}$$

Con:

$$U = [u_1, u_2, ...u_m]$$
$$V = [v_1, v_2, ...v_n] \tag{2.18}$$
$$S = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

Dove [U] è una matrice ortogonale MxM, $[V]^T$ è la matrice trasposta coniugata della matrice [V] di dimensione NxN, mentre [S] è una matrice diagonale NxM, pertanto con valori diversi da zero presenti solo sulla diagonale sempre più decrescenti chiamati Singular Value (σ_i di X).
Ognuno delle colonne di [U] e $[V]^T$ sono chiamate rispettivamente singular vectors sinistro e singolar vectors destro.
I Singular Vectors Sinistri (LSC) di X sono gli autovettori di XX^T, mentre i singular Vectors destri (RSC) di X sono autovettori di X^TX. I singular Values (SV) invece corrispondono alla radice quadrata degli autovalori di XX^T o X^TX [23].
Ogni SV indica il contributo che ogni corrispondente coppia di singular vectors (SC) ha nella ricostruzione finale dell'immagine.
Per facilitare la lettura in questa tesi, d'ora in avanti la matrice [X] sarà rappresentato dall'operatore $[\mathcal{S}]$.

Algoritmo TSVD

L'algoritmo TSVD permette di ottenere la funzione contrasto differenziale sconosciuta attraverso la formula d'inversione esplicita:

$$\Delta\chi = \sum_{n=1}^{L_t} \frac{1}{\sigma_n} \langle [\Delta S], [u_n] \rangle [v_n] \tag{2.19}$$

dove σ_n, $[u_n]$ e $[v_n]$ sono i singolar values e i singolar vectors sinistro e destro dell'operatore di scattering discretizzato $\mathcal{S}$, rispettivamente [3] [21] [5] [4].

L_t è l'indice di troncamento della SVD, che funge da parametro di regolarizzazione ed è stato scelto per ottenere un buon compromesso tra stabilità e accuratezza della ricostruzione.
Ruolo importante in questo algoritmo è svolto dal truncation factor che può essere considerato come una soglia tra l'informazione valida e il rumore di fondo. Generalmente per scegliere un valore, viene considerato il grafico degli autovalori. Infine viene quindi ottenuta la seguente immagine dell'algoritmo TSVD, fig (2.12).

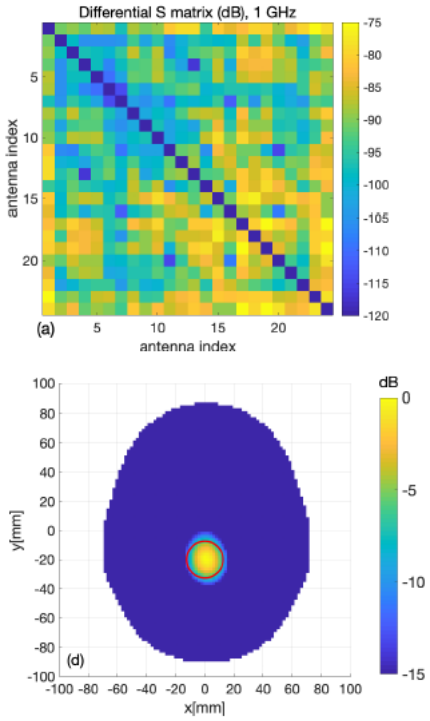


Figura 2.12: Matrice di Scattering e Output dell'algortimo TSVD, in particolare si può notare la ricostruzione in sezione dell'immagine del cervello avente una pallina al centro che ne simula il coagulo di sangue (ictus emorragico) [5].

2.5 Obiettivo della tesi

In genere, la MWI richiede programmi molto costosi dal punto di vista computazionale per ricostruire un'immagine delle proprietà dielettriche di un dato volume. È qui che entra in gioco l'apprendimento automatico: in confronto, è più veloce ed economico dal punto di vista computazionale.
La maggior parte del lavoro nell'ambito della diagnostica a microonde si è concentrato sulla ricostruzione dell'immagine. Con i recenti progressi nell'apprendimento automatico, è possibile un approccio fondamentalmente diverso: utilizzare tecniche di riconoscimento dei modelli per differenziare i pazienti con una lesione da soggetti sani.
L'apprendimento automatico indica metodi che possono dare "ai computer la capacità di apprendere senza essere programmati esplicitamente". L'apprendimento automatico gioca un ruolo centrale nell'intelligenza artificiale e viene utilizzato con successo in numerosi campi diversi, ad esempio, scrittura a mano e riconoscimento vocale, guida autonoma e applicazioni mediche.

In passato molti studi si sono concentrati sull'applicazione delle tecniche di Machine Learning (ML), ad esempio nel rilevamento del cancro al seno [24]. In genere venivano utilizzati tre algoritmi di ML meno intensivi dal punto di vista computazionale (come Support Vector Machine (SVM) e Multilayer Perceptron (MLP)), che non richiedono una ricerca approfondita dei parametri per la loro ottimizzazione [25]. Il sistema è progettato per facilitarne un uso remoto senza l'accesso alle strutture sanitarie tradizionali, rendendo il dispositivo e tutto il sistema portatile e con un uso necessariamente senza un operatore qualificato.

L'apprendimento automatico è promettente anche per l'uso nella diagnostica a microonde di "traumatic brain injury" (TBI) ovvero trauma cranico ed ictus.
Questo approccio è stato convalidato con successo con studi numerici e matematici, per stimare la dimensione e la posizione dei sanguinamenti provocati da un emorragia.
Il vantaggio di un approccio di apprendimento automatico rispetto all'imaging è che una diagnosi può essere generata in tempo reale [12].
Per l'identificazione della tipologia di ictus insieme alla sua localizzazione all'interno del cervello, sono state adottate strategie di Learning By Examples (LBE), con un rilevamento rapido e affidabile. Anche in questo caso il problema di classificazione multi-classe, è stato risolto attraverso l'algoritmo SVM.
Il tutto è stato verificato rispetto ai dati reali di scattering controllati in laboratori che mostravano una notevole accuratezza di predizione insieme a un'elevatissima efficienza computazionale. Essendo queste caratteristiche di fondamentale importanza per un intervento rapido, lo sfruttamento di tale strategia LBE-MWI sembra essere praticabile come strumento alternativo ed utile per la diagnosi clinica dell'ictus cerebrale [26].

Per realizzare tutto ciò, in teoria si doveva costruire un dataset di misure sperimentali con le quali successivamente bisognava testare ed allenare gli algoritmi di Machine Learning; ma poiché idealmente la costruzione delle scene richieste avrebbe impiegato un tempo molto lungo e avrebbe comportato la realizzazione di un numero elevato di misure sperimentali, sono stati simulati, tramite piattaforma Matlab, tutti i segnali di scattering che insieme hanno formato il dataset di interesse.

Capitolo 3

Machine Learning e principio di funzionamento

Come idea generale, il Machine Learning (ML) utilizza algoritmi e tecniche che consentono ai computer di apprendere e generare da soli nuove informazioni. Più concretamente, le definizioni orientate all'ingegneria descrivono il ML come l'istanza in cui i programmi possono apprendere da una certa esperienza E, rispetto ad alcune classi di compiti T e qualche misura delle prestazioni espresse da una metrica P, fig (3.1). Si dice, quindi, che un tale programma ha effettivamente appreso se la sua prestazione su T, misurata da P, migliora con E [27] [28].

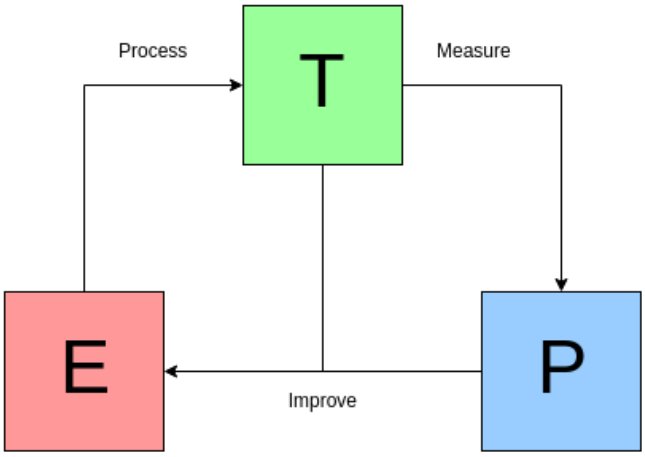


Figura 3.1: Approccio generale Machine Learning [29]

Spesso si ricorre all'utilizzo del ML quando ci sono [28]:

- problemi per i quali soluzioni già esistenti richiedono molti passaggi di ottimizzazione o più regole;
- problemi complessi per i quali gli approcci tradizionali offrono soluzioni scadenti;
- problemi per i quali è necessario acquisire informazioni su grandi quantità di dati o sulla natura del problema stesso.

In questo senso, e in questo contesto, utilizzando il ML si può spesso semplificare il codice e ottenere prestazioni migliori attraverso nuovi algoritmi e modelli.

3.1 Approccio generale

Di solito, e in generale, un flusso di lavoro di Machine Learning inizia analizzando attentamente il problema. Ci sono diversi punti da considerare durante questa fase iniziale e aspetti da descrivere, tra cui [28]:

- definire l'obiettivo del problema;
- analizzare soluzioni già esistenti al problema;
- inquadrare il problema (supervisionato / non supervisionato);
- stabilire una metrica di performance appropriata che sia allineata con l'obiettivo;
- determinare il valore minimo della metrica delle prestazioni affinché la soluzione sia considerata sufficientemente adeguata;
- verifica delle ipotesi.

I passaggi appena descritti fanno parte del riquadro nella Figura (3.2). Il prossimo passo è raccogliere i dati. In questo caso, è sempre importante sapere quali e quanti dati sono necessari, oltre a ottenere le autorizzazioni necessarie per accedere ai dati. Inoltre, è anche importante convertirlo in un formato che può essere manipolato facilmente, come gli array NumPy nel caso di questo progetto. Inoltre, alcuni progetti potrebbero richiedere la protezione o addirittura l'eliminazione dei dati sensibili. A questo punto è utile iniziare a organizzare i dati. In particolare, i campioni disponibili dovrebbero essere suddivisi in diversi set, che vengono utilizzati per scopi diversi. Sebbene sia possibile utilizzare diverse tecniche e tipi di set, i set tipici utilizzati per Machine Learning sono i set di training, validation e test [28]. Il training set viene utilizzato per addestrare il modello. Il validation set, da parte sua, serve per validare i risultati ottenuti in fase di formazione. Il test set

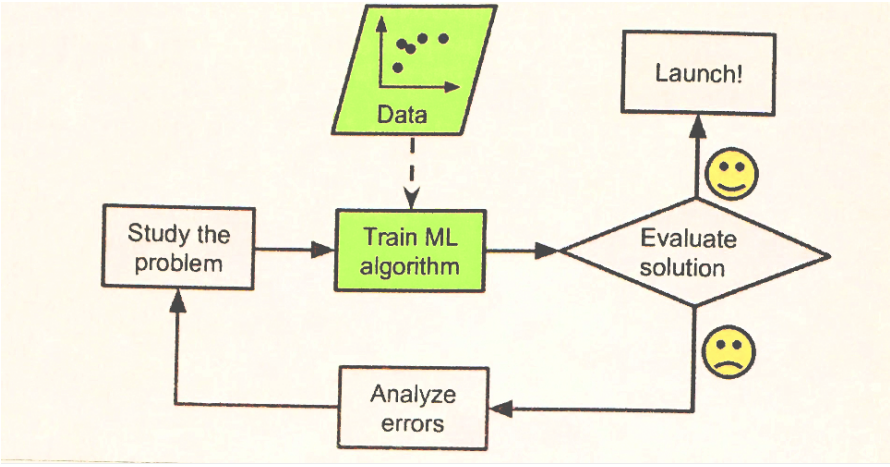


Figura 3.2: Schema logico per risolvere un problema di ML [27]

viene solitamente utilizzata alla fine del flusso di lavoro per analizzare gli errori. Pertanto, viene messo da parte all'inizio e non viene utilizzato fino a quando non viene raggiunta la fase di analisi degli errori di un progetto di Machine Learning.
Il terzo passaggio nel flusso di lavoro consiste nell'esplorare i dati stessi, questo consentirà di rilevare le caratteristiche nei dati.
Una volta completate tutte le attività precedenti, il flusso di lavoro può passare alla quarta fase, ovvero la preparazione dei dati.
Di solito, i progetti di Machine Learning includono questa quarta fase, in cui si dice che i dati siano preelaborati. Esistono diverse trasformazioni che possono essere applicate ai campioni per facilitare la fase di addestramento. Le trasformazioni rilevanti che sono state applicate in questo progetto saranno discusse più avanti in questo capitolo.
Tutto ciò che è stato descritto fino ad ora è incluso nel riquadro chiamato "Data" nella Figura (3.2).
A seguire sono inclusi nel riquadro denominato "Train ML Algorithm" della Figura (3.2), tutta la formazione di training degli algoritmi, che in seguito in questo capitolo verrà discussa in modo particolare. Inoltre, ci sono diverse tecniche che possono essere utilizzate per misurare le loro prestazioni per confrontarle come la Cross Validation (CV) [28].
Un'altra tecnica comunemente utilizzata consiste nel valutare le prestazioni utilizzando un set di test.
Quest'ultima attività, in cui vengono confrontate le prestazioni, è inclusa nel riquadro denominato "Evaluate Solution" nella Figura (3.2). A questo punto è anche necessario analizzare non solo i valori numerici ottenuti per le metriche utilizzate per valutare le prestazioni, ma anche per indagare sugli errori commessi

da ciascun modello candidato. Questo fa parte del riquadro chiamato "Analyze Errors" nella Figura (3.2). Il processo di training dei modelli e confronto delle loro prestazioni è iterativo. Di solito, questo ciclo viene ripetuto rapidamente una o due volte e potrebbe essere necessario studiare / controllare nuovamente il problema o i dati. Una volta fatto tutto questo, il flusso di lavoro può avanzare verso un sesto passaggio che consiste nell'ottimizzazione degli iperparametri per i modelli candidati.
Un iperparametro è un parametro il cui valore viene utilizzato per controllare il processo di apprendimento, in seguito verrà fatta una maggior descrizione.
Esistono diverse tecniche utilizzate per eseguire questa operazione, che è anche iterativa.
Molto spesso, l'ottimizzazione degli iperparametri viene eseguita utilizzando nuovamente la cross validation in combinazione con altre tecniche come GridSearchCV [30] e RandomizedSearchCV. La Grid Search è stata presa in considerazione per questo progetto, verrà quindi spiegata nelle sottosezioni seguenti.
Verso la fine di questo sesto passaggio che si concentra sull'ottimizzazione, se ci sono diversi modelli candidati per il problema è possibile provare a metterli insieme [27] [28].
Quindi, una volta ottenuto un modello finale promettente, di solito le prestazioni vengono valutate utilizzando un set di test a questo punto. È particolarmente importante elencare tutti i presupposti o i limiti rilevanti del modello sviluppato.

Tutto sommato quindi, in sintesi, gli otto passaggi principali di un progetto di Machine Learning, in generale, sono [28]:

1. Analisi del problema
2. Acquisizione dei dati
3. Esplorazione del problema
4. Preparazione dei dati (preprocessing)
5. Esplorazione preliminare dei modelli candidati
6. Regolazione degli iperparametri (ottimizzazione)
7. Presentazione della soluzione
8. Avvio al modello finale

3.1.1 Uso di Google Colab e Scikit-Learn

In questa tesi è stato utilizzato Google Colab, uno strumento gratuito presente nella suite Google che consente di scrivere ed eseguire in cloud il codice python direttamente dal proprio browser.

Colab è una piattaforma online che offre un servizio di cloud hosting per notebook Jupyter dove creare ricchi documenti che contengono righe di codice, grafici, testi, link e molto altro.
Essendo presente nell'ambiente di Google può essere condiviso con altri utenti che hanno la possibilità di modificarlo e lasciare commenti direttamente nel notebook. Il notebook Jupiter viene infatti eseguito su macchine virtuali di server Google, consentendo lo svincolo dalla parte hardware e di concentrarsi solamente sul codice Python e sui contenuti che si vuole integrare nel notebook.
Le macchine virtuali messe a disposizione in Google Colab ospitano un ambiente configurato che consente di concentrarsi sin da subito sui progetti di Data Science: sono presenti numerose librerie Python.

Inoltre, viene utilizzata Scikit-learn, una libreria open source di apprendimento automatico per linguaccio Python. Contiene algoritmi di classificazione, regressione e clustering, e SVM, ed è progettato per lavorare con librerie Numpy.

Il successivo paragrafo descriverà i principali tipi di sistemi ML.

3.2 Sistemi di Machine Learning

Esistono diversi tipi di sistemi di Machine Learning. Inoltre, è utile classificarli in modo ampio in modo che possano essere descritti secondo i seguenti criteri [27] [28]:

- Sistemi la cui fase di training includeva o meno la supervisione umana (supervisionato, non supervisionato, semisupervato, Reinforcement learning).
- Sistemi con la capacità di apprendere in modo incrementale o meno dai flussi di dati in entrata (online vs batch learning).
- Sistemi che possono funzionare confrontando nuovi dati con dati esistenti in precedenza o rilevando modelli nei campioni di addestramento per costruire modelli predittivi (instance-based vs model-based learning).

3.2.1 Classificazioni basate sulla fase di training

Come accennato in precedenza, la natura della fase di training può portare a classificazioni diverse. In particolare, il primo passo per descrivere correttamente un sistema di Machine Learning in base al suo training è identificare se durante questa fase c'era o meno una componente di supervisione umana e in che misura. Di conseguenza, un primo tipo di classificazione ML è l'apprendimento supervisionato [28].

Apprendimento Supervisionato

Nel training supervisionato, ogni campione in un set di addestramento è associato a un valore di output specifico (noto anche come target label). Si ritiene che in questo caso ci sia la supervisione umana, poiché le label per ogni campione dovrebbero essere fornite da qualcuno prima dell'inizio della formazione. In altre parole, il programma non deduce da solo le label della formazione impostata. Le tipiche attività di apprendimento supervisionato includono [28]:

- Attività di classificazione: dove è richiesta la classificazione dei campioni come appartenenti a una classe specifica, come per le e-mail spam e non.
- Attività di regressione: dove è necessaria la previsione dei valori numerici di destinazione e dove i campioni hanno un determinato numero di caratteristiche, chiamate anche predittori.

Nel primo caso, molti esempi delle diverse classi, insieme alle class label, dovrebbero essere assegnati al modello in formazione. Nella seconda, i campioni che includono sia i predittori che le label dovrebbero essere inviati al modello.
Il termine features dovrebbe essere spiegato a questo punto. In Machine Learning, un attributo è un tipo di dati (come la marca di un'auto, ad esempio) [27]. Una feature può avere diversi significati ma di solito indica un attributo e il suo valore [27], quindi, ad esempio, quando si fa riferimento alla popolazione di un'area, popolazione = 5 milioni (persone). Molti usano le parole attributo e feature in modo intercambiabile [28].
In questa tesi sono stati utilizzati tre algoritmi supervisionati (MLP, SVM, k-NN) che nel successivo paragrafo verranno brevemente descritti.

Apprendimento non Supervisionato

Al contrario, un'altra classificazione basata sulla fase di training è l'apprendimento non supervisionata, in cui le istanze sono prive di label. Fondamentalmente, il modello troverà i modelli nei dati e li gestirà a seconda dei modelli trovati. Le applicazioni tipiche in cui ciò è utile includono [28]:

- Divisione dei dati (in gruppi), come quando si identificano gruppi di consumatori in un particolare mercato o tipi di utenti che visitano un blog.
- Rilevamento di anomalie, come strane transazioni con carta di credito o difetti di fabbricazione.

Entrambi (algoritmi supervisionati e non), possono essere combinati per produrre una terza categoria basata sulla fase di formazione, che è quella dei sistemi di apprendimento semisupervised.

Algoritmi Semisupervisionati

Gli algoritmi di apprendimento semisupervised possono gestire i dati parzialmente etichettati. Alcune applicazioni in cui vengono utilizzati questi algoritmi includono servizi di hosting di foto per siti Web di social media. Gli algoritmi sono in grado di identificare i volti nelle immagini utilizzando l'apprendimento senza supervisione. Quindi, una volta che un utente "tagga" una persona, la parte di apprendimento supervisionato dell'algoritmo sarà in grado di nominare la stessa persona in immagini diverse [28].

Algoritmi Reinforcement Learning

Infine, l'ultima classificazione basata sulla fase di formazione corrisponde al Reinforcement Learning. Questo è un tipo di sistema di apprendimento molto comunemente usato in Robotica. In particolare, il sistema, chiamato agente, osserva l'ambiente ed esegue determinate azioni. Può ricevere premi o penalità, in base al comportamento scelto.
Di conseguenza, il sistema impara da solo ad ottenere il maggior numero di ricompense possibili nel tempo, adattando il proprio comportamento secondo necessità [28].

3.2.2 Categorizzazione in base all'output del sistema di ML

- Nella classificazione, gli output sono divisi in due o più classi e il sistema di apprendimento deve produrre un modello che assegni i dati di input alle rispettive classi. Questo discorso viene affrontato solitamente in maniera supervisionata.
- Nella regressione, il modello viene utilizzato per prevedere il valore di una variabile di output (variabile dipendente) in base al valore di un'altra variabile di input (variabile indipendente). La variabile di output avrà un valore continuo a differenza della classificazione in cui si ha un valore finito. Anche in questo caso si tratta di un problema supervisionato.
- Nel clustering un insieme di input viene diviso in gruppi non noti a priori (l'output del sistema) ma generati durante l'apprendimento, pertanto si tratta di un problema non supervisionato.

3.3 Fattori limitanti per i modelli di Machine Learning

I problemi nei modelli di Machine Learning possono avere origine in due possibili aree.

3.3.1 Problemi e sfide relative ai dati

Per quanto riguarda i dati, ci sono quattro diversi aspetti che potrebbero influenzare un algoritmo di Machine Learning. Il primo è la quantità di dati; in particolare, una quantità insufficiente di campioni probabilmente porterà a prestazioni scadenti. La maggior parte dei progetti sul Machine Learning, anche quelli molto semplici, richiedono migliaia di esempi per la formazione. Problemi più complessi, come il riconoscimento vocale, avranno bisogno di milioni di esempi per addestrare correttamente l'algoritmo ML [27] [28].
Un secondo aspetto relativo ai dati da considerare è quanto siano rappresentativi dell'intera situazione in esame. In particolare, i dati di formazione che non sono rappresentativi creeranno problemi.
Più specificamente, affinché un modello di Machine Learning possa generalizzarsi bene a nuovi campioni, i dati di addestramento dovrebbero essere rappresentativi dei nuovi casi. Pertanto, i dati di training dovrebbero mirare a rappresentare l'intera situazione nel suo insieme, non solo una parte del caso di studio.
Nel caso in cui i set di dati siano piccoli, potrebbe esserci rumore di campionamento, il che significa che ci saranno campioni non rappresentativi nel set a causa della casualità. D'altra parte, quando i set di dati sono più grandi, la metodologia di campionamento utilizzata per raccogliere i dati potrebbe essere non ottimale. In casi come questo, il termine usato per descrivere questa situazione è chiamato bias di campionamento.
Mentre i campioni non rappresentativi non considerano un'intera situazione sotto studio, i dati sono di scarsa qualità quando i campioni hanno molti errori, valori anomali e/o rumore. Occorre dedicare una notevole quantità di tempo alla correzione di dati di scarsa qualità. In particolare [27] [28]:

- Quando ci sono campioni che sono chiaramente valori anomali, vale la pena considerare di rimuoverli o correggere gli errori manualmente.
- Se mancano funzionalità (un errore) per alcuni campioni, ci sono diverse opzioni che possono essere prese. Per esempio:
  - Gli attributi potrebbero essere ignorati.
  - I campioni che presentano le caratteristiche mancanti potrebbero essere scartati.
  - I valori mancanti potrebbero essere riempiti, ad esempio, con valori medi per quella caratteristica.

L'ultimo e il quarto aspetto relativo ai dati che di solito crea problemi è quando i campioni hanno caratteristiche irrilevanti. Di solito le attività che possono essere svolte per evitare un problema relativo alle funzionalità sono [27] [28]:

- Selezione delle funzionalità più rilevanti tra quelle disponibili per la fase di training (*feature selection*).
- Combinazione di features in altre più utili (*feature extraction*; possono essere utili algoritmi di riduzione della dimensionalità come PCA).

3.3.2 Problemi e sfide relative all'algoritmo

L'altra area in cui un progetto di Machine Learning potrebbe incontrare difficoltà è nella scelta del tipo di algoritmo. Se il tipo di algoritmo scelto non è ben scelto, sarà possibile osservare casi di overfitting o underfitting [28].

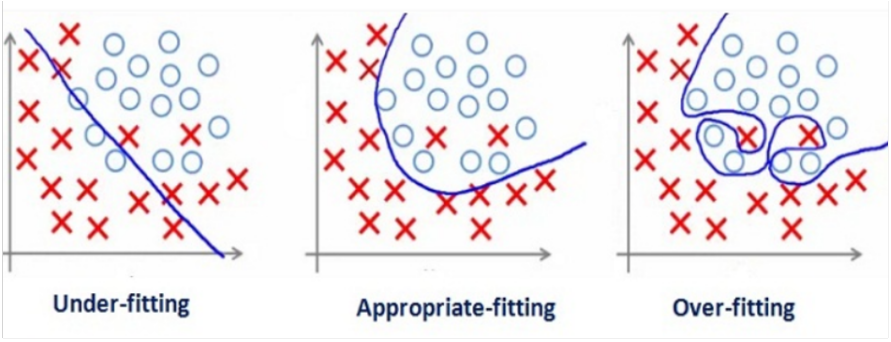


Figura 3.3: Underfitting, Normal Fitting, Overfitting [31]

La Figura (3.3) mostra i tre possibili casi di come un algoritmo può adattarsi ai dati di addestramento disponibili. A sinistra, c'è il caso di underfitting, al centro il caso di un buon adattamento e a destra il caso di overfitting. L'underfitting e l'overfitting sono applicabili a tutti i tipi di algoritmi ML [28].

Underfitting

L'underfitting si verifica quando un modello è troppo semplice per apprendere in modo appropriato dai dati di addestramento. Ciò potrebbe accadere se l'architettura è troppo semplice o se una certa feature viene penalizzata troppo duramente [28]. Ora, come esempio, è possibile osservare che a sinistra nella Figura (3.3) la linea retta non si adatta affatto ai campioni. Questo semplice modello sarebbe un pessimo predittore di nuovi dati. Forse, ad esempio, un modello leggermente più complesso che utilizza un grado polinomiale più elevato sarebbe più adatto se si utilizza la regressione lineare per un caso come questo.
In generale, quindi, è possibile rilevare l'underfitting durante la fase di training poiché le prestazioni saranno scarse.
Inoltre, è possibile migliorare una situazione di underfitting provando quanto segue [27] [28]:

- Utilizzo di un modello più complesso, con più parametri.
- Selezione di funzionalità migliori e più informative in modo che il modello possa apprendere dalle informazioni pertinenti.

Overfitting

In una situazione di Overfitting, il modello si adatterà troppo bene ai dati di addestramento. Tuttavia, il modello avrà prestazioni notoriamente scarse rispetto a qualsiasi nuovo dato disponibile. In altre parole, l'algoritmo ML dato non sarà in grado di generalizzare correttamente. L'overfitting può essere definito come una situazione in cui un modello è troppo complesso in relazione sia alla quantità che alla rumorosità dei dati di addestramento [27].
Alcune strategie per migliorare questa situazione sono [28]:

- Semplificando il modello: riducendo il numero di parametri, diminuendo il numero di caratteristiche nei dati o vincolando il modello (usando la regolarizzazione).
- Aumentare il set di allenamento.
- Riduzione del rumore rimuovendo i valori anomali o correggendo gli errori dei dati.

3.4 L'importanza della validazione e della fase di testing

È già stato spiegato che i dati sono generalmente suddivisi in un set di training, validation e test. Ora, di solito, i dati originali vengono prima suddivisi in un set di training e un set di test. Le proporzioni tipiche utilizzate per questo compito sono 80/20 (o 80% per il training set e 20% per il test set) o 90/10 [32]. In generale la suddivisione dei dati dipenderà dalla dimensione del set di dati [27] [28].
In precedenza in questo capitolo, è stato anche spiegato che lo scopo principale del set di test è quello di avere una stima dell'errore di generalizzazione (o anche noto come errore fuori campione).
Da parte sua, una volta che il training set e il test set sono pronti, è possibile passare attraverso una seconda divisione per ottenere il validation set. Per questo, il set di training originale viene suddiviso per creare un set di training finale e il validation set. La validazione è in realtà un aspetto chiave nella metodologia di un progetto di Machine Learning.

3.4.1 Fase di validazione: validation set, cross validation (CV) e il train-dev set

All'inizio, potrebbe sembrare che usare solo un set di training e un set di test dovrebbe essere sufficiente.
Tuttavia, questo approccio presenta un problema cruciale. Per capirlo, supponiamo che esistessero versioni diverse dello stesso modello. Per selezionare il migliore, il modo naturale sarebbe utilizzare il set di test e misurarne le prestazioni. In particolare, quello con l'errore di generalizzazione più basso dovrebbe essere l'opzione migliore [28].
Tuttavia, probabilmente accadrebbe che dopo il lancio di questo modello "migliore", il tasso di errore sarebbe molto più alto del previsto. Essenzialmente, il problema è che quando si utilizza solo un training con un set di test, molto probabilmente il modello finirà per adattarsi eccessivamente al set di test nella ricerca della migliore versione del modello o del miglior tipo di modello. Di conseguenza, è necessario un approccio diverso per evitare questa situazione che crea incertezza sui risultati futuri.
È qui che è utile l'idea di avere un set di validazione. In particolare, lo scopo del validation set è semplicemente valutare diversi modelli per selezionare il miglior candidato prima di utilizzare un il test set, che dovrebbe essere utilizzato alla fine. In questo contesto, un flusso di lavoro meticoloso addestrerebbe diversi modelli sul set di addestramento ridotto e quindi selezionerebbe il migliore con validation set. Successivamente, il miglior candidato verrà formato utilizzando il set di training ridotto e il set di validation combinati. Infine, verrebbe valutato l'errore con il test set [27] [32].
Come previsto, ci sono aspetti da considerare riguardo a questa tecnica [28]:

- Se il validation set è troppo piccolo, potrebbe accadere che un modello non ottimale venga selezionato per passare alla fase successiva.
- Se, al contrario, il validation set è troppo grande, il training set ridotto sarà troppo piccolo. Ciò, ovviamente, influenzerà lo sviluppo di un modello.

Un modo per evitare queste situazioni il più possibile è usare un'altra tecnica di validazione chiamata cross validation (o anche conosciuta come "CV"). Questa tecnica, che verrà descritta immediatamente, evita anche di dover addestrare modelli su training set ridotti ma consente anche di eseguire la validazione.
In particolare, CV funziona dividendo l'intero set di formazione in diversi sottoinsiemi. Di solito, il numero di divisioni è 3 o 5. Quindi, un sottoinsieme diventa l'insieme di validazione, mentre gli altri sottoinsiemi diventano l'insieme di training. Una volta che il modello è stato addestrato e convalidato, il processo viene ripetuto nuovamente fino a quando tutti i sottoinsiemi non hanno agito una volta come validation set.

Il tempo di addestramento è evidentemente aumentato ed è quindi moltiplicato per il numero di set di validazione [27] [28].

3.5 Data preprocessing utilizzando Scikit

Finora sono stati discussi argomenti relativi agli algoritmi o ai modelli ML. Altrettanto importanti, come sottolineato in precedenza negli altri paragrafi, sono gli aspetti di elaborazione preliminare dei dati. Le due tecniche principali per il preprocessing dei dati e come queste vengono implementate da Scikit-Learn, grazie al pacchetto *sklearn.preprocessing*, che fornisce diverse funzioni di utilità comuni e classi di trasformatori per modificare le features grezze in una rappresentazione più adatta per l'addestramento.

La prima tecnica corrisponde alla standardizzazione. Per eseguire la trasformazione dei dati, Scikit fornisce un oggetto "scaler" creato utilizzando *StandardScaler*.
Nello specifico, ciò che fa questo processo è quello di eseguire quanto segue [33] [28]:

$$z_k^{(i)} = \frac{x_k^{(i)} - \mu_k}{s_k} \tag{3.1}$$

Con k = 1, .., n, con n il numero di features e dove:

- $z_k^{(i)}$ è la feature k-esima trasformata del campione i-esimo, $z^{(i)}$;
- $x_k^{(i)}$ è la feature k-esima del campione i, $x^{(i)}$;
- μ_k è la media della feature k-esima (calcolata utilizzando i campioni di training);
- s_k è la deviazione standard relativa alla k-esima feature (calcolata utilizzando l'insieme di training).

Quello che fa la funzione StandardScaler è quello di standardizzare le features. Inoltre, come indicato in precedenza e spiegato in [33], le azioni di *centering e scaling* avvengono indipendentemente su ciascuna feature dei dati calcolando le statistiche richieste con l'insieme di training. L'oggetto "scaler" può essere salvato e riutilizzato in seguito. Questo perché questo oggetto memorizzerà le statistiche richieste che sono state calcolate con il training set. In particolare, il metodo "fit" dovrebbe essere utilizzato con il training set come input. Quindi, il metodo "trasform" dovrebbe essere utilizzato con il training set e qualsiasi validation/test set.
La standardizzazione è molto comunemente utilizzata nel Machine Learning ed è stata utilizzata in questo progetto di tesi. In genere la feature scaling viene usato per rendere i dati analizzati più gestibili e allo stesso tempo preservandone il

significato.
Molto spesso è, infatti, un requisito, poiché i modelli potrebbero non funzionare adeguatamente quando le feature non sono distribuite normalmente (es. Distribuzione gaussiana con media uguale a zero e deviazione standard unitaria).

La seconda tecnica di preprocessing è la Principal Component Analysis (PCA), che viene eseguita in questo progetto di tesi dopo la standardizzazione delle features. Lo scopo di questa tecnica è quello di ridurre il numero di variabili che descrivono un insieme di dati a un numero minore di variabili rappresentative, limitando il più possibile la perdita di informazioni. La riduzione della dimensionalità lineare avviene utilizzando la decomposizione del valore singolare (SVD) dei dati per proiettarli in uno spazio dimensionale inferiore.
PCA permette di trovare le direzioni della massima varianza dei dati con un ampia dimensione e di proiettarle su un nuovo sottospazio con dimensioni uguali o inferiori a quello originale. Utilizzando la proiezione matematica, il set di dati originale, che potrebbe aver coinvolto molte variabili, viene ad essere interpretato solo da poche variabili (dette componenti principali). L'output di PCA sono proprio queste componenti principali, il cui numero è inferiore o uguale al numero di variabili originali.
Il tutto si riduce ad un calcolo di autovettori e autovalori, per poi applicare la seguente formula per trovare le rispettive componenti principali CP:

$$CP = W^T B^T \tag{3.2}$$

dove:

- CP è la matrice delle componenti principali;
- W^T è la matrice trasposta degli autovettori calcolata dagli autovalori più alti;
- B^T è la versione trasposta ridimensionata/standardizzata dei dati di partenza.

In scikit-learn, la PCA è implementata come un oggetto *trasform* che apprende n componenti con il metodo *fit* e può essere utilizzato su nuovi dati per proiettarlo su queste componenti.

3.6 Concetti di parametri, iperparametri e Grid Search

Il concetto di parametri (parametri del modello) è già stato spiegato in precedenza in questo capitolo. In particolare, si riferisce ai parametri che vengono ottimizzati durante l'allenamento per minimizzare una funzione di costo (o massimizzare una

funzione di utilità in alcuni casi). Il parametro è una caratteristica interna del modello e il suo valore può essere stimato dai dati. Ad esempio, i coefficienti β di regressione lineare / logistica o vettori di supporto in Support Vector Machines [28].

Al contrario, gli iperparametri sono parametri (non modello) che fanno parte di un algoritmo di apprendimento ma non del modello stesso [27], il cui valore non può essere stimato dai dati. Esempi in questa categoria sono i parametri di regolarizzazione, che vengono utilizzati per penalizzare i parametri del modello durante la fase di addestramento. Il valore dell'iperparametro deve essere impostato prima che inizi il processo di apprendimento. Ad esempio, "c" in Support Vector Machines, "k" in k-Nearest Neighbors, il numero dei hidden-layers nelle reti neurali [34].

La Grid Search viene utilizzata per trovare gli iperparametri ottimali di un modello che si traduce nelle previsioni più "accurate".
Crea quindi un modello per ogni combinazione degli iperparametri specificati e valuta ogni modello.

Per l'ottimizzazione degli iperparametri, Scikit offre diverse opzioni. Questi consistono, fondamentalmente, nell'utilizzo di griglie. Una griglia in questo contesto è una raccolta di array. Ogni array è associato a un iperparametro, il cui nome è già definito da Scikit.

Inoltre, ogni matrice della griglia conterrà diversi valori da testare. È a questo punto, quando si selezionano i valori da provare per ogni iperparametro, quando le tecniche di griglia sviluppate da Scikit differiscono.
GridSearchCV: corrisponde ad una ricerca esaustiva, il che significa che vengono testate tutte le possibili combinazioni di valori di iperparametro che appartengono alla griglia.

Viene quindi restituito un oggetto, che ha attributi specifici. Dopo aver terminato il processo di ricerca, sarà possibile ottenere i risultati studiando questi attributi. In particolare, best_ params_ e best_ score_ sono molto utili. Questi restituiscono rispettivamente la migliore combinazione e il suo punteggio. Infine, va notato che per GridSearchCV è coinvolta un cross validation. Infatti uno dei parametri di input per inizializzare la ricerca è "cv".
In altre parole, ogni combinazione viene testata eseguendo la convalida incrociata sul set di addestramento e salvando il suo punteggio, che di solito è l'accuratezza predefinita.
Grazie a questa ricerca, sarà possibile ottenere il miglior modello possibile e sfruttarlo così per l'addestramento.

3.7 Classificazione e metriche di performance

Insieme alla regressione, la classificazione è uno dei compiti più comuni svolti dai sistemi di apprendimento supervisionato. Questa tesi, in particolare, si è concentrata sull'utilizzo di algoritmi con apprendimento automatico per la classificazione multiclass dell'ictus cerebrale in funzione della dimensione e della posizione nello spazio.

I termini rilevanti da discutere consistono principalmente nelle metriche di performance. Naturalmente, è possibile utilizzare la Cross Validation (CV) per valutare i modelli candidati come spiegato in precedenza in questo capitolo. In particolare, è possibile ottenere l'accuratezza media per ogni iterazione di CV [28].

3.7.1 Accuracy Score

L'accuratezza della classificazione è ciò che di solito intendiamo quando usiamo il termine accuratezza. L'accuratezza della classificazione può essere definita matematicamente come:

$$A = \frac{CP}{TP} \tag{3.3}$$

Dove "CP" sta per il numero di "previsioni corrette" e "TP" per il numero di "previsioni totali" [35].
Purtroppo questa metrica funziona bene solo se ci sono un numero uguale di campioni appartenenti a ciascuna classe.
Ad esempio, ipotizzando di addestrare il modello con un training set con il 98% di campioni di classe A e il 2% di campioni di classe B. In teoria, il modello può ottenere un accuratezza di training del 98% semplicemente prevedendo ogni campione di training appartenente alla classe A.
Tuttavia, quando lo stesso modello viene testato su un test set con il 60% di campioni di classe A e il 40% di campioni di classe B, l'accuratezza del test scende al 60%. L'accuratezza della classificazione quindi sembrava ottima inizialmente, ma in realtà testando il modello con un set di dati con percentuali diverse di campioni appartenenti alla classe A rispetto alla classe B, il risultato cambia e anche notevolmente.
A volte, infatti, i risultati di accuratezza per queste attività potrebbero essere fuorvianti. Ciò è particolarmente vero quando si lavora con classi distorte. Si tratta di classi in cui la quantità di campioni di una classe è notevolmente maggiore rispetto ai campioni appartenenti alle altre classi.
Pertanto, in casi come questi, il classificatore (o l'algoritmo di classificazione ML) potrebbe potenzialmente imparare a classificare la maggior parte dei campioni di dati di input come appartenenti alla classe più ampia.
Tuttavia, nonostante la possibilità di ottenere questi risultati apparentemente

buoni, i campioni non appartenenti alla classe dominante saranno probabilmente classificati invece in modo errato.
Quindi, sebbene l'accuratezza possa sembrare inizialmente promettente per queste attività, dovrebbe essere utilizzata solo come stima iniziale o metrica complementare.
In altre parole, non dovrebbe essere utilizzata come metrica decisiva e nemmeno come metrica descrittiva per misurare le prestazioni di classificazione.

Di conseguenza, sorge la necessità di metriche più dettagliate. Per i classificatori, quindi, sono stati sviluppati metodi più descrittivi per misurare la qualità delle prestazioni.

3.7.2 Confusion Matrix

La Confusion Matrix è un layout di tabella specifico che consente la visualizzazione le performance di un algoritmo, tipicamente di apprendimento supervisionato. Il nome deriva dal fatto che rende facile vedere se il sistema confonde due classi (cioè comunemente etichettando erroneamente l'una come l'altra).
Ogni riga della matrice rappresenta le istanze della classe "Actual" nel set di dati mentre ogni colonna rappresenta le istanze della classe stimate dal modello, figura (3.4). Il modo più semplice per comprendere la confusion matrix è iniziare con

		Predicted Value: Negative	Predicted Value: Positive
Actual Value	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Figura 3.4: Confusion Matrix Actual Value vs Actual Predicted.

il caso della classificazione binaria (solo due classi possibili). In questo caso, la matrice di confusione sarebbe una matrice 2x2. Le righe rappresentano le previsioni del classificatore (o delle classi previste) e le colonne le classi effettive.
Nella classificazione binaria, le due classi sono generalmente chiamate "Positiva" e "Negativa".

Un classificatore perfetto avrebbe solo voci nelle celle diagonali della matrice di confusione, fig (3.5). Tuttavia, i classificatori in realtà non sono esenti da errori, quindi le altre celle nella matrice verranno riempite. In particolare, nella

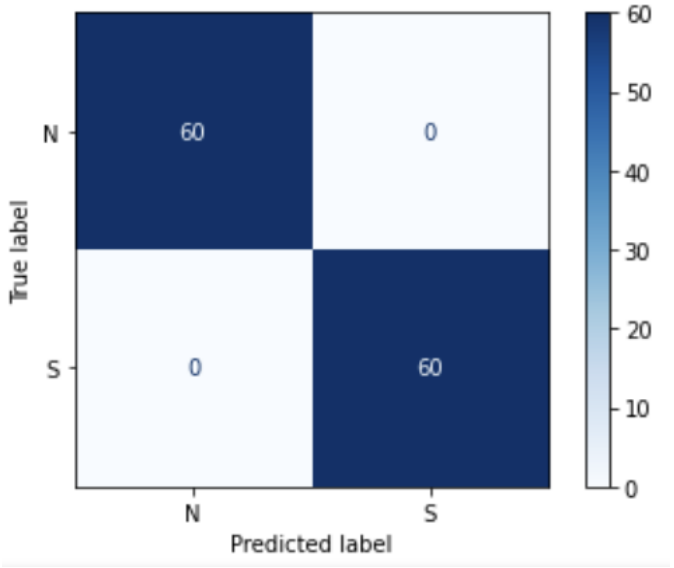


Figura 3.5: Confusion Matrix

classificazione binaria, ci saranno due tipi di errori: i positivi classificati come negativi (falsi negativi o FN) e i negativi classificati come positivi (falsi positivi o FP).
D'altra parte, i positivi classificati correttamente sono noti come veri positivi o TP. Allo stesso modo, i negativi classificati correttamente sono chiamati veri negativi o TN.
Si può lavorare anche con problemi multiclass, dove la matrice di confusione per un classificatore multiclasse sarà una matrice NxN, dove N è il numero di classi, come ad esempio la Figura (3.6), dove appunto è possibile notare che la matrice di confusione è una matrice 3x3, mettendo in risalto che appunto ci sono tre classi. Scikit funziona anche con matrici di confusione normalizzate. Il concetto è molto semplice: si divide ogni voce in una riga per la somma degli elementi in quella riga. In altre parole, la matrice di confusione normalizzata in Scikit normalizza ogni voce di una riga rispetto alla quantità totale di campioni per la classe associata a quella riga, figura (3.7).

3.7.3 Precision and Recall

Quando si analizzano più a fondo i risultati ottenuti da un classificatore, una domanda naturale da porsi è: quale proporzione di ciò che è stato classificato come

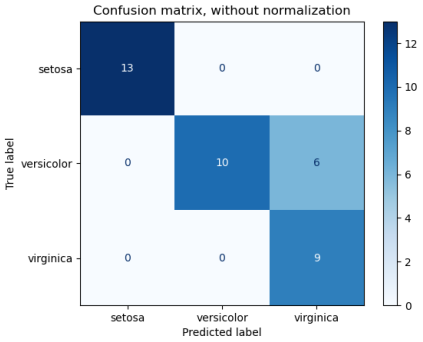


Figura 3.6: Confusion Matrix Multiclass [36].

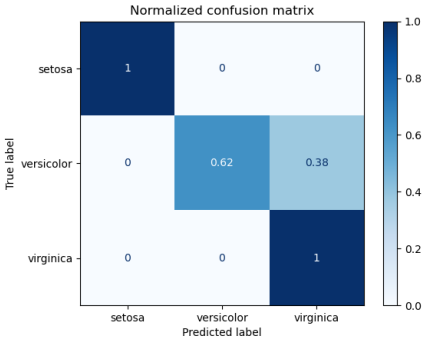


Figura 3.7: Confusion Matrix Multiclass Normalizzata [36].

Positivo (la classe di interesse in molti casi), è veramente Positiva?
La metrica delle performance nota come precisione tenta di rispondere a questa domanda.
Nello specifico, la Precision (P) è matematicamente definita come:

$$P = \frac{TP}{TP + FP} \tag{3.4}$$

Inoltre, è importante notare che il denominatore della formula è in realtà la somma della prima riga della matrice di confusione [37] [28].
Un'altra domanda che è possibile porre, riguardo alle prestazioni di un classificatore, è: quale percentuale dei campioni classificati positivi e' effettivamente positiva?
La metrica che può rispondere a questa domanda è la Recall (R), definita matematicamente come:

$$R = \frac{TP}{TP + FN} \tag{3.5}$$

Va notato che, contrariamente alla Precision, il denominatore di Recall è la somma della prima colonna della matrice di confusione.
Inoltre, ricordando ora la definizione di Precision data prima, è anche possibile ridefinirla in termini di nuovi concetti: TP, TN, FP, FN.
L'accuratezza è un termine più ampio, quindi la domanda a cui cerca di rispondere per le attività di classificazione è: quale proporzione dei campioni di input, sia positivi che negativi, sono stati classificati correttamente?
In considerazione di ciò, è possibile ridefinire l'accuratezza A come:

$$A = \frac{TP + TN}{TP + TN + FN + FP} \tag{3.6}$$

Il tutto è riassunto nella figura (3.8).

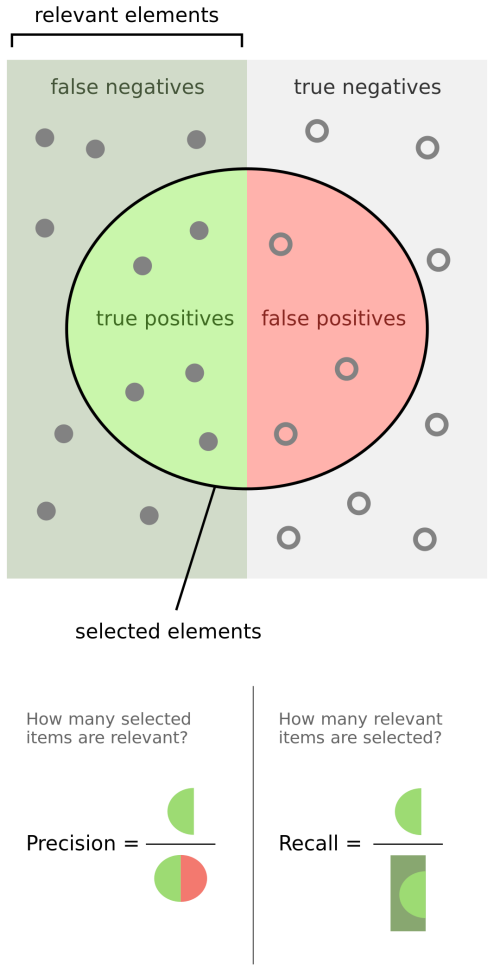


Figura 3.8: Precision e Recall [38].

In genere, c'è un compromesso tra Precision e Recall. Ciò è dovuto al fatto che i risultati saranno influenzati dalle soglie definite per un classificatore per aiutarlo a determinare se un campione deve essere classificato come positivo o negativo [27] [37] [28].

Quindi, anche nei problemi con classificazione multiclass, nel calcolo di Precision e Recall si può sempre sfruttare la confusion matrix. Precisione e richiamo possono essere calcolati ancora una volta, per classe, seguendo la stessa logica.
Infatti, quando si calcola Precisione e Richiamo per una particolare classe, è possibile "comprimere" la matrice di confusione multiclasse in una matrice di confusione binaria. In altre parole, è possibile fingere, per un secondo, che la classe di interesse sia la classe "Positiva" e le altre classi siano una grande classe "Negativa".

È possibile notare che esiste un modello simile a quello descritto prima per il caso binario. Infatti, per Precision, il denominatore è ancora la somma di una riga. Allo stesso modo, per Recall, il denominatore è associato alla somma degli elementi nella colonna.
Di conseguenza, è possibile generalizzare la descrizione precedente a tutte le classi. La precisione per una classe sarà, quindi, il TP per quella classe diviso per la somma delle voci nella riga in cui quella classe è nella matrice. Allo stesso modo, Recall sarà il TP per una classe specifica diviso per la somma delle celle nella colonna in cui quella classe è nella matrice di confusione [37].
Nella diagnosi medica, la sensibilità (Recall) al test è la capacità di un test di identificare correttamente coloro che hanno la malattia (tasso di veri positivi), mentre la specificità (Precision) del test è la capacità del test di identificare correttamente quelli senza la malattia (tasso di veri negativi). Per questo motivo è molto importante costruire classificatori con un alta sensibilità a discapito della specificità, in modo da evitare un mancato riconoscimento di una malattia nei casi in cui si parla di diagnosticare malattie gravi.
In genere per la valutazione delle performance dei modelli ML, sulla base delle metriche discusse prima, viene costruita la Curva Precision-Recall. Essa mostra la relazione tra Precision e Recall in un modello.
Con questo grafico è possibile confrontare le curve Precision-Recall per ogni modello al fine di determinare quale modello ha una relazione accettabile tra i due parametri per uno specifico problema. In particolare, il grafico mostra anche tre andamenti diversi: la Macro Average di Precision-Recall, la Micro Average di Precision-Recall e il valore di Precision-Recall associato a tutte le classi per un modello.
La Macro-Average calcola la metrica indipendentemente per ciascuna classe, facendone poi una media (considerando equamente tutte le classi).
La Micro-Average aggregherà i contributi di tutte le classi per calcolare la metrica media.
In una configurazione di classificazione multi-classe, la Micro-Average è preferibile se si sospetta che ci possa essere uno squilibrio di classe (cioè nel caso in cui ci sono molti più esempi di una classe rispetto ad altre classi).

3.7.4 Curva ROC e AUC

In Machine Learning, la misurazione delle prestazioni è un'attività essenziale. Pertanto, quando si tratta di un problema di classificazione, il più delle volte si ricorre alla curva ROC e al parametro AUC (Area Under The Curve).
È una delle metriche di valutazione più importanti per controllare le prestazioni di qualsiasi modello di classificazione.
ROC è una curva di probabilità e AUC rappresenta il grado o la misura di separabilità. Indica quanto il modello è in grado di distinguere tra le classi. Maggiore

è l'AUC, migliore è il modello nel prevedere 0 come 0 e 1 come 1. Per analogia, maggiore è l'AUC, migliore è il modello nel distinguere tra pazienti con malattia e senza malattia. Vengono definiti, i seguenti parametri:

$$TPR = \frac{TP}{TP + FN} \tag{3.7}$$

$$TNR = \frac{TN}{TN + FP} \tag{3.8}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.9}$$

Che sono rispettivamente:

- TPR (Sensibilità): il Rate di Vero Positivo definito nell'equazione (3.7), corrisponde alla proporzione di punti di dati positivi che sono correttamente considerati positivi, rispetto a tutti i punti di dati positivi;

- TNR (Specificità): il Rate di Vero Negativo definito nell'equazione (3.8), corrisponde alla proporzione di punti di dati negativi che sono correttamente considerati negativi, rispetto a tutti i punti di dati negativi;

- FPR : il Rate di Falsi Positivi definito nell'equazione (3.9), corrisponde alla proporzione di punti dati negativi erroneamente considerati positivi, rispetto a tutti i punti dati negativi.

La curva ROC viene tracciata con TPR rispetto all'FPR; si tratta di una curva parametrica in quanto un punto (TPR ,FPR) della curva è ottenuto per uno specifico valore della soglia (threshold T) che verrà utilizzata per determinare la classificazione in base allo "score" X (se X $<$ T o $>=$ T rispettivamente sarà positiva o negativa, e quindi verrà assegnata ad una certa classe). [39].

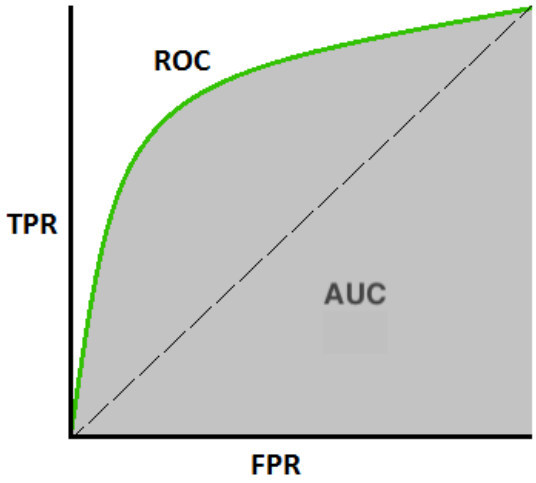


Figura 3.9: Curva ROC/AUC [39].

Le prestazioni di un modello eccellente ha AUC vicino a 1, il che significa che ha una buona misura di separabilità. Un modello scadente ha AUC vicino allo 0, il che significa che ha la peggiore misura di separabilità. In effetti significa che sta scambiando il risultato. Prevede 0 come 1 e 1 come 0. E quando AUC è 0.5, significa che il modello non ha alcuna capacità di separazione delle classi.
ROC è una curva ottenuta a partire dalle distribuzioni di probabilità della classe positiva e negativa.

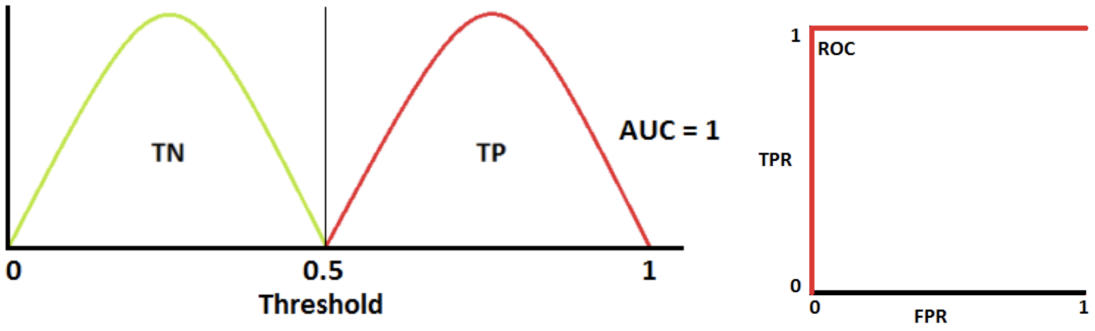


Figura 3.10: Curva ROC con AUC=1 [39]. Nota: la curva di distribuzione rossa è di classe positiva e la curva di distribuzione verde è di classe negativa.

In figura (3.10) è possibile osservare una situazione ideale. Quando due curve non si sovrappongono affatto significa che il modello ha una misura ideale di separabilità. È perfettamente in grado di distinguere tra classe positiva e classe negativa [39].

Quando l'AUC è 0.7, significa che c'è il 70% di possibilità che il modello sia in grado di distinguere tra classe positiva e classe negativa [39].

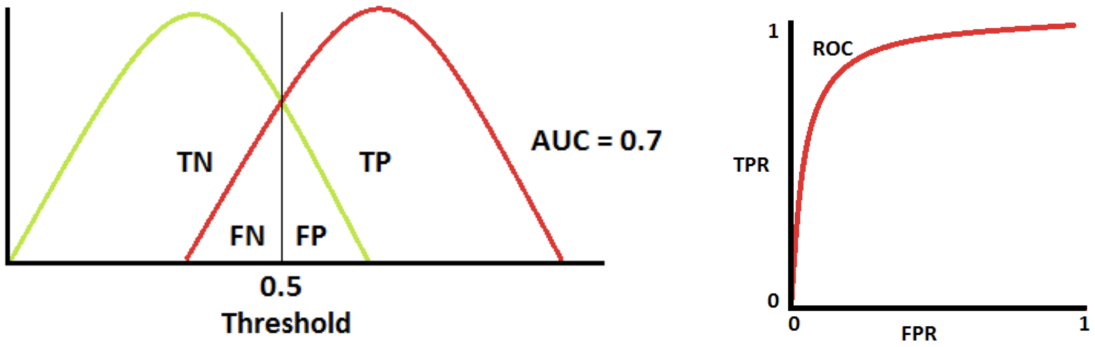


Figura 3.11: Curva ROC con AUC=0.7 [39].

Quando l'AUC è approssimativamente 0, figura (3.12), il modello in realtà scambia le classi; questo significa che il modello prevede la classe negativa come classe positiva e viceversa.

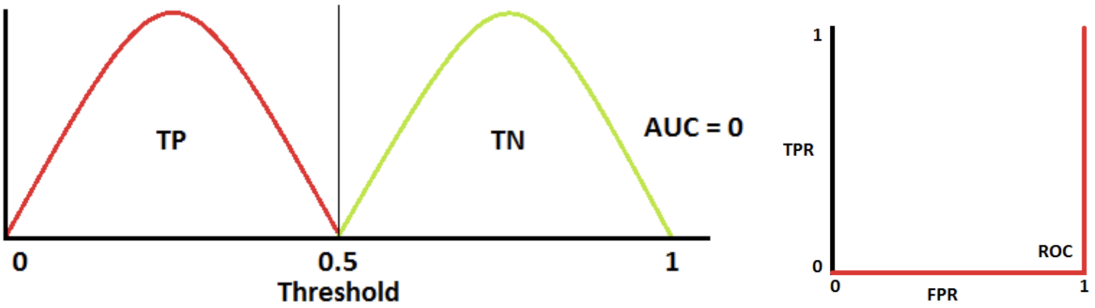


Figura 3.12: Curva ROC con AUC=0 [39].

Nel modello multi-classe, possiamo tracciare un numero N di curve ROC per N classi utilizzando la metodologia One vs ALL. Quindi, ad esempio, se si hanno tre classi denominate X, Y e Z, si avranno una curva ROC per X classificato contro Y e Z, un altra curva ROC per Y classificato contro X e Z e una terza curva per Z classificato contro Y e X. [39].

3.8 Descrizione degli algoritmi proposti: MLP, SVM, k-NN

Dopo aver illustrato i sistemi di machine learning da un punto di vista generale, in questo paragrafo verranno descritti gli algoritmi che sono stati proposti per questa tesi. In generale, per poter utilizzare gli algoritmi di apprendimento automatico, bisogna trovare per prima cosa quale sia il miglior criterio per separare le classi di dati del training set (fase di training). Una volta ottenuto il separatore migliore si potrà poi passare alla fase di test vera e propria.

3.8.1 Multilayer Perceptron (MLP)

Prima di capire come funziona un MLP, è importante parlare della rete neurale di base Perceptron. Un Perceptron si basa su un tipo specifico di neurone artificiale chiamato unità di soglia logica (TLU) o anche unità di soglia lineare (LTU) [28]. Essa è costituita da nient'altro che, una serie di nodi di input e un nodo di output uniti da connessioni pesate; difatti i dati che si spostano da un nodo all'altro vengono moltiplicati per dei pesi. Questi valori scalari sono in realtà la chiave della funzionalità del Perceptron: i pesi vengono modificati durante il processo di training e, regolando automaticamente i suoi pesi in base ai modelli contenuti nei dati di training, la rete acquisisce la capacità di produrre output utili.
Il livello di output, così come eventuali livelli aggiuntivi tra input e output, contengono i nodi di calcolo della rete. Quando i dati numerici arrivano ai nodi computazionali, vengono prima sommati e poi sottoposti a una funzione di "attivazione", figura (3.13).

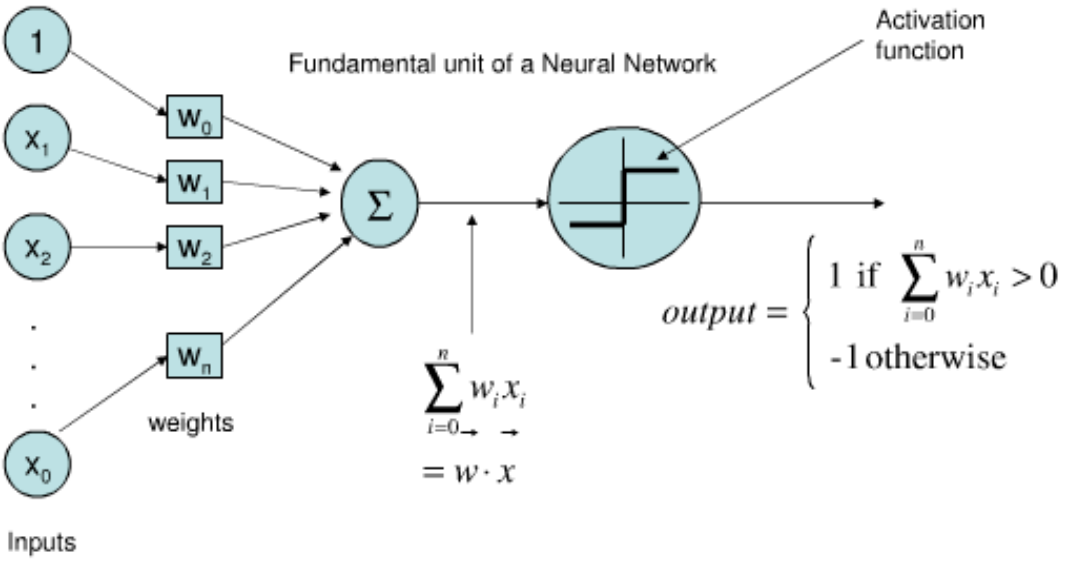


Figura 3.13: TLU-Perceptron a singolo strato [40].

Il concetto di attivazione risale al comportamento dei neuroni (biologici), che comunicano tramite potenziali d'azione attivi o inattivi. Nel contesto delle reti neurali artificiali, i nodi, chiamati anche neuroni, possono imitare il comportamento neuronale applicando una funzione di soglia che emette 1 quando l'input è maggiore della soglia e 0 altrimenti. In sostanza questa funzione gradino, introduce la non linearità nel sistema, e senza questa non linearità la funzionalità di una rete neurale è molto limitata.
In generale i nodi nel livello di input distribuiscono i dati, mentre i nodi negli altri livelli eseguono la somma e quindi applicano una funzione di attivazione. Le connessioni tra questi nodi sono pesate, il che significa che ciascuna connessione moltiplica il dato trasferito per un valore scalare. Questa configurazione è chiamata Perceptron a singolo strato [41].
Il processo che consente a una rete neurale di creare un percorso matematico dall'input all'output è chiamato training. Vengono forniti alla rete dati di training costituiti da valori di input e corrispondenti valori di output, con l'obiettivo di applicare una procedura di modifica graduale dei pesi alla rete in modo tale che la rete sia in grado di calcolare i valori di output corretti anche con dati di input che non ha mai visto prima. Si tratta essenzialmente di trovare modelli nei dati di addestramento e generare pesi che produrranno risultati utili applicando questi modelli a nuovi dati.
Quando il primo calcolo dell'output è completo, si ottengono valori di peso, ma non aiutano a ottenere la classificazione perché sono generati in modo casuale. Si

trasforma così la rete neurale in un sistema di classificazione efficace nel quale si modificando ripetutamente i pesi in modo tale che riflettano gradualmente la relazione matematica tra i dati di input dei valori di output desiderati. La modifica del peso si ottiene applicando la seguente regola di apprendimento per ogni riga del training set [41]:

$$w_{new} = w + (\alpha \cdot (output_{expected} - output_{calculated}) \cdot input) \tag{3.10}$$

Il simbolo α indica il *learning rate*. Per calcolare un nuovo valore di peso, viene moltiplicato il valore di input corrispondente, per il learning rate e per la differenza tra l'output atteso (che è stato fornito dal training set) e l'output calcolato; il risultato di questa moltiplicazione viene così aggiunto al valore del peso corrente. Definendo δ come $output_{expected} - output_{calculated}$, ovvero l'errore, si può scrivere [41]:

$$w_{new} = w + (\alpha \cdot \delta \cdot input) \tag{3.11}$$

La discesa verso il minimo errore si verifica quando i pesi stanno cambiando. Il training obbliga la rete a modificare i propri pesi in un modo che si traduce in una minimizzazione della funzione di errore.
Il learning rate ne influenza la velocità con cui la rete neurale apprende. La rete in sostanza impara ad approssimare la relazione input-output contenuta nei dati di training. La manifestazione di apprendimento è dettata dalla modifica del peso e il learning rate ne influenza il modo in cui i pesi vengono modificati.
Il learning rate quindi influenza la dimensione del passo che porta al minimo errore. Ogni volta che si applica (3.11), il peso salta a un nuovo punto sulla curva di errore. Se δ è grande, anche quei salti potrebbero essere abbastanza grandi e la rete potrebbe non allenarsi in modo efficace perché i pesi non stanno convergendo gradualmente verso l'errore minimo, ma stanno rimbalzando in modo un po' caotico. Poiché δ viene moltiplicato per il learning rate prima che la modifica venga applicata al peso, si può ridurre la dimensione dei salti scegliendo $\alpha < 1$. L'obiettivo è utilizzare la velocità di apprendimento per promuovere una convergenza moderatamente veloce e coerente. In genere si può valutare la scelta di questo parametro confrontando l'accuratezza della classificazione in base al learning rate [41].

Il "Summed Squared Error" è la funzione d'errore; l'aggiornamento dei pesi tramite la Discesa del Gradiente richiede di trovare la derivata parziale della funzione di errore rispetto al peso che vogliamo aggiornare. L'esecuzione di questa differenziazione rivela che il gradiente di errore rispetto ad un peso è dato da un'espressione che include la derivata della funzione di attivazione [41].

La funzione gradino unitario consente dei calcoli all'interno di un nodo di essere molto semplici, ma questo vantaggio diventa privo di significato nel contesto della discesa del gradiente perché questa funzione non è differenziabile.

Se intendiamo addestrare una rete neurale utilizzando la Discesa del Gradiente, abbiamo bisogno di una funzione di attivazione differenziabile. Poiché la funzione unitaria è coerente con il comportamento on/off dei neuroni biologici all'interno di sistemi costituiti da neuroni artificiali, ha senso considerare una funzione di attivazione simile ma che sia differenziabile. Ci si concentrerà sulla funzione sigmoidea logistica [41].

Il MLP è una classe di reti neurali artificiali, con apprendimento supervisionato. Consiste in almeno tre strati di nodi. Ognuno di questi è un neurone che usa una funzione di attivazione non lineare. Questo, lo distingue dal Percettrone a singolo strato descritto in precedenza che riconosce solo classi linearmente separate.

In generale, il MLP è semplicemente la combinazione di strati di TLU: è composto dal livello di input, uno o più livelli di TLU chiamati hidden layers e infine dal livello di output, figura (3.14). Di solito, gli strati che sono vicini ai neuroni di input sono indicati come strati inferiori e quelli più vicini ai neuroni di output come strati superiori. Inoltre, questa particolare architettura è una rete neurale "feedforward" (FNN), poiché i segnali fluiscono in una direzione dagli ingressi alle uscite.
L'algoritmo di backpropagation è l'elemento costitutivo più fondamentale in una rete neurale.
L'algoritmo viene utilizzato per addestrare efficacemente una rete neurale attraverso un metodo chiamato *chain rule*. In termini semplici, il nome backpropagation deriva dal fatto che ad ogni epoca viene calcolato l'errore tra uscita della rete e target desiderato e questo errore viene propagato all'indietro per aggiustare il peso del layer precedente.
Utilizza la somma degli errori quadratici medi come misura della performance della rete, più è bassa la somma degli errori, migliori saranno le performance della rete.
Le due funzioni di attivazione più comuni sono entrambe delle sigmoidi e sono descritte come [42]:

$$y(v_i) = \tanh v_i \tag{3.12}$$

$$y(v_i) = \frac{L}{(1 + e^{-kv_i})} \tag{3.13}$$

In questo caso $y(v_i)$ è l'output dell'i-esimo nodo o neurone e v_i è la somma pesata delle connessioni in input.
La costante L determina il valore massimo della curva mentre la costante k influenza la pendenza della curva.
Poiché le modifiche di peso sono proporzionali alla derivata della funzione di attivazione, le variazioni di peso saranno maggiori per i nodi che "non sono ancora impegnati ad essere attivi o disattivati" e questo potrebbe contribuire alla stabilità dell'apprendimento del sistema. Tuttavia, se si è calcolato l'output della funzione

logistica per un dato valore di input, non è necessario utilizzare l'espressione per la derivata, perché risulta che la derivata della funzione logistica è correlata alla funzione logistica originale come segue:

$$f'(x) = f(x)(1 - f(x)) \tag{3.14}$$

Ogni nodo di ogni strato è collegato al successivo strato. Ogni connessione ha un certo "*peso*" chiamato w_{ij}. L'apprendimento della rete consiste nel cambiare i pesi che le connessioni hanno, in base agli errori che vengono prodotti in output rispetto all'aspettativa,il cosiddetto meccanismo "*backpropagation*". Si devono aggiornare i pesi dall'input allo strato nascosto in base alla differenza tra l'output generato dalla rete e i valori di output target forniti dai dati di training; questi pesi influenzano l'output generato indirettamente. I nodi di bias possono essere incorporati nel

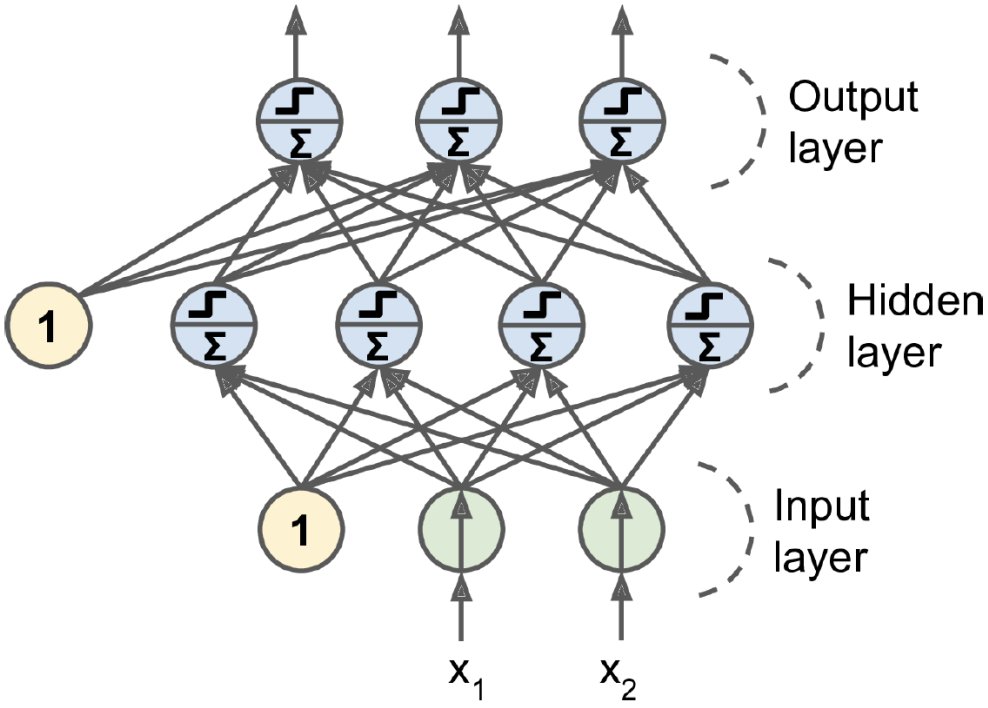


Figura 3.14: Esempio di un MLP [28]

livello di input, nel livello nascosto o in entrambi. I loro pesi sono come qualsiasi altro peso e vengono aggiornati utilizzando la stessa procedura di backpropagation. Un bias sposta il segnale che viene elaborato dalla funzione di attivazione e può quindi rendere la rete più flessibile e robusta. L'uso della lettera b per denotare il valore di bias ricorda l'intercetta y nell'equazione standard per una linea retta: y = mx + b. Si nota anche che la matrice di pesi è equivalente ad una pendenza.
Ciò che fa questo algoritmo di training è quello di trovare ogni valore di peso

e bias, ottimizzandoli, in modo da ridurre l'errore del NN. Una volta ottenuti i gradienti, viene eseguita una fase di Discesa del Gradiente. L'elenco dei passaggi che l'algoritmo esegue, sono descritti di seguito [28]:

- Gestisce un mini-batch alla volta. Un mini-batch è un sottoinsieme dei campioni totali. Questo termine è indicato anche come dimensione del batch in Scikit.
- L'algoritmo di backpropagation passa anche attraverso l'intero set di training più volte. Un'epoca è quando l'algoritmo passa attraverso tutti i campioni e quindi ricomincia il processo per avviare un'altra epoca (a meno che non sia l'ultima).
- Viene eseguito il passaggio in avanti ("forward pass"): un mini-batch viene passato al livello di input, che passa il mini-batch al primo livello nascosto. Backpropagation quindi calcola l'output per ogni neurone in quello strato nascosto e passa questi risultati allo strato successivo. Questo viene ripetuto fino a raggiungere l'ultimo strato. Tutti i risultati vengono salvati.
- Backpropagation misura quindi l'errore di output utilizzando una funzione costo. In particolare, la funzione costo confronterà l'obiettivo e gli output effettivi e li restituirà con una metrica di errore. Tutti i risultati vengono salvati.
- Pertanto, la backpropagation calcolerà quanto ogni output di rete ha contribuito all'errore. Questa operazione viene eseguita applicando la derivazione, quindi è un passaggio veloce.
- L'algoritmo misurerà quindi la quantità di questi contributi di errore provenire dal livello precedente. Questo viene fatto ancora una volta utilizzando la derivazione, seguendo le connessioni tra il livello corrente e quello precedente. Questo viene fatto fino a quando non viene raggiunto il livello di input.
- L'ultimo passaggio consiste nell'eseguire uno step nella Discesa Gradiente per aggiornare i pesi e i bias.

In sintesi, per ogni campione, l'algoritmo di backpropagation esegue prima una previsione (forward pass). Quindi, misura l'errore. Successivamente, ritorna indietro attraverso tutti i livelli, misurando il contributo all'errore di ciascuna connessione. Infine, esegue un passaggio sulla Discesa del Gradiente.
Per l'addestramento, è necessario considerare che tutte le connessioni (i pesi) devono essere inizializzate in modo casuale, altrimenti l'addestramento fallirà. L'errore nel nodo di output j dell'n-esimo dato è rappresentato con $e_j(n) = y_j(n) - d_j(n)$, dove y è il valore che atteso e d il valore prodotto dal percettrone. A questo punto i valori dei pesi dei nodi vengono aggiustati in modo da minimizzare l'errore quadratico

medio dell'intero output [42]:

$$\epsilon(n) = \frac{1}{2}\sum_{j=1} e_j^2(n) \tag{3.15}$$

Utilizzando la Discesa del Gradiente, la variazione dei pesi risulta [42]:

$$\Delta w_{ij}(n) = -\alpha \frac{\partial \epsilon(n)}{\partial v_j(n)} d_i(n) \tag{3.16}$$

dove d_i è l'output del neurone precedente e α è il *learning rate*, che viene selezionato per garantire che i pesi convergano rapidamente ad un valore senza oscillazioni. L'espressione della derivata da calcolare risulta abbastanza semplice per un nodo di output e in particolare diventa [42]:

$$-\frac{\partial \epsilon(n)}{\partial v_j(n)} = e_j(n)\phi'(v_j(n)) \tag{3.17}$$

dove ϕ' è la derivata della funzione di attivazione. Per un nodo di un hidden layer l'analisi diventa più complessa e può essere scritta come [42]:

$$-\frac{\partial \epsilon(n)}{\partial v_j(n)} = \phi'(v_j(n)) \sum_k -\frac{\partial \epsilon(n)}{\partial v_k(n)} w_{kj}(n) \tag{3.18}$$

Quando si implementa la Discesa del Gradiente, ogni modifica del peso sarà proporzionale alla pendenza della funzione di errore rispetto al peso che viene modificato.

Il MLP può essere utilizzato per diverse attività, come la regressione e la classificazione. Rispetto alla classificazione, il MLP è utile per:

- Classificazione binaria: è necessario un singolo neurone di output, che emette un valore compreso tra 0 e 1. Questo valore viene interpretato come la probabilità che un dato campione appartenga o meno alla classe positiva (classe '1').
- Multilabel binary classification: in cui la quantità di neuroni in uscita corrisponderà al numero di classi positive. La classificazione binaria con etichette multiple si verifica quando, ad esempio, un campione può appartenere a due classi diverse contemporaneamente.
  - Gli output sono, ancora una volta, valori compresi tra 0 e 1 che vengono interpretati come la probabilità che un campione appartenga a ciascuna classe, separatamente.

  - Per lo stesso motivo, le probabilità in uscita non si sommano a 1.

- Multiclass Classification: questo è il caso in cui sono presenti più classi di output e ogni campione può appartenere ad una sola classe. In questo caso, è richiesto un neurone di output per ogni classe nel livello di output.

  - In questo caso, le classi saranno esclusive. Esiste una funzione di attivazione speciale che garantisce che le probabilità in uscita si sommino a 1 chiamata softmax.

L'ultimo tipo è il caso di questo progetto di tesi: classificazione multiclasse.

Per le reti neurali, così come per gli MLP, i seguenti iperparametri possono essere regolati per migliorare le prestazioni [28]:

- Numero di layer nascosti (la quantità di livelli tra quelli di input e quelli di output).
- Numero di neuroni in ogni layer nascosto.
- Learning rate.
- Optimizer, eseguono i passaggi di un determinato algoritmo di apprendimento. Tutti gli ottimizzatori funzionano allo stesso modo: aggiornano i parametri del modello, valutano una funzione di perdita e decidono se aggiornare nuovamente i parametri. Questo processo viene ripetuto fino a quando non viene raggiunta una soluzione. In sostanza fanno quello che in precedenza è stato chiamato come Discesa del Gradiente. Gli algoritmi di training più utilizzati sono Adam oppure Stochastic Gradient Descent (SGD).
- Dimensione batch.
- Funzione di attivazione per i livelli nascosti (quella per il livello di output dipende dall'attività).
- Numero di iterazioni (iterazioni di addestramento).
- Dropout rate (porzione di neuroni ignorata) e parametri di regolarizzazione (se utilizzati).

3.8.2 Support Vector Machine (SVM)

La SVM è un modello di apprendimento supervisionato associato ad una famiglia di classificatori molto potenti che permettono di ricondursi ad una classificazione lineare anche se si ha a che fare con problemi di natura non lineare.
Per prima cosa si analizzerà il problema di classificazione binaria, alla base del funzionamento delle SVM, e da cui si potrà fare riferimento anche per la risoluzione

di problemi più complessi. Il problema sarà formato da coppie di dati formati dai campioni (x) e dalle label (y) della corrispondente classe di appartenenza. Si utilizzeranno questi dati per trovare una funzione che permetta di classificare nuovi dati.
Ipotizzando che il training set sia linearmente separabile, si determinerà un iperpiano che separerà le classi di dati, in modo che i punti si possano suddividere tra due semispazi. Tutti i punti che giacciono sull'iperpiano separatore dovranno soddisfare l'equazione:

$$w \cdot x + b = 0 \tag{3.19}$$

Dove w è il vettore normale all'iperpiano e b l'intercetta all'origine. Supponendo che tutti i dati del training set soddisfino la seguente disuguaglianza [43]:

$$y_i(w \cdot x_i + b) \geq 1 \qquad \forall i \tag{3.20}$$

Teoricamente esistono infiniti iperpiani che soddisfano (3.20), ma per una classificazione corretta è necessario determinare i parametri caratteristici (w, b) che tra tutti separa i dati al meglio. Per far ciò si fa ricorso all'utilizzo degli spazi immagine o feature space [42].
Questo metodo consiste nel mappare i dati in uno spazio di dimensione superiore rispetto a quello di input. Dovremo trovare quindi la funzione $\phi : \Re^n \rightarrow \Re^m$, con m > n tale che partendo da dati non separati in $\Re^n$ riesca a separare linearmente i dati nello spazio $\Re^m$ come in figura (3.15).

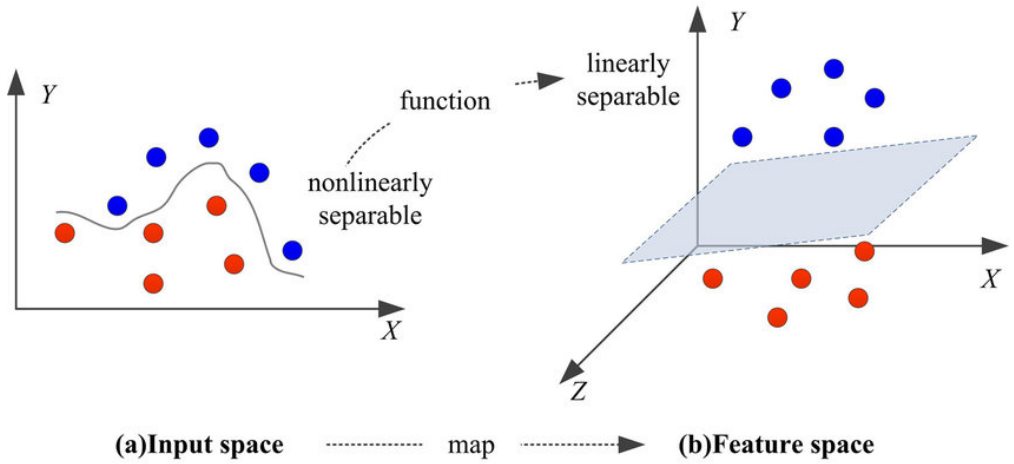


Figura 3.15: Esempio di SVM [44]

La tecnica degli spazi immagine risulta molto interessante per dati di training x_i esprimibili solo attraverso prodotti scalari $x_i \cdot x_j$. In questo caso non è necessario trovare sia $\phi(x_i)$ che $\phi(x_j)$, ma basta calcolare il loro prodotto scalare $\phi(x_i) \cdot \phi(x_j)$,

che chiameremo funzione kernel e indicheremo con $\mathrm{K}(x, y)$. I kernel più utilizzati sono [43]:

- Lineare: $\mathrm{K}(x, y) = x \cdot y$
- Polinomiale: $\mathrm{K}(x, y) = (x \cdot y)^d$
- Gaussian Radial Basis function: $\mathrm{K}(x, y) = e^{-\frac{(|x-y|)^2}{2\sigma^2}}$
- Sigmoide: $\mathrm{K} = \tanh(kx \cdot y - \delta)$

Considerando i punti che soddisfano la disuguaglianza (3.20) essi giacciono sull'iperpiano H1 e sull'iperpiano H2, figura (3.16). H1 e H2 risultano paralleli e nello spazio che li separa non cadono punti del training. La distanza tra i due piani inoltre, che tramite la geometria si deduce essere uguale a $\frac{2}{\|w\|}$, viene definita margine.

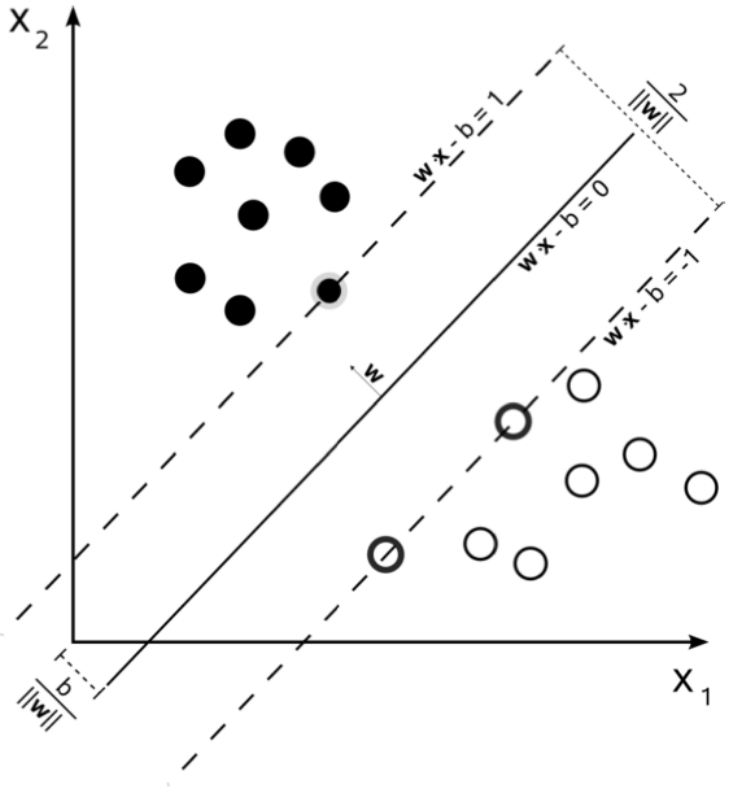


Figura 3.16: Separatore lineare SVM [43]

L'algoritmo dei SVM ha l'obiettivo di rendere massimo il valore del margine, per

far si che lo spazio tra le due classi sia il più ampio possibile. Un primo metodo si basa direttamente sul cercare i due iperpiani che minimizzano $\|w\|^2$, e con w e b che rispettano la disuguaglianza (3.20). Il problema della ricerca da risolvere può essere formalizzato nel modo seguente, per cui si cerca di minimizzare la seguente espressione:

$$\begin{cases} min_{w,b}\frac{1}{2}\|w\|^2 \\ y_i(w \cdot x_i + b) \geq +1 \qquad i = 1, .., N \end{cases}$$

Per quanto riguarda l'utilizzo di funzioni kernel per i casi di problema non lineare, è necessaria una formulazione Lagrangiana, grazie alla quale i dati appariranno solo sotto forma di prodotto scalare. La Lagrangiana ottenuta è la seguente [43]:

$$L_p(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i[y_i(w \cdot x_i + b)] + \sum_{i=1}^{N} \alpha_i \tag{3.21}$$

Nella funzione da minimizzare vengono quindi introdotti dei moltiplicatori di Lagrange α_i. Il problema di ottimizzazione ricade sul minimizzare L_p rispetto a w e b, e contemporaneamente ottenere le derivate di L_p rispetto a tutti gli α_i nulle. Tutti i punti che nella soluzione precedente soddisfano la condizione α_i vengono chiamati vettori di supporto (support vectors) e giacciono su uno dei due iperpiani H1, H2. Essi, essendo i più vicini alla frontiera di decisione, sono punti critici per le SVM.

Si parla invece di dati linearmente non separabili quando ci sono campioni anomali del training set che si trovano nel semipiano sbagliato e la classificazione binaria tramite iperpiani non può essere eseguita in maniera corretta. È quindi necessario rendere più flessibili i vincoli definendo delle variabili slack $\xi_i \geq 0$, tanto maggiori quanto più lontani sono i punti anomali. Si suppone quindi che i dati del training set rispettano la seguente disuguaglianza:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i \qquad \forall i \tag{3.22}$$

Per ogni campione anomalo, la corrispondente ξ_i deve essere più grande dell'unità. La funzione costo da minimizzare viene riscritta rispetto al caso di dati linearmente separabili:

$$\frac{1}{2}\|w\|^2 + C(\sum_i \xi_i)^k \tag{3.23}$$

Dove C e k sono dei parametri scelti a priori. Con un alto valore di C corrisponde un elevato peso degli errori commessi e che con k = 1 o 2 il problema rimane di programmazione quadratica e quindi per risolverlo sarà possibile utilizzare il metodo della Lagrangiana (3.21). Con k = 1 si ha il vantaggio che la formulazione del problema risulta identica a quella già analizzata nel caso in cui i dati del

training set erano separabili linearmente. Dopo la fase di training si ottengono i parametri dell'iperpiano per la separazione ottima dei dati, le SVM procedono alla classificazione vera e propria di nuovi dati. Questa fase viene chiamata fase di test e consiste nel collocare nella giusta classe un campione arbitrario, si utilizza a tal proposito la seguente funzione di decisione:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i y_i x_i \cdot x + b) \tag{3.24}$$

Nel caso di un training set con dati appartenenti a due classi che possono essere separate in maniera non lineare, la situazione va analizzata in maniera diversa. Anche in questo caso ,grazie allo stratagemma matematico dei kernel, le SVM renderanno possibile la generalizzazione di quanto visto in precedenza anche per questa classe di problemi.
Dopo aver risolto il problema di ottimizzazione, si avranno a disposizione i parametri del separatore e si avranno quindi tutti gli strumenti per costruire un classificatore binario non lineare. Tutti i prodotti scalari verranno sostituiti dai kernel corrispondenti. Si otterrà quindi il seguente classificatore:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i y_i k(x_i, x_j) + b) \tag{3.25}$$

Riassumendo le regole generali per l'utilizzo del SVM per la classificazione non lineare sono [43]:

- Scegliere il parametro C che rappresenta un compromesso tra la minimizzazione dell'errore sul training set e la massimizzazione del margine
- Scegliere la funzione kernel da utilizzare
- Risolvere il problema di programmazione quadratica che, essendo la funzione da minimizzare convessa, darà sicuramente una soluzione unica e globale
- Classificare nuovi campioni tramite la funzione di decisione

Per una classificazione multiclass si usano due approcci diversamente: l'approccio one-aganist-all e l'approccio one-aganist-one.
Nell'approccio one-aganist-all, si costruiscono k classificatori in cui k è il numero di classi. Il j-esimo classificatore binario separa i vettori della classe j da quelli di tutte le altre classi. Quindi è addestrato considerando gli elementi della classe j-esima come positivi e tutti gli altri negativi. Per ogni j = 1,..., k si deve addestrare una SVMj e dunque si devono risolvere per ogni j = 1,..., k dei problemi di margine

massimo [45]:

$$min\left\|w^j\right\|^2 + C(\sum_{i=1}^{l}\xi_i)^j \tag{3.26}$$

$$y_i(w^j \cdot x_i + b^j) \geq 1 - \xi_i^j \qquad i = 1, ..l \tag{3.27}$$

Una volta ottenuta la soluzione dei k problemi saranno disponibile k funzioni di decisione $f_k(x)$ come (3.24). In sostanza si risolvono i problemi duali al SVM binario, dovendo risolvere k problemi di dimensione pari a l. Un possibile svantaggio di questo approccio è che richiede la creazione di un modello per ogni classe. Ciò potrebbe essere un problema per set di dati di grandi dimensioni o un numero molto elevato di classi (ad esempio centinaia di classi). Nell'approccio one-aganist-one invece si costruiscono $\frac{k(k-1)}{2}$ classificatori ognuno addestrato sui dati relativi a due sole classi; si divide il set di dati in un set di dati per ogni classe rispetto a ogni altra classe.
Si tratta di un numero significativamente maggiore di set di dati e, a sua volta, modelli rispetto alla strategia one-aganist-all. Dunque, per ogni coppia di indici m e n $\in$ 1,...,k si costruisce un insieme di dati di training che sarà un sottoinsieme dei dati originari. In particolare, per ogni coppia di indici m e n ,si deve addestrare una SVM_{mn} e dunque si devono risolvere per ogni m,n $\in 1, ..., k$ i problemi di margine massimo [45]:

$$min\|w^{mn}\|^2 + C(\sum_{i=1}^{l}\xi_i)^{mn} \tag{3.28}$$

$$y_i(w^{mn} \cdot x_i + b^{mn}) \geq 1 - \xi_i^{mn} \qquad i = 1, ..l \tag{3.29}$$

Una volta ottenuta la soluzione saranno disponibili $\frac{k(k-1)}{2}$ funzioni di decisione.

3.8.3 k-Nearest Neighbour (k-NN)

Il k-NN fa parte della famiglia dell'apprendimento supervisionato, è un algoritmo robusto e versatile utilizzato nel classificare oggetti basandosi sulle caratteristiche degli oggetti vicini a quello considerato. Quello che fa è memorizzare il dataset come "conoscenza" per fare previsioni. A differenza della maggior parte degli altri metodi di classificazione, k-NN rientra nell'apprendimento pigro (lazy learning), il che significa che non esiste una fase di training esplicita prima della classificazione. Qualsiasi tentativo di generalizzare o astrarre i dati viene effettuato al momento della classificazione. Per questo è anche un modello non parametrico, ciò significa che non vi è alcuna ipotesi per la distribuzione dei dati sottostante, ovvero la struttura del modello determinata dal set di dati. Una volta formato il set di dati per l'addestramento, rappresentato come una matrice M x N dove M è il numero

di punti dati e N è il numero di features, inizia la classificazione. In generale l'algoritmo k-NN:

- Calcola un valore di distanza tra l'elemento da classificare e ogni elemento nel training set

- Sceglie i k punti più vicini al campione da classificare (gli elementi con le k distanze più basse)

- Assegna l'etichetta di classe con un voto a maggioranza.

Il concetto di funzionamento è molto semplice e si riduce ad un confronto tra il campione da classificare e le K più simili istanze del dataset [42]. Fatto questo il dato da classificare apparterrà alla classe a cui appartengono la maggior parte dei K elementi etichettati più vicini. I più vicini vengono appunto determinati proprio con la distanza tra due dati.
Ci sono due importanti decisioni che devono essere prese prima di fare le classifiche di maggioranza. La prima risiede sul valore di k che verrà utilizzato che può essere deciso arbitrariamente, oppure attraverso la Cross Validation per trovare un valore ottimale. La successiva, e la più complessa, è la metrica della distanza che verrà utilizzata. In figura (3.17) è rappresentato un esempio dove il punto sotto osservazione è il pallino verde. Le classi sono due:

- quella dei triangolini rossi;

- quella dei quadratini blu.

Se k = 3 (cioè vengono considerati i 3 oggetti più vicini), allora il pallino verde viene inserito nella stessa classe dei triangolini rossi perché sono presenti 2 triangolini e 1 quadratino. Se k = 5 allora viene inserito nella stessa classe dei quadratini blu perché sono presenti 3 quadratini e 2 triangolini, [46].

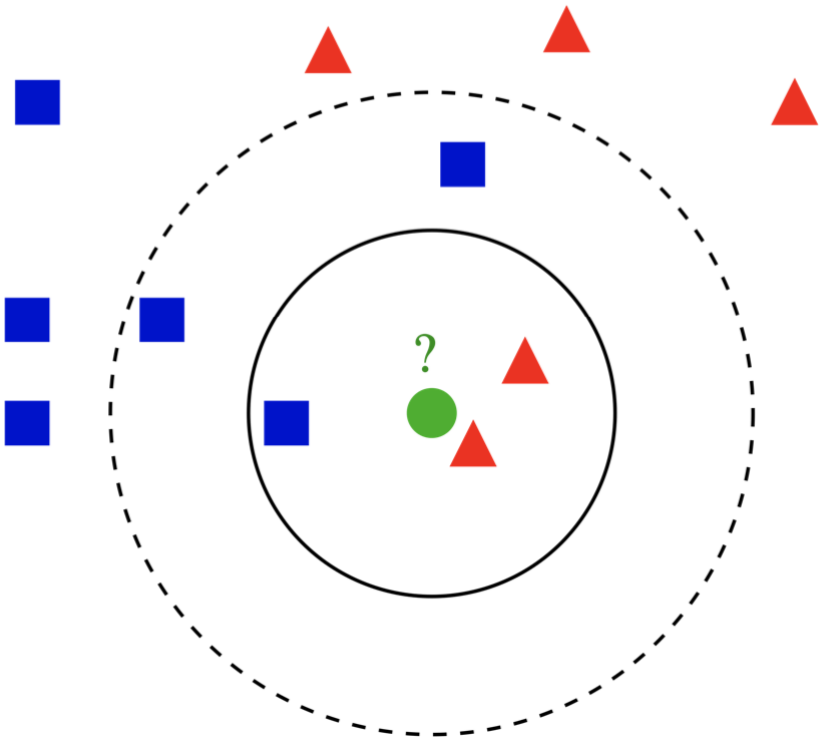


Figura 3.17: Rappresentazione dell'algoritmo k-NN, [46]

Esistono molti modi diversi per calcolare la distanza, poiché è una nozione abbastanza ambigua e la metrica corretta da utilizzare sarà sempre determinata dal set di dati e dall'attività di classificazione. Due popolari, tuttavia, sono la distanza Euclidea e il "Cosine similarity".
La distanza Euclidea è definita come [42]:

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + ... + (x_n - x'_n)^2} \tag{3.30}$$

Quella Euclidea è comunque un esempio di distanza ma in alcuni casi altre distanze come quella di Manhattan, di Chebyshev o di Hamming possono essere più performanti.

Invece con Cosine similarity, piuttosto che calcolare una magnitudine, si utilizza invece la differenza di direzione tra due vettori A e B, che in questo caso corrispondono ai vettori posizione.

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} \tag{3.31}$$

La scelta di una metrica può essere spesso complicata e potrebbe essere meglio utilizzare la Cross Validation per decidere.

In generale comunque, il parametro da fornire all'algoritmo è solo il K, ovvero il numero dei vicini che effettueranno la votazione per classificare il dato incognito [42]. Si può intuire che un K piccolo, limita la regione di una data previsione e costringe il classificatore a considerare meno la distribuzione generale,figura (3.18). D'altra parte, un K maggiore implicherà confini più uniformi come visibile in figura

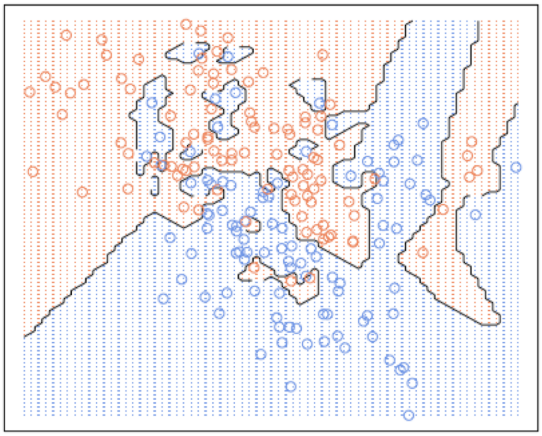


Figura 3.18: Regioni di previsione con K=1, [47]

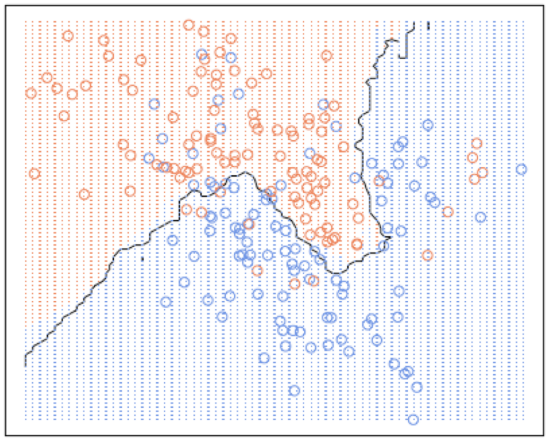


Figura 3.19: Regioni di previsione con K=20, [47]

(3.19), che praticamente si tradurrà in una previsione più resistente ai valori anomali e più legata alla distribuzione generale [42]. Alcuni studi hanno dimostrato che un k minore è più flessibile ed avrà un bias basso ma con una varianza elevata, mentre un k maggiore avrà un confine decisionale più fluido, il che significa una varianza inferiore ma un bias più elevato. In aggiunta al calcolo della distanza, nella selezione delle k istanze, viene utilizzato un valore di peso inversamente proporzionale al quadrato della distanza, in modo da dare maggiore peso alle istanze di training più vicine al punto che rappresenta l'istanza di test. Dopo aver selezionato le k istanze di training più vicine, viene scelto il valore di classe più presente tra le k istanze selezionate e viene assegnato all'istanza di test.

Capitolo 4

Metodologia

Questo capitolo ha lo scopo di spiegare il flusso logico seguito durante questo progetto di tesi, per dare un'idea chiara sulle fasi di:

1. Raccolta, analisi ed elaborazione dei dati.
2. Inversione dell'algoritmo TSVD per simulare i segnali di scattering, ovvero dati sintetici ottenuti del problema "forward" equivalente.
3. Verifica e Validazione della procedura adottata.
4. Costruzione del dataset.
5. Addestramento degli algoritmi di ML con i dati simulati.

Come diretta conseguenza, ciò faciliterà anche la comprensione dei risultati presentati nel prossimo capitolo.

4.1 Raccolta, analisi ed elaborazione dei dati

I dati che sono serviti per farne una prima analisi sono stati elaborati tramite Matlab.
In particolare, i dati di input sono:

- L'Operatore $\mathcal{S}$: e quindi le matrici [U],[V],[Σ] dell'algortimo SVD (2.17);
- Le coordinate x,y,z dei tetraedri;
- Gli indici corrispondenti ai vertici dei tetraedri.

Le matrici [U],[V] e [Σ] sono appunto utili ai fini dell'applicazione dell'algoritmo TSVD (2.19), spiegato nei capitoli precedenti.

In particolare nel caso in esame, l'operatore $\mathcal{S}$ rappresenta il prodotto dei campi di tutte le antenne a 1 GHz, mentre i singular vectors (sinistro e destro) ne descrivono rispettivamente lo spazio dove si andrà a ricostruire la variazione di contrasto $\Delta\chi$ (combinazione lineare delle funzioni di base), e lo spazio dei parametri S.

A disposizione ci sono le coordinate e gli indici dei tetraedri, che ne rappresentano tutto il dominio in modo discreto come in figura (4.1). Ogni coordinata x,y,z corrisponde ad uno dei 4 vertici di ogni tetraedro.

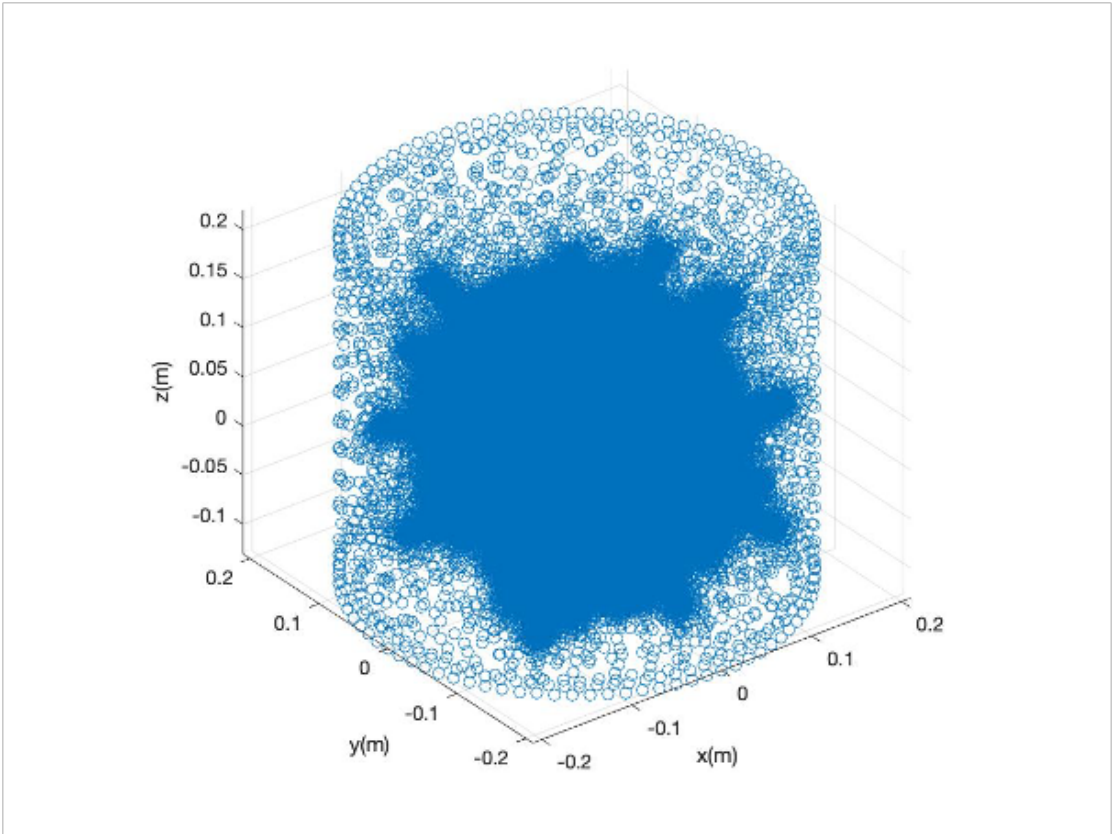


Figura 4.1: Scatter3 Coordinate Tetraedri di tutto il dominio

Le coordinate sono servite per determinare il baricentro di ogni tetraedro (facendone la media delle coordinate corrispondenti ai 4 vertici) e di conseguenza sono state ricavate le coordinate baricentriche. Queste hanno permesso una migliore visualizzazione della regione di interesse (ROI), rappresentata dalla testa, figura (4.2).

Avendo a disposizione tutta la ROI discretizzata, è stato poi possibile costruirne una maschera che è servita a selezionare i punti di interesse, in modo da poter

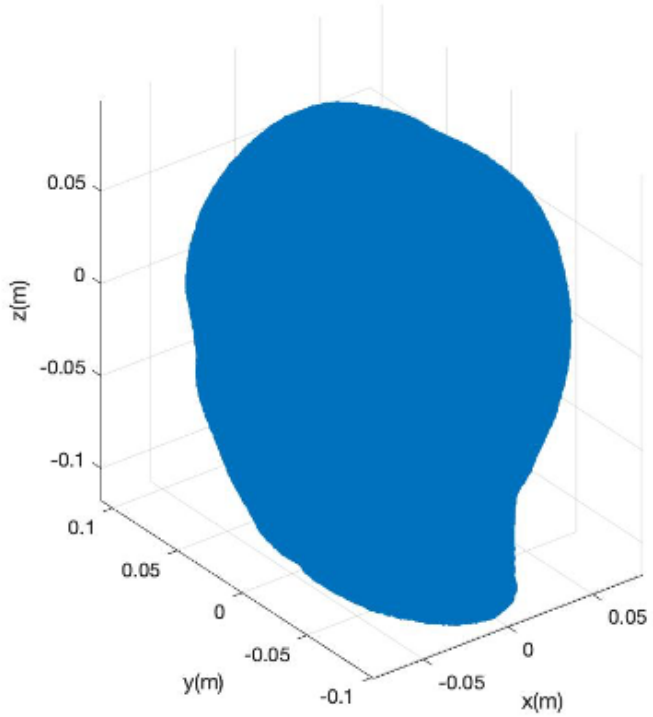


Figura 4.2: Scatter3 Coordinate Baricentriche dei Tetraedri nello spazio ROI

simulare una variazione di contrasto $\Delta\chi$ nel punto desiderato.

La finalità quindi è quella di generare tante possibili configurazioni di ictus, con dimensione e posizione diversa, grazie alla selezione di gruppi di tetraedri (pixel) ai quali si assegneranno valori di contrasto 1-0.

Successivamente, dopo la realizzazione di una variazione di contrasto $\Delta\chi$, è stato possibile simulare i segnali di scattering sfruttandone l'inversione dell'algoritmo TSVD.

4.2 Inversione dell'algoritmo TSVD per simulare i segnali di scattering

Normalmente l'algoritmo TSVD, viene utilizzato per ricostruire un'immagine: vengono difatti utilizzati i segnali di scattering per dedurne la variazione di contrasto

$\Delta\chi$ corrispondente, che a sua volta corrisponderà ad un valore ben preciso di voxel producendone l'immagine.

L'obiettivo della tesi, come accennato nei precedenti capitoli, è quello di valutare se alcuni algoritmi di ML operanti su dati di scattering EM possano rilevare e classificare l'ictus cerebrale.
Tuttavia, è noto che l'allenamento di questi algoritmi richiede un numero molto elevato di campioni (dataset). Di conseguenza questo avrebbe richiesto molto tempo per la realizzazione di un numero elevato di misure sperimentali, ragion per cui sono stati simulati i segnali di Scattering. D'altronde si hanno altri vantaggi come il maggior controllo nel generare i dati sintetici (rispetto alle misure) e alla assenza di disturbi esterni.

In particolare, quindi si è sfruttata l'inversione dell'algoritmo TSVD, dove in sostanza anziché utilizzare quest'ultimo per ricavare la variazione di contrasto $\Delta\chi$ da una variazione ΔS, equazione (2.19), al contrario è stata utilizzata la simulazione di una variazione di contrasto $\Delta\chi$ per ricavare una variazione della matrice di scattering ΔS.

Definendo i coefficienti c_n come:

$$c_n = \frac{1}{\sigma_n} \langle [\Delta S], [u_n] \rangle \tag{4.1}$$

Dalle proprietà della matrici si può scrivere che:

$$\Delta\chi = \sum_{n=1}^{M} \langle [\Delta\chi], [v_n] \rangle [v_n] \tag{4.2}$$

Chiamando $c_n = \langle [\Delta\chi], [v_n] \rangle$, si assume che quest'ultimo sia uguale al coefficiente c_n (4.1):

$$\frac{1}{\sigma_n} \langle [\Delta S], [u_n] \rangle = \langle [\Delta\chi], [v_n] \rangle \tag{4.3}$$

Da questa relazione, scrivendo la proprietà delle matrici anche per ΔS:

$$\Delta S = \sum_{n=1}^{M} \langle [\Delta S], [u_n] \rangle [u_n] \tag{4.4}$$

Si arriva, sfruttando l'equazione (4.4), alla seguente equazione:

$$\Delta S = \sum_{n=1}^{M} \langle [\Delta\chi], [v_n] \rangle \sigma_n [u_n] \tag{4.5}$$

Da questa relazione (4.5), conoscendo $[v_n]$, σ_n, $[u_n]$, date appunto dall'operatore $\mathcal{S}$, e forzando manualmente variazioni di contrasto $\Delta\chi$ (con varie posizioni e dimensioni),

è stata ricavata la matrice ΔS.
Il valore M, ossia l'indice di troncamento dell'algoritmo SVD, è stato scelto pari a 300, poiché dal grafico relativo alla matrice diagonale dei Singolar Values (4.3), si evince che da quell'indice in poi non si otteneva alcuna informazione dalla radice degli autovalori.

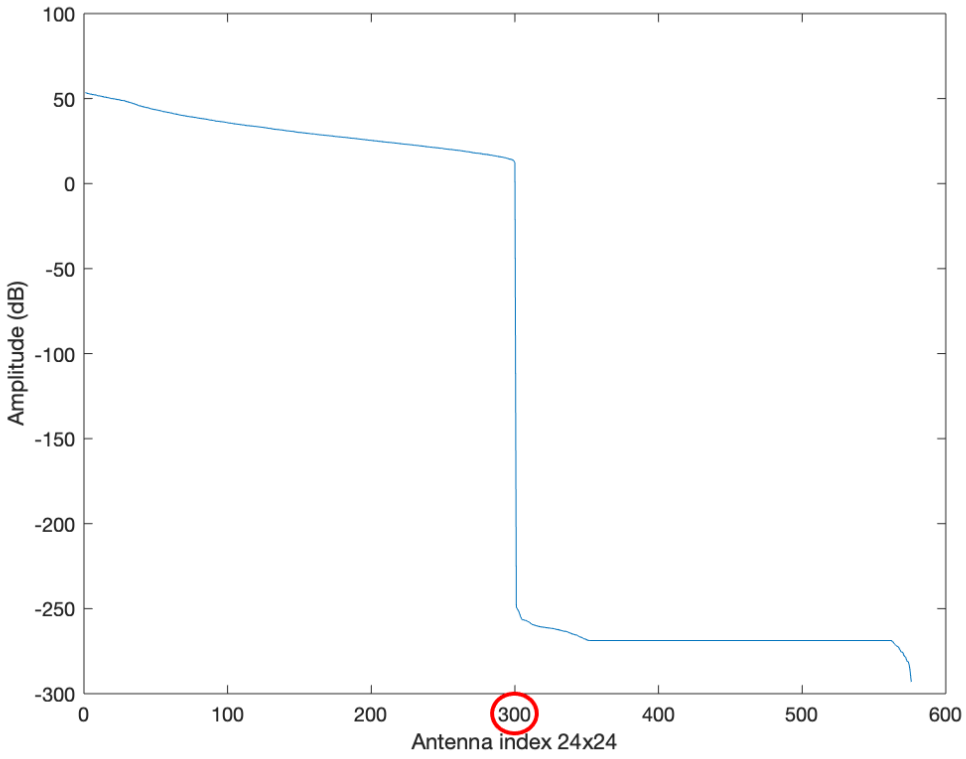


Figura 4.3: Plot della matrice diagonale dei Singular Values.

4.3 Verifica e validazione della procedura adottata

Prima di procedere alla successiva fase, relativa alla creazione del dataset di segnali di scattering, in questo paragrafo viene spiegato com'è stata verificata e poi validata l'equazione (4.5).

4.3.1 Confronto con il caso di misurazione di riferimento

Per il confronto è stato preso in considerazione un caso di riferimento con misurazioni dei parametri S in due istanti differenti, realizzate con simulazioni full-wave del caso con e senza target (non sono dati di misure anche se le simulazioni mimano proprio il sistema di misura composto dal VNA), dove appunto è stata simulata una condizione di ictus di forma sferica con raggio 1.25 cm e coordinate (x=0 m; y=-0.02 m; z=0.05 m).
Le matrici di scattering delle due misurazioni sono riportate in figura (4.4).

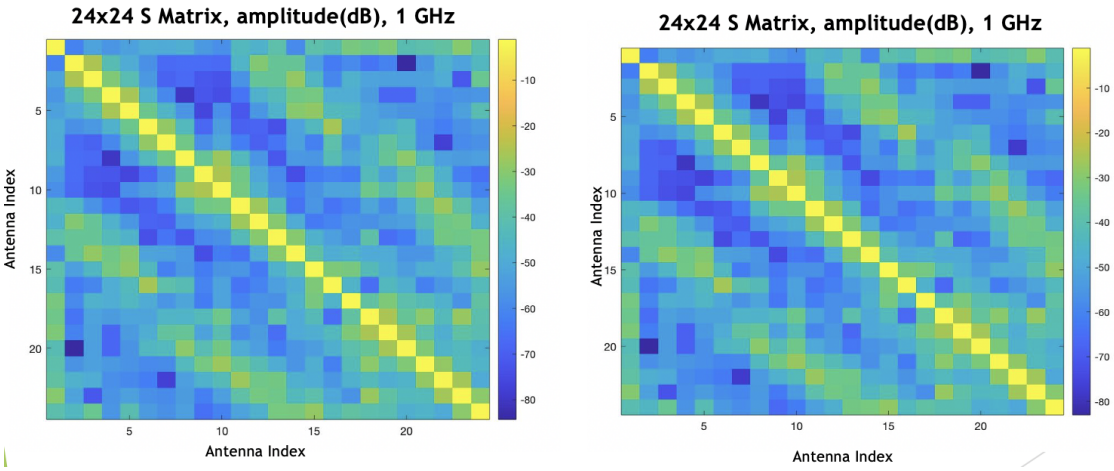


Figura 4.4: Matrici di Scattering in due istanti differenti.

Successivamente è stata ottenuta la matrice di Scattering ΔS data dalla differenza dei parametri S valutata in due istanti di tempo, figura (4.5), che in seguito in questo capitolo è servita per un doppio confronto.

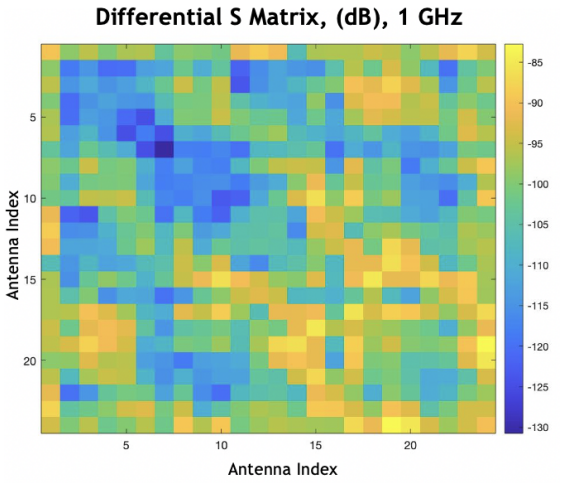


Figura 4.5: Matrice di Scattering del caso di riferimento.

Su Matlab è stata ricreata la stessa scena del caso di riferimento. In particolare, attraverso la ROI composta da tutte le coordinate dei baricentri dei tetraedri e una maschera [1,0], sono stati quindi selezionati solo i punti che ne simulavano un ictus sferico di raggio e coordinate uguali al caso noto.
Da qui è stato possibile fare un primo confronto tra la variazione di $\Delta\chi$ simulata con Matlab e il $\Delta\chi$ del caso di riferimento ricavato della TSVD (2.19), figura (4.6).

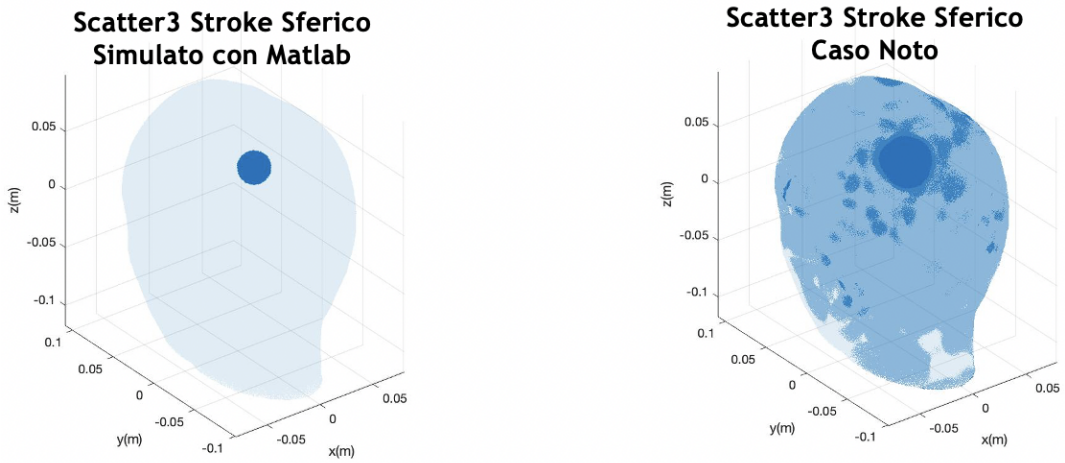


Figura 4.6: Scatter3 della variazione di contrasto $\Delta\chi$ del caso simulato e del caso noto (di riferimento).

In figura (4.7) è possibile visualizzare la stessa rappresentazione di quella precedente

ma con altri strumenti di visualizzazione.

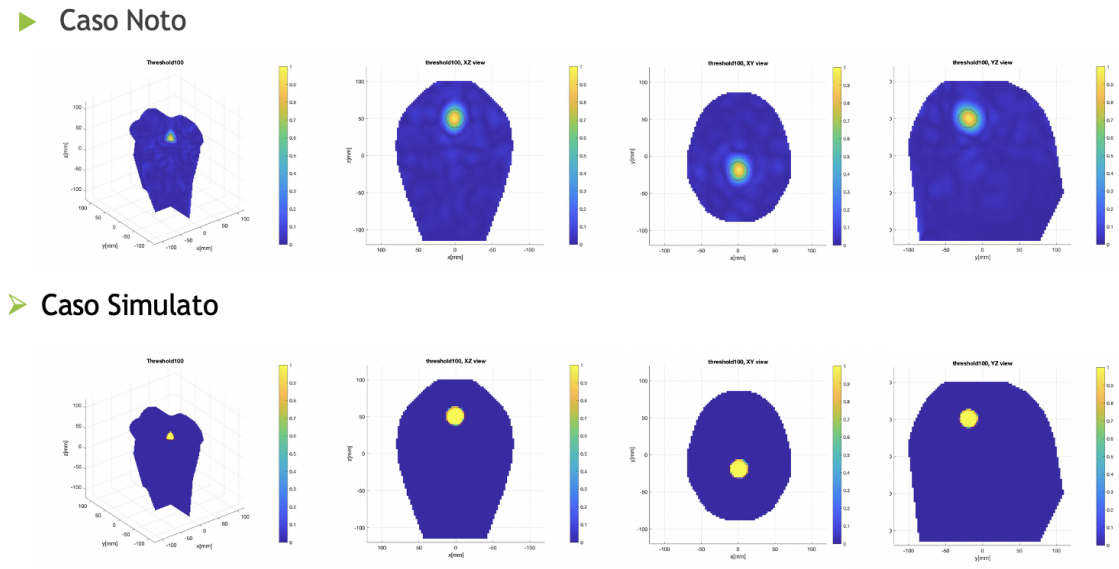


Figura 4.7: Plotslices3D della variazione di contrasto $\Delta\chi$ del caso noto e del caso simulato.

Inoltre è stato fatto il confronto tra le due matrici di Scattering, rispettivamente del caso noto (4.5) e del caso simulato (4.8), dove la matrice ΔS è stata ottenuta dall'equazione (4.5).

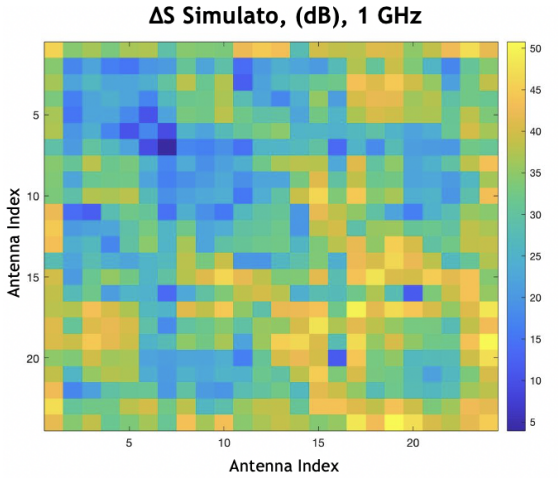


Figura 4.8: Matrice di scattering ΔS simulata con l'inversione dell'algoritmo TSVD.

Dalle figure (4.5) e (4.8), si può notare che la matrice ΔS simulata ha lo stesso pattern del caso di riferimento a meno di un fattore di scala, quindi la prima verifica è stata validata correttamente.

4.3.2 Controllo TSVD: dalla matrice di scattering simulata alla variazione di contrasto

Una volta simulata la variazione di contrasto $\Delta\chi$ con una maschera con contrasto complesso (di 0.4533 + j0.1799, che ne simula la variazione di contrasto causata da un coagulo di sangue) e implementata l'equazione (4.5) che dà origine alla matrice ΔS, si vuole verificare che il processo inverso dalla matrice ΔS appena ottenuta alla variazione di contrasto $\Delta\chi$ mediante l'equazione (2.19), dia origine alla variazione di contrasto $\Delta\chi$ simulata originariamente.

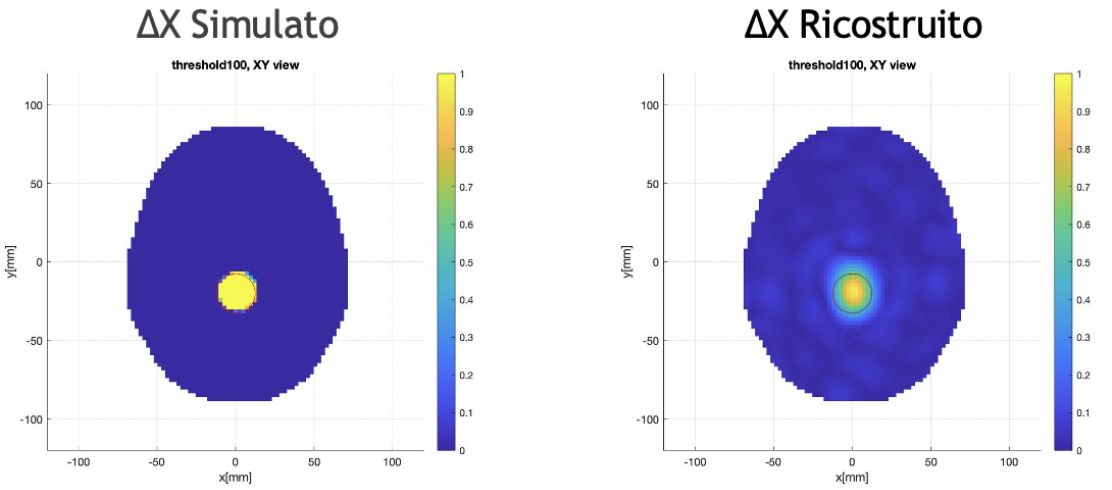


Figura 4.9: Plotslices sul piano XY della variazione di contrasto $\Delta\chi$, simulata e ricostruita.

Dalla figura (4.9) si può notare che la ricostruzione della variazione di contrasto $\Delta\chi$ è quindi validata correttamente confrontandola alla variazione di contrasto $\Delta\chi$ simulata.

4.4 Costruzione del dataset

Finita la fase di validazione della procedura adottata per simulare i segnali di scattering, è stato possibile passare alla fase successiva, che riguarda la costruzione del dataset.

In particolare, è stato creato un dataset bilanciato, ossia con lo stesso numero di dati con target positivo (Ictus/Pallina Presente) e con target negativo (Ictus/Pallina Assente).
In definitiva saranno costruite 9 classi, spiegate di seguito in questo paragrafo.

In prima analisi è stato creato un dataset ridotto di 600 features, successivamente è stato creato un dataset con 12000 features per migliorare il processo di apprendimento.

4.4.1 Costruzione del dataset con target positivo

Per la creazione del dataset con target positivo, è stata ricreata la situazione con una variazione di contrasto $\Delta\chi$ simile alla corrispondente variazione di contrasto che si avrebbe con un coagulo di sangue, in modo da simulare una situazione in cui appunto è presente una forma di ictus; sono stati presi come valori di riferimento la costante dielettrica relativa ε_r e la conduttività elettrica σ ad 1 GHz, consultando [20]. Inoltre, per rendere più realistica ogni simulazione effettuata è stato aggiunto il rumore gaussiano bianco ad 80 dB.

Successivamente, per la realizzazione del dataset con le caratteristiche sopra elencate, sono state supposte quattro possibili posizioni dell'ictus rispetto agli assi x ed y, lasciando invariato l'asse z con coordinata z=0.05:

- Right (R) con coordinata x=0.02;
- Left (L) con coordinata x=-0.02;
- Frontal (F) con coordinata y=0.02;
- Back (B) con coordinata y=-0.02.

In figura (4.10 sono rappresentate le quattro posizioni ipotizzate nello spazio 3D.

Inoltre, sono state ipotizzate ulteriori informazioni riguardo la dimensione dell'ictus; difatti supponendo un raggio con range da 0.009 a 0.05 m, per valori inferiori a 0.0295 m (valore di soglia corrispondente alla metà del range di valori) la dimensione sarà classificata come Small (S), viceversa per valori maggiori la dimensione sarà classificata come Large (L).
Pertanto, combinando le quattro possibili posizioni con la dimensione che l'ictus può assumere nello spazio, sono state realizzate 8 classi:

1. S_LF
2. L_LF
3. S_LB

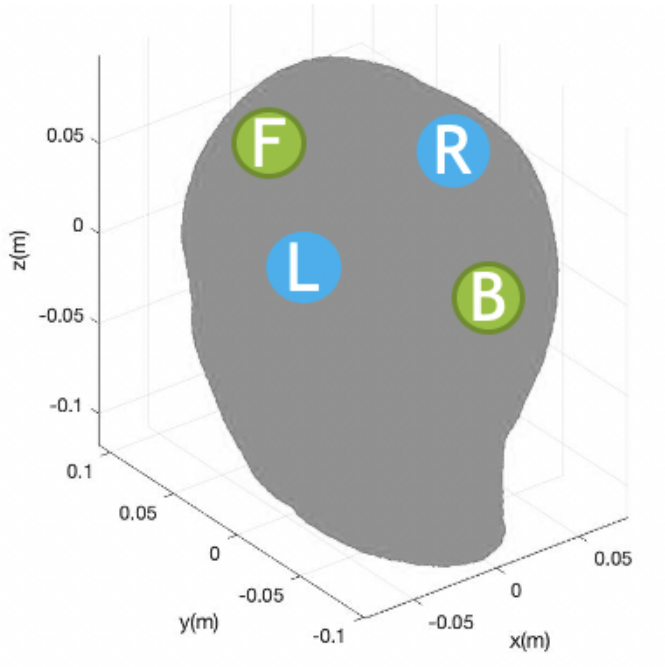


Figura 4.10: Simulazione delle quattro ipotetiche posizioni (R,L,F,B) assunte dall'ictus.

4. L_LB

5. S_RF

6. L_RF

7. S_RB

8. L_RB

Nelle figure (4.11),(4.12),(4.13),(4.14), è possibile visualizzare la variazione di contrasto $\Delta\chi$ simulata con quattro corrispondenti posizioni (con dimensione variabile), nei piani XY, XZ, YZ.

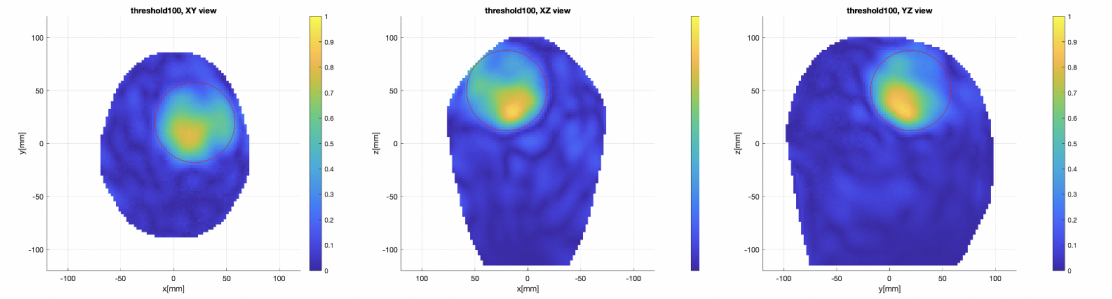


Figura 4.11: Rappresentazione nei piani XY,YZ,XZ della variazione di contrasto $\Delta\chi$ simulata con configurazione "Right Frontal" e raggio=0.0378 m, pertanto appartenente alla classe L_RF.

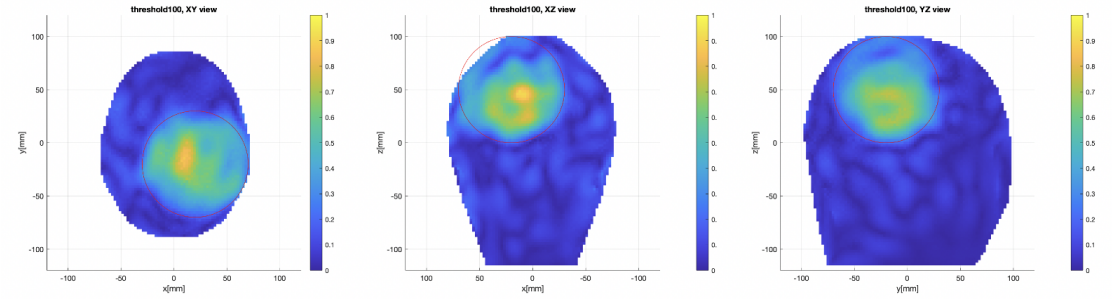


Figura 4.12: Rappresentazione nei piani XY,YZ,XZ della variazione di contrasto $\Delta\chi$ simulata con configurazione "Right Back" e raggio=0.05 m, pertanto appartenente alla classe L_RB.

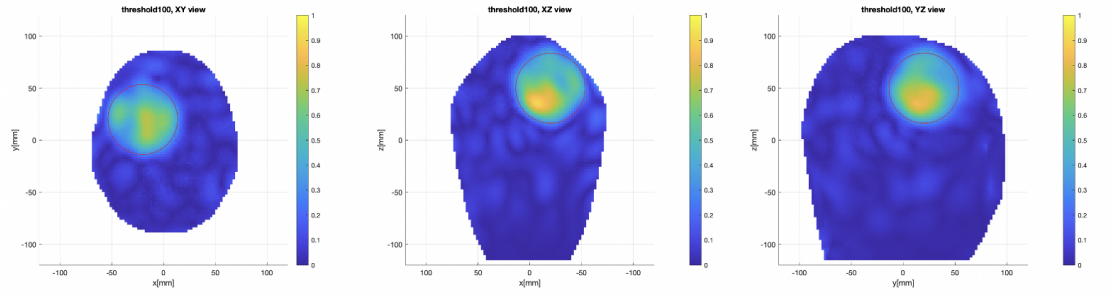


Figura 4.13: Rappresentazione nei piani XY,YZ,XZ della variazione di contrasto $\Delta\chi$ simulata con configurazione "Left Frontal" e raggio=0.0336 m, pertanto appartenente alla classe L_LF.

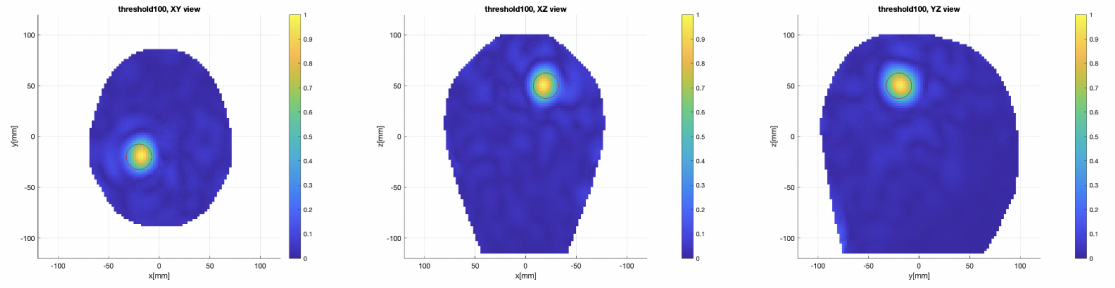


Figura 4.14: Rappresentazione nei piani XY,YZ,XZ della variazione di contrasto $\Delta\chi$ simulata con configurazione "Left Back" e raggio=0.0125 m, pertanto appartenente alla classe S_LB.

Da queste assunzioni, utilizzando l'inversione dell'algoritmo TSVD (4.5), sono stati quindi ottenuti i segnali di scattering ΔS che costituiranno i dati del target positivo.

I segnali di scattering realizzati, sono stati soggetti ad un ulteriore verifica, in particolare, per ogni classe sono state rappresentate le variazioni massime, minime e medie corrispondenti alle misure simulate su ogni coppia di antenna.

Nei seguenti grafici, viene rappresentato sull'asse y il valore di ampiezza espresso in dB della matrice di scattering e sull'asse x di fatto l'indice della coppia di antenne 24x24 trasformato in un indice che va da 1 a 576. Per ognuna delle coppie (riga,colonna) si hanno valori di misura diversi a seconda del campione considerato. Infatti ognuno dei 576 valori della matrice scattering rappresenta una misura di trasmissione di segnale tra un'antenna e un'altra. Se questa non variasse mai nelle tot misure relative ad ogni classe (o variasse molto poco), significherebbe che nel momento in cui si utilizzeranno gli algoritmi di ML sarà facile "apprendere" il loro comportamento ottenendone sempre un'elevata accuratezza.
Invece se variasse abbastanza significherebbe che esiste una buona correlazione con la scena che si sta misurando (diverso tipo di target) e quindi gli algoritmi di ML apprenderebbero difficilmente. In questo caso un buon risultato di classificazione significherebbe che gli algoritmi funzionano molto bene. L'algoritmo di ML cioè riesce a generalizzare bene dal training set al test set.
Inoltre, viene fatta anche una rappresentazione 3D con gli assi x ed y corrispondenti alle coppie di antenne con indici da 1 a 24, mentre sull'asse z al valore di ampiezza espresso in dB della matrice di scattering.

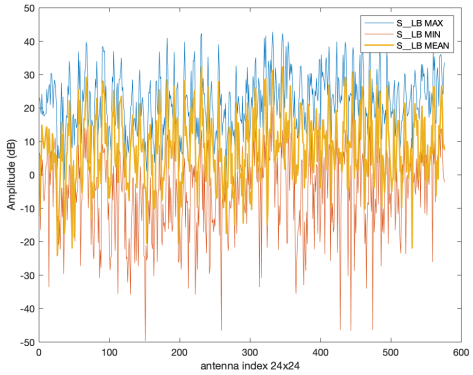


Figura 4.15: Grafico 2D classe S_LB

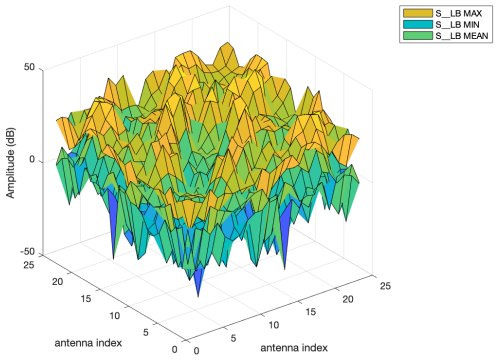


Figura 4.16: Grafico 3D classe S_LB

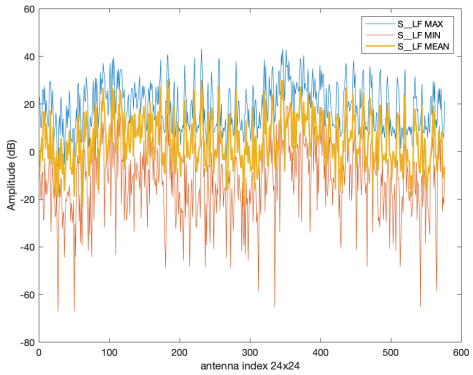


Figura 4.17: Grafico 2D classe S_LF

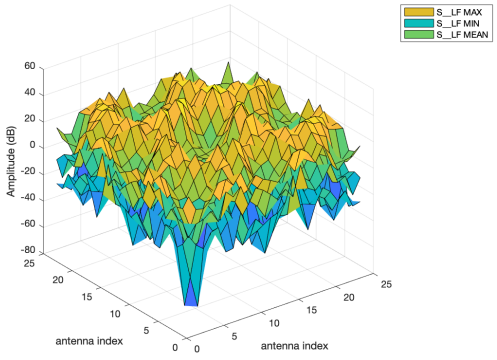


Figura 4.18: Grafico 3D classe S_LF

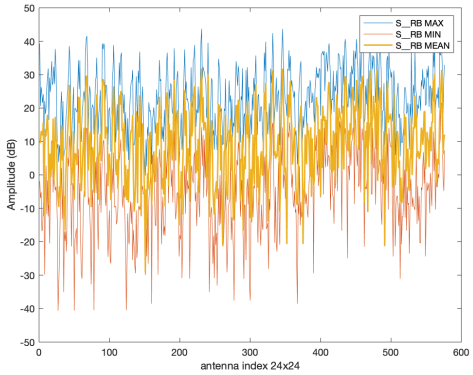


Figura 4.19: Grafico 2D classe S_RB

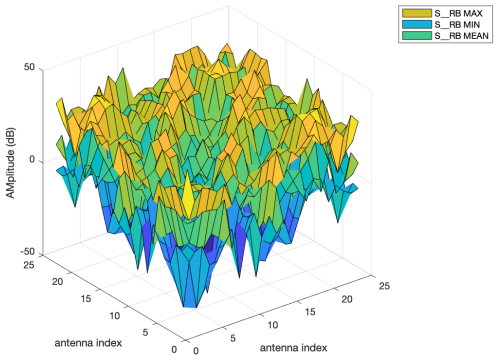


Figura 4.20: Grafico 3D classe S_RB

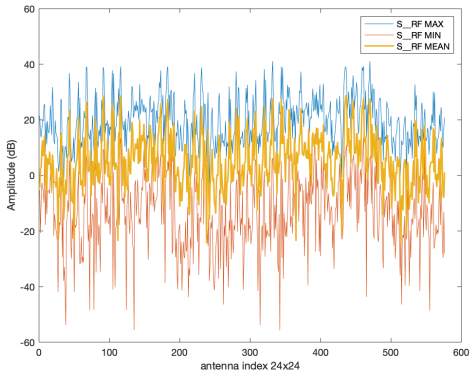


Figura 4.21: Grafico 2D classe S_RF

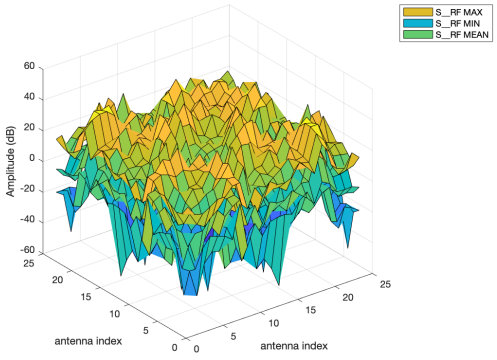


Figura 4.22: Grafico 3D classe S_RF

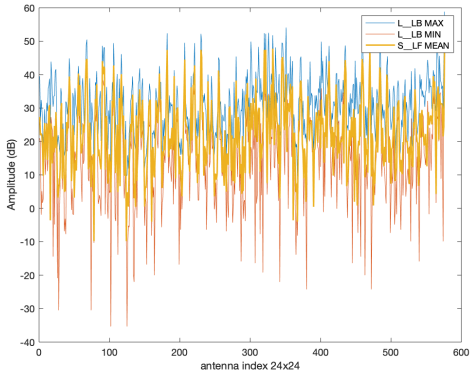


Figura 4.23: Grafico 2D classe L_LB

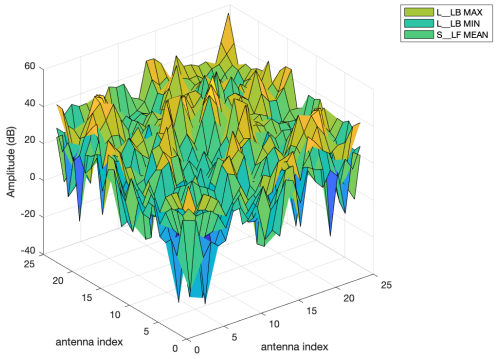


Figura 4.24: Grafico 3D classe L_LB

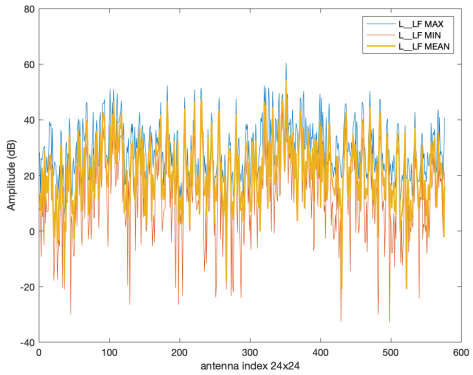


Figura 4.25: Grafico 2D classe L_LF

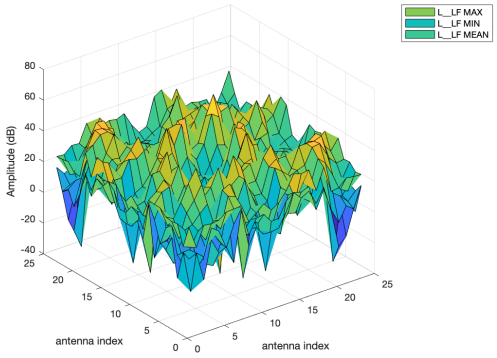


Figura 4.26: Grafico 3D classe L_LF

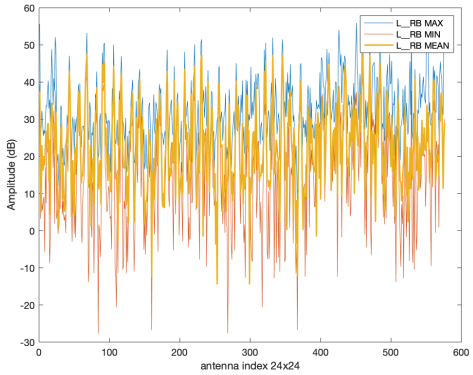


Figura 4.27: Grafico 2D classe L_RB

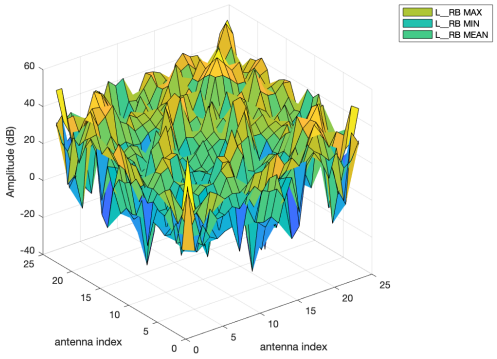


Figura 4.28: Grafico 3D classe L_RB

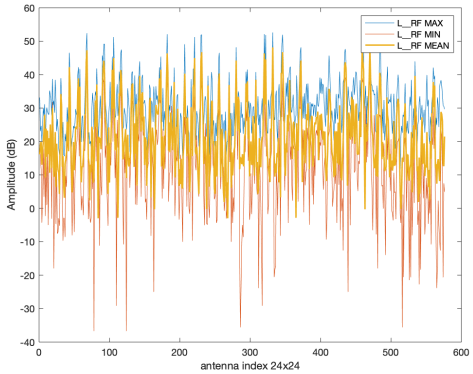


Figura 4.29: Grafico 2D classe L_RF

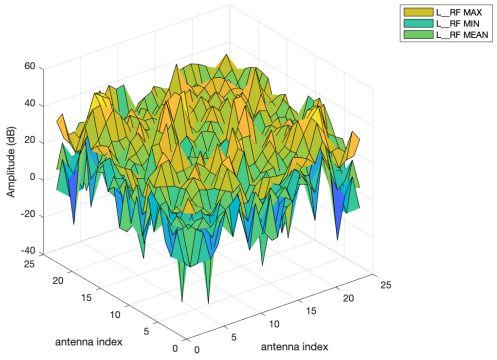


Figura 4.30: Grafico 3D classe L_RF

Con la presente analisi si è voluto verificare che i dati per ogni classe fossero ben distribuiti e che quindi le istanze non fossero simili tra di loro. Se così non fosse e quindi i dati non sarebbero ben distribuiti e fossero tutti molto simili, non ci sarebbe molta differenza neppure tra training e test set nel momento in cui si testeranno gli algoritmi di ML, e quindi questo comporterebbe il fatto che un buon allenamento porterebbe ad un buon risultato di accuratezza anche nel test set.
Dai grafici infatti, si può notare come ci sia un certo distacco tra i valori massimi e minimi, e che l'andamento medio ricade proprio nel mezzo delle due distribuzioni.

4.4.2 Costruzione del dataset con target negativo

Per la creazione di dati con target negativo, è stata ricreata la situazione con una variazione di contrasto $\Delta\chi$ molto bassa in modo da simulare una situazione in cui non è presente l'ictus.

In particolare, attraverso la consultazione, dal sito di riferimento [20], dei valori tipici di costante dielettrica relativa ε_r e conduttività elettrica σ ad 1 GHz, sono stati considerati i valori corrispondenti alla materia grigia e alla materia bianca, ed infine è stata fatta una media. Il tutto è stato riscritto sfruttando la seguente formula:

$$\varepsilon = \varepsilon_0 \varepsilon_r - j\frac{\sigma}{w} \tag{4.6}$$

Successivamente è stato calcolata la variazione di contrasto $\Delta\chi$ dalla formula (2.14), con un ε_b dato dalla media dei valori ε dei tessuti cerebrali, ottenendo la variazione di contrasto desiderata.
Anche in questo caso, per rendere più realistica la simulazione è stato aggiunto del rumore gaussiano bianco con 80 dB.

Nelle figura (4.31), è possibile visualizzare la variazione di contrasto $\Delta\chi$ simulata nei piani XY, XZ, YZ.

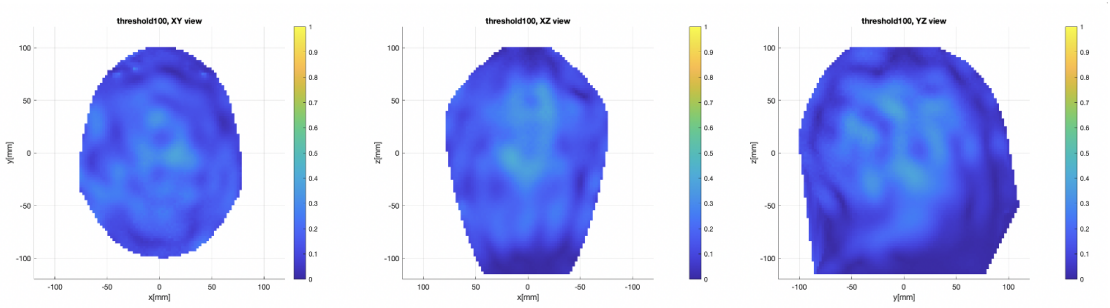


Figura 4.31: Rappresentazione nei piani XY,YZ,XZ della variazione di contrasto $\Delta\chi$ simulata per ottenere i dati con target negativo.

Da questa assunzione, utilizzando l'inversione dell'algoritmo TSVD (4.5), sono stati quindi ottenuti i segnali di scattering ΔS che costituiranno i dati del target negativo. Questa sarà pertanto la nona classe, nominata come classe "N".

I segnali di scattering ottenuti sono stati soggetti ad un ulteriore verifica, in modo analogo come per i dati del target positivo; in particolare, per ogni classe sono state rappresentate le variazioni massime, minime e medie corrispondenti alle misure simulate su ogni coppia di antenna.

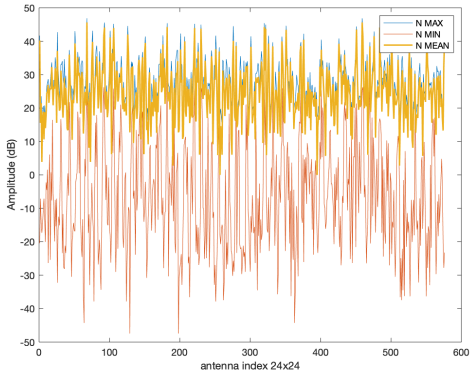


Figura 4.32: Grafico 2D classe N

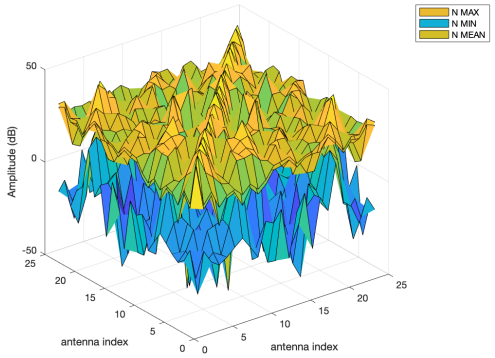


Figura 4.33: Grafico 3D classe N

Anche in questo caso, con tale analisi si è verificato che i dati per ogni classe fossero ben distribuiti e che quindi le istanze non fossero simili tra di loro.
Dai grafici infatti si può notare come ci sia un certo distacco tra i valori massimi e minimi, e che l'andamento medio ricade pressoché nel mezzo delle due distribuzioni.

4.5 Addestramento degli algoritmi di ML con il dataset realizzato

Una volta completata la fase di costruzione del dataset, i dati sono stati raccolti e raggruppati in delle cartelle (dove ciascuna cartella corrisponde ad una rispettiva classe). In particolare, per agevolare una buona lettura dei dati in formato NumPy, i dati complessi sono stati riscritti in modo che la parte immaginaria di ogni campione susseguisse la parte reale dello stesso separata da una virgola.
Successivamente, i dati sono stati caricati su Google Colab.

4.5.1 Caricamento dei dati su Google Colab

I dati sono stati caricati correttamente sulla piattaforma online Google Colab, e per prima cosa è sta effettuata la conversione in formato NumPy, generando arrays float in quanto più facili da manipolare con le librerie di Sklearn.

4.5.2 Preprocessing

Prima di procedere con il preprocessing, il dataset è stato suddiviso in training set e test set; in particolare, il primo set di campioni serve per l'addestramento

del modello, mentre il secondo set di campioni di dati viene utilizzato per fornire una valutazione imparziale del modello finale costruito con il training set. La dimensione del training set è stata sempre scelta come l'80 % della dimensione del dataset. Questa suddivisione in genere viene fatta per verificare un eventuale overfitting o underfitting che potrebbe essere causata dal training set.

Una volta suddivisi i dati in training set e test set, in questa fase di progetto, attraverso l'analisi dei campioni, è stato stabilito se il preprocessing fosse necessario oppure no.
In generale, il preprocessing viene effettuato per consentire un migliore processo di addestramento da parte degli algoritmi di ML al fine di ottenere una buona accuratezza nella classificazione.
Per avere una valutazione concreta, sono stati calcolati i seguenti valori:

- il massimo dei massimi;
- il minimo dei massimi;
- il massimo dei minimi;
- il minimo dei minimi.

Grazie a questi valori è stato possibile comprendere la distribuzione dei dati e quindi stabilire se i dati avessero bisogno del preprocessing.

Pertanto, una volta stabilito di procedere con il preprocessing, si utilizzano le librerie di Sklearn, effettuando feature standardization (3.1) e feature reduction con PCA.

4.5.3 Learning Curves

La Learning Curve, è una rappresentazione grafica che mostra lo "score" (in questo caso rappresentato dall'accuratezza) di validazione e d'addestramento di uno stimatore per un numero variabile di campioni di training. È uno strumento per scoprire quanto un modello di apprendimento automatico trae vantaggio dall'aggiunta di più dati di addestramento e se lo stimatore soffre maggiormente di un errore di varianza o di un errore di bias. Se lo score di validazione e di addestramento convergono a un valore troppo basso con l'aumentare delle dimensioni del training set, non si trae molto vantaggio con un set di training più ampio.
La Learning Curve è utile per molti scopi, tra cui il confronto di diversi algoritmi, la scelta dei parametri del modello durante la progettazione, la regolazione dell'ottimizzazione per migliorare la convergenza e la determinazione della quantità di dati utilizzati per l'addestramento.
Si rappresenta graficamente l'esperienza del modello come il numero di esempi di addestramento utilizzati per l'apprendimento o il numero di iterazioni utilizzate

nell'addestramento del modello. In sintesi, determina uno "Score" di test e di training attraverso la Cross Validation per diverse dimensioni del training set.

In genere, una curva di apprendimento mostra come cambia lo score, in questo caso rappresentato dall'errore, all'aumentare delle dimensioni del training set. Man mano che si aumenta la dimensione del training set, il modello non può più adattarsi perfettamente al training set. Quindi l'errore di addestramento diventa maggiore. Tuttavia, il modello viene addestrato su più dati, quindi riesce ad adattarsi meglio al set di validazione. Pertanto, l'errore di validazione diminuisce.

Questo indica qualcosa di estremamente importante: l'aggiunta di più punti di dati di addestramento non porterà a modelli significativamente migliori.

Se l'errore di addestramento è molto basso, significa che i dati di addestramento sono adattati molto bene dal modello stimato. Se il modello si adatta molto bene ai dati di addestramento, significa che ha un bias basso rispetto a quel set di dati. Se l'errore di addestramento è alto, significa che i dati di addestramento non sono adattati abbastanza bene dal modello stimato. Se il modello non si adatta bene ai dati di addestramento, significa che ha un bias elevato rispetto a quel set di dati.

Esaminando il divario tra la curva di apprendimento di validazione e la learning curve del training set. Esaminando l'errore di addestramento: il suo valore e la sua evoluzione all'aumentare delle dimensioni del training set.

4.5.4 Addestramento, classificazione e metriche di performance

La fase successiva, è quella di addestramento dei dati precedentemente elaborati, con gli algoritmi di ML.
Gli algoritmi di ML supervisionato utilizzati, come più volte accennato, sono stati il MLP, il SVM, e il k-NN.

Per ogni algoritmo è stata fatta una Grid-Search per la ricerca degli iperparametri che hanno permesso una migliore accuratezza nella classificazione finale.
Nella scelta degli iperparametri per il MLP, sono stati scelti i parametri:

- *Activation*: che rispecchia la funzione di attivazione dei layer nascosti;
- *Alpha*: che è un parametro di regolarizzazione, (penalità L2);
- *Hidden_layer_sizes*: che rispecchia il numero di layers nascosti e di quanti neuroni ogni layer è composto;
- *Learning_rate*: programma il rate di apprendimento per gli aggiornamenti dei pesi;

- *Solver*: che è il risolutore per l'ottimizzazione dei pesi.

Nella scelta degli iperparametri per il SVM, sono stati scelti i parametri:

- *Kernel*: che specifica il tipo di kernel utilizzato dall'algoritmo;
- *C*: che è un parametro di regolarizzazione. L'intensità della regolarizzazione è inversamente proporzionale a C.

Nella scelta degli iperparametri per il k-NN, è stato scelto il parametro:

- *n_neighbors*: che corrisponde al numero di neighbors.

Il numero massimo di iterazioni (*max_iter*) per tutti gli algoritmi di ML, non è stato scelto come iperparametro poiché la convergenza veniva raggiunta con un minimo di iterazioni, pertanto è stato lasciato il parametro di default.

Hypermarameters MLP	
Activation	'tanh', 'relu'
Alpha	0.0001, 0.05
Hidden_layer_sizes	[1000],[1000,500],[1000,500,250]
Learning_rate	'constant', 'adaptive'
Solver	'sgd', 'adam'

Tabella 4.1: Iperparametri MLP

Hypermarameters SVM	
Kernel	'linear', 'poly', 'rbf', 'sigmoid'
C	1, 2, 3, 300, 500

Tabella 4.2: Iperparametri SVM

Hypermarameters k-NN	
n_neighbors	2, 3, 4, 5, 6, 7, 8

Tabella 4.3: Iperparametri k-NN

La scelta degli iperparametri che aumentavano maggiormente le performance degli algoritmi sono elencate nelle tabelle (4.1), (4.2), (4.3).

Dopo la scelta degli iperparametri dai quali si otteneva una maggiore accuratezza nella classificazione, si è proseguito con l'addestramento del modello.

Inoltre, per valutare e allo stesso tempo validare le performance di classificazione degli algoritmi, in modo da farne un confronto tra gli stessi, è stata utilizzata la metrica Acc_Score, calcolando per ogni algoritmo la Confusion matrix, realizzandone curva ROC (con il parametro AUC), e curva Precision-Recall.

Nel capitolo successivo verranno presentati e poi discussi i risultati ottenuti con questi algoritmi per ogni seguente metrica adottata.

Capitolo 5

Risultati e Discussione

In questo capitolo vengono illustrati e discussi i risultati ottenuti con gli algoritmi di Machine Learning.

Come prima analisi di studio è stato testato un dataset ridotto con 600 features, discutendone i risultati relativi all'addestramento con questi dati.
Successivamente verranno presentati i risultati relativi ad un dataset con 12000 features, che, come ci si potrà aspettare, gli algoritmi avranno delle performance migliori dal momento che aumentando il numero di campioni per l'addestramento questi funzioneranno in maniera più ottimale.

5.1 Risultati dataset ridotto

Il dataset con 600 features è formato rispettivamente da 300 campioni corrispondenti al target positivo (presenza dell'ictus) e 300 campioni corrispondenti al target negativo (assenza dell'ictus).

Innanzitutto, gli algoritmi sono stati testati suddividendo i dati in sole due classi (positiva e negativa), successivamente è stato affrontato il problema multiclass.
Questa procedura è stata adottata per verificare che gli algoritmi funzionino correttamente nella classificazione con i dati realizzati.

Per stabilire se fosse necessario il preprocessing, sono stati calcolati i parametri precedentemente descritti nel paragrafo "Preprocessing" relativo al capitolo "Metodologia", tabella (5.1).

Massimo dei Massimi	812007.32
Minimo dei Massimi	0.000199
Massimo dei Minimi	-0.000189
Minimo dei Minimi	-465339.17

Tabella 5.1: Valori massimi e minimi della distribuzione dei dati.

Analizzando i valori nella tabella (5.1), si evince che la distribuzione dei dati ha una media non nulla con una forte varianza, dunque, è stato necessario procedere con il preprocessing con feature standardization. I nuovi dati standardizzati, avranno media nulla e variabilità unitaria.
Successivamente, è stata effettuata la feature reduction con PCA per estrarre le componenti principali; da un numero di componenti pari a 1152 (poiché ogni valore dei 576 valori è un numero complesso formato da parte reale e da parte immaginaria, dunque suddividendo tali valori prendendo ogni valore a sé si avranno 576x2 valori), dopo la PCA il numero di componenti principali è stato pari ad 11.

5.1.1 Risultati dataset ridotto con problema binario

Dopo aver suddiviso i dati in due classi e di conseguenza in due cartelle, i dati sono stati importati con le funzioni delle libreria di sklearn (*sklearn. datasets._loadfiles*). Successivamente, è stato effettuato il preprocessing.

Una volta ultimato il preprocessing, si è passati alla ricerca degli iperparametri ideali per ogni algoritmo. In particolare sono stati scelti i seguenti parametri per ogni algoritmo, elencati in tabella (5.2).

Hypermarameters MLP	
Activation	'relu'
Alpha	0.0001
Hidden_layer_sizes	[1000,500]
Learning_rate	'adaptive'
Solver	'adam'
Hypermarameters SVM	
Kernel	'rbf'
C	300
Hypermarameters k-NN	
n_neighbors	2

Tabella 5.2: Iperparametri MLP, SVM, k-NN.

Con questi iperparametri è stata ottenuta l'accuratezza nella classificazione con i dati training e con quelli di test, elencate in tabella (5.3.

Accuracy_score		
Algoritmo	**Accuratezza Training Set**	**Accuratezza Test Set**
MLP	1	1
SVM	0.995	1
k-NN	1	1

Tabella 5.3: Accuracy_score MLP, SVM, k-NN per classe binaria.

Per visualizzare le performance di ogni algoritmo, è stata realizzata la Confusion Matrix; per tutt'e tre è stata ottenuta la stessa Confusion Matrix, figura (5.1).

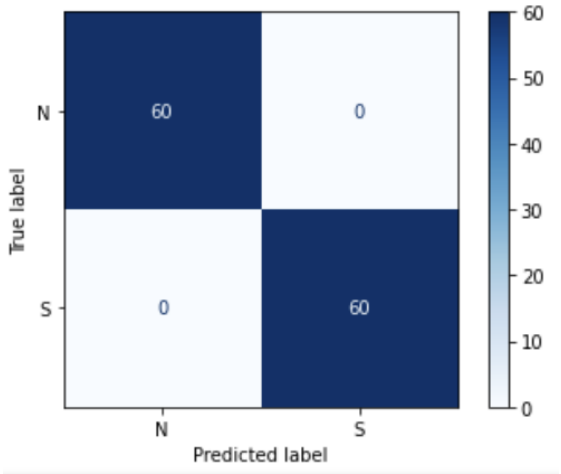


Figura 5.1: Confusion matrix binaria per MLP, SVM, k-NN.

Dalla figura (5.1), si può osservare che, come ci si può aspettare dai valori di accuratezza, è riempita solamente la diagonale della matrice (dunque solamente VP e TN); pertanto, la classificazione è ottima.
Di conseguenza, anche Precision e Recall per tutti e tre gli algoritmi sono al 100%.

Per quanto riguarda le curve ROC e Precision-Recall, per tutti e tre gli algoritmi, sono stati ottenuti gli stessi andamenti, figura (5.2).

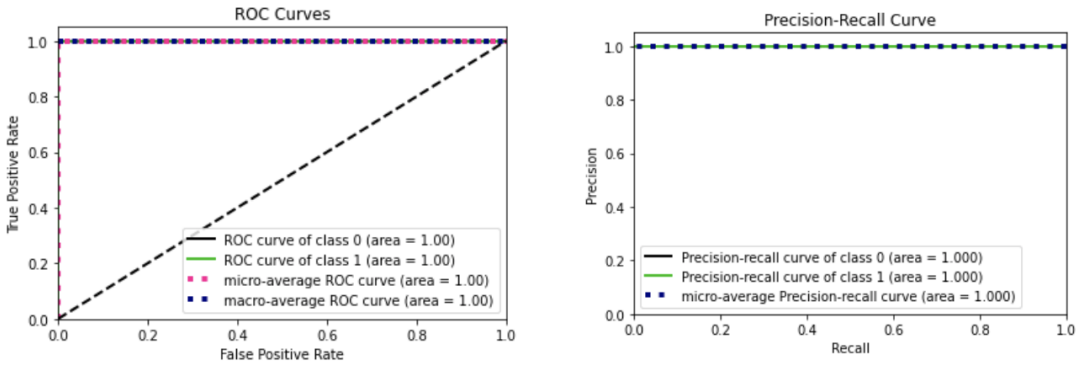


Figura 5.2: Curve ROC e Precision-Recall per classi binaria.

Infatti, come si può notare dal grafico (5.2), è possibile notare che le curve ROC e Precision-Recall per ogni classe (insieme all'andamento Micro e Macro Average) hanno il comportamento ideale di un classificatore con Precision e Recall pari ad 1. Pertanto è stato ampiamente dimostrato che tutti e tre gli algoritmi funzionano

abbastanza bene con un dataset con classi binarie. Il prossimo paragrafo illustrerà i risultati ottenuti con le classificazione multiclass.

5.1.2 Risultati dataset ridotto con problema multiclass

É importante ribadire che, a causa di ipotesi errate non coerenti con quanto supposto durante la costruzione del dataset con target positivo, sono state realizzate solamente sei classi e non otto come inizialmente ipotizzato. Pertanto, le classi S_RF e L_RF non sono state realizzate.

Dopo questa premessa, avendo suddiviso i dati in sette classi e quindi in sette cartelle, attraverso il caricamento dei dati con le funzioni delle libreria di Sklearn (*sklearn.datasets._loadfiles*), successivamente è stato effettuato il preprocessing dei dati.

Una volta ultimato il preprocessing, si è passati alla ricerca degli iperparametri ideali per ogni algoritmo mediante la Grid Search. In particolare, per ciascuno degli algoritmi, sono stati ottenuti i seguenti parametri elencati in tabella (5.4).

Hypermarameters MLP	
Activation	'relu'
Alpha	0.005
Hidden_layer_sizes	[1000,500,250]
Learning_rate	'constant'
Solver	'adam'
Hypermarameters SVM	
Kernel	'linear'
C	300
Hypermarameters k-NN	
n_neighbors	7

Tabella 5.4: Iperparametri MLP, SVM, k-NN per multiclass.

Per comprendere il processo di apprendimento di ogni algoritmo utilizzato, sono state realizzate le Learning Curves.

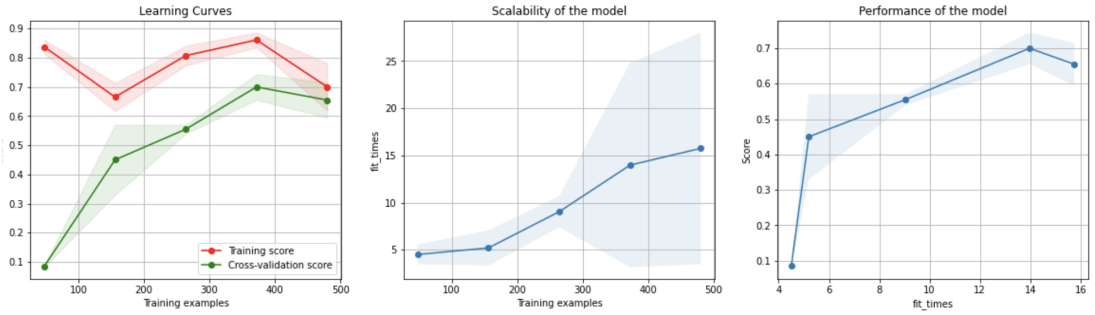


Figura 5.3: Learning Curves per MLP.

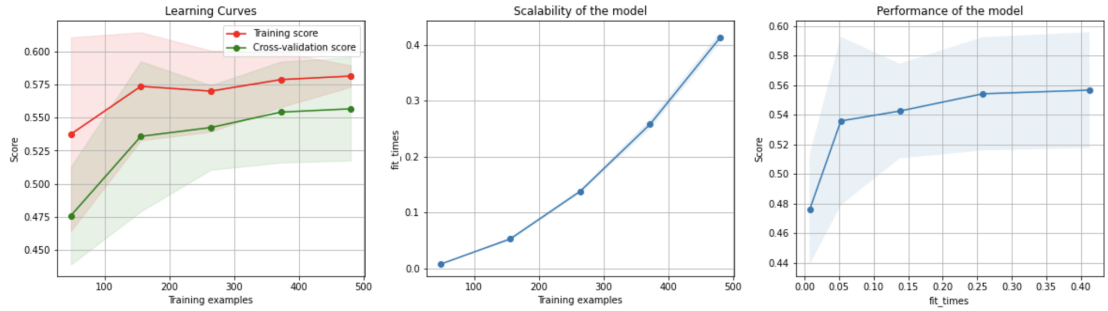


Figura 5.4: Learning Curves per SVM.

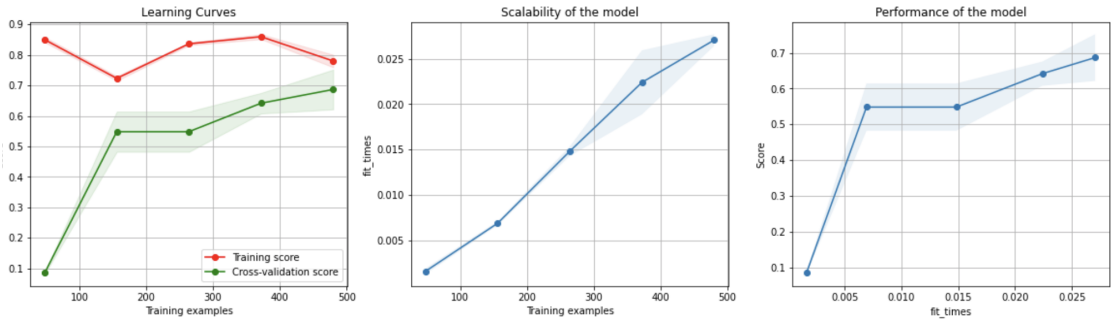


Figura 5.5: Learning Curves per k-NN.

Dalle figure (5.3), (5.4), (5.5), si evince che all'aumentare delle dimensione del training set, osservando la curva di training e la curva di validazione, si ha un bias elevato e una bassa varianza. Questo ci suggerisce che gli algoritmi di apprendimento non soddisfano i dati di addestramento (underfitting dei dati). Una soluzione a questo punto è passare a un algoritmo di apprendimento più complesso. Addestrare

l'algoritmo di apprendimento corrente su più features dovrebbe ridurre il bias aumentando la complessità del modello.

D'altra parte, con gli iperparametri ricavati dalla Grid Search, l'accuratezza nella classificazione con i dati di training e di test per ciascun algoritmo sono elencate in tabella (5.5).

Accuracy_score		
Algoritmo	**Accuratezza Training Set**	**Accuratezza Test Set**
MLP	0.721	0.708
SVM	0.748	0.75
k-NN	0.8	0.742

Tabella 5.5: Accuracy_score MLP, SVM, k-NN per multiclass.

Per visualizzare le performance di ogni algoritmo, sono state ottenute le seguenti Confusion Matrix, figure (5.6), (5.7), (5.8).

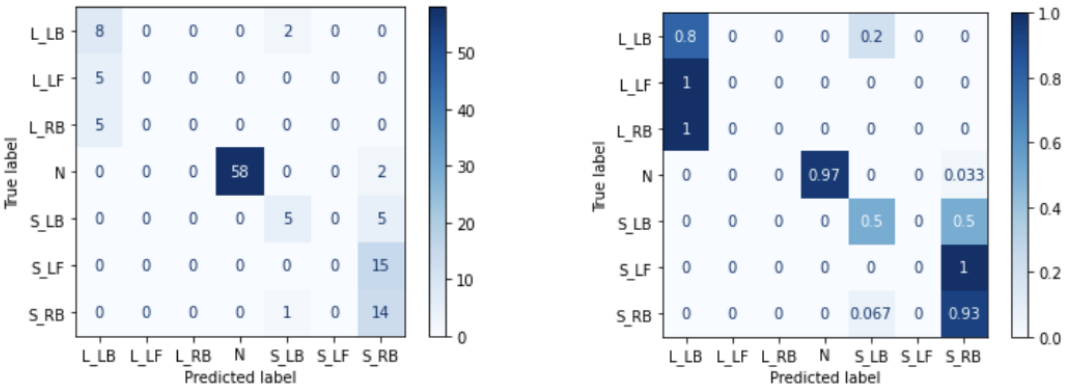


Figura 5.6: Confusion matrix multiclass per MLP.

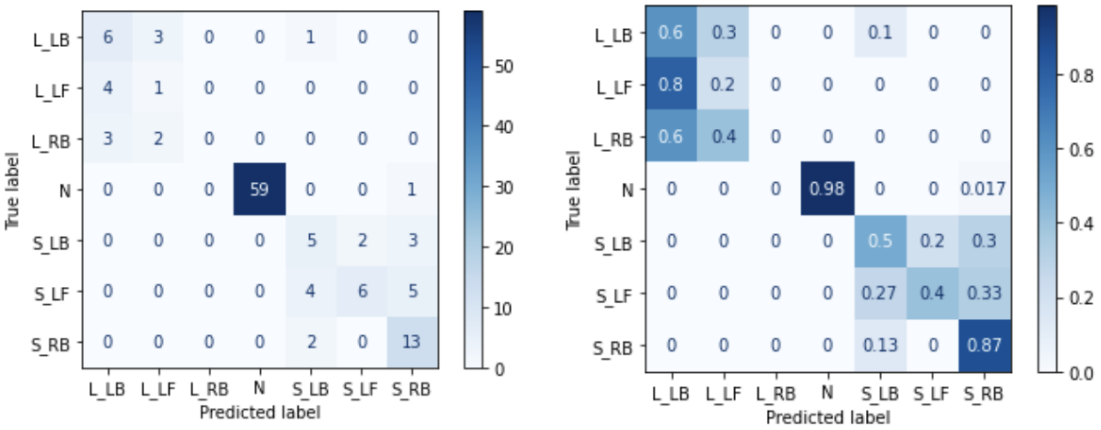


Figura 5.7: Confusion matrix multiclass per SVM.

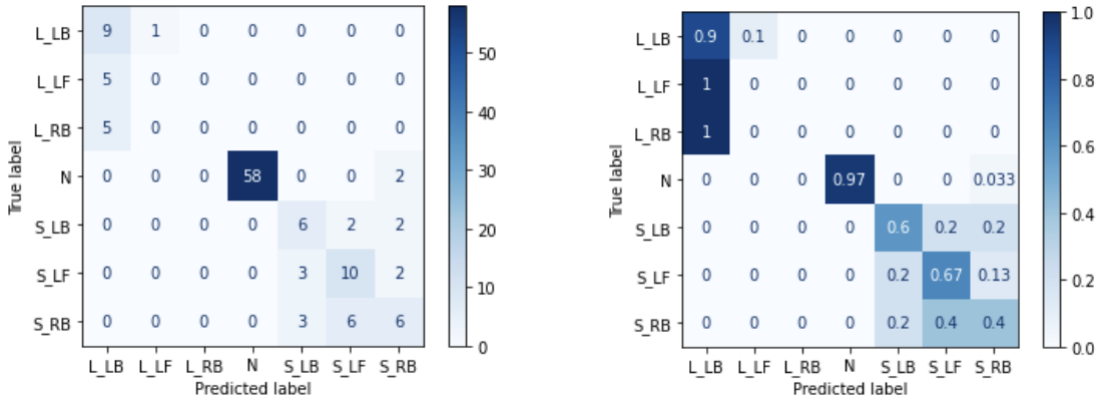


Figura 5.8: Confusion matrix multiclass per k-NN.

Dalla figura (5.6) si può notare che per l'algoritmo MLP, il classificatore scambia la classi L_LF e L_RB con L_LB, e la classe S_LF con S_RB.
Per questo motivo, il parametro Recall medio calcolato è pari al 52.8 %, mentre il parametro Precision medio è pari a 35.1 %.

Per quanto riguarda la Confusion Matrix dell'algoritmo SVM, dalla figura (5.7) si può notare che il classificatore scambia la classe L_RB con le classi L_LB e L_LF. In questo caso, il parametro Recall medio calcolato è pari al 50.7 %, mentre il parametro Precision medio è pari a 48.4 %.

Infine, per quanto riguarda la Confusion Matrix dell'algoritmo k-NN, dalla figura (5.8) si può notare che il classificatore scambia le classi L_LF e L_RB con la classe

L_LB.
In questo caso, il parametro Recall medio calcolato è pari al 50.5 %, mentre il parametro Precision medio è pari a 43.2 %.

Per quanto riguarda le curve ROC e Precision-Recall, sono state ottenute le seguenti curve, figure (5.9), (5.11), (5.13).

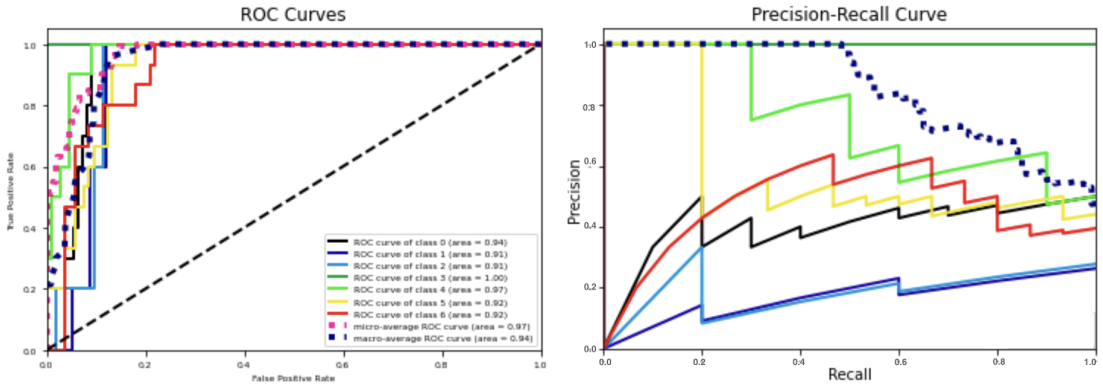


Figura 5.9: Curve ROC e Precision-Recall multiclass per MLP.

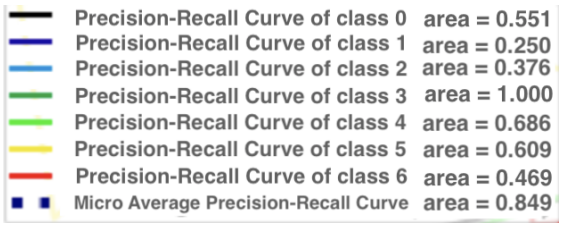


Figura 5.10: Legenda Curve Precision-Recall per MLP.

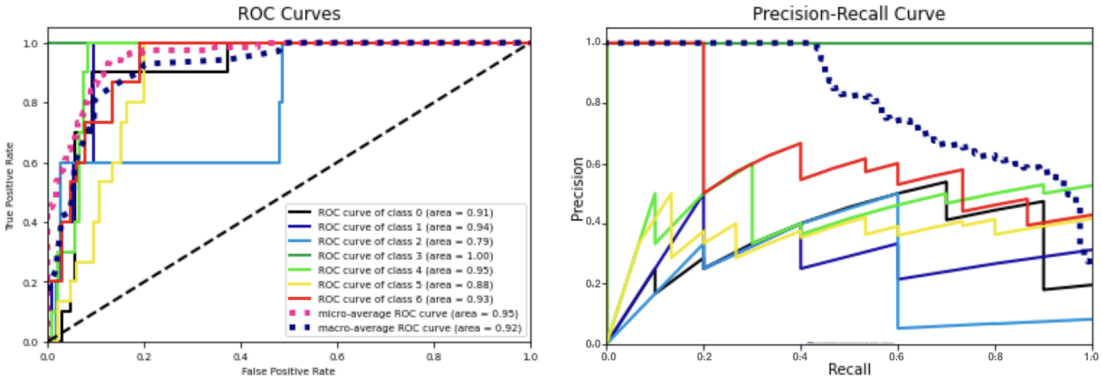


Figura 5.11: Curve ROC e Precision-Recall multiclass per SVM.

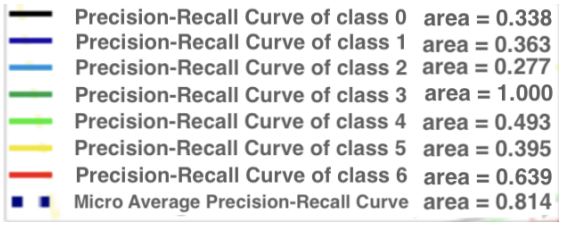


Figura 5.12: Legenda Curve Precision-Recall per SVM.

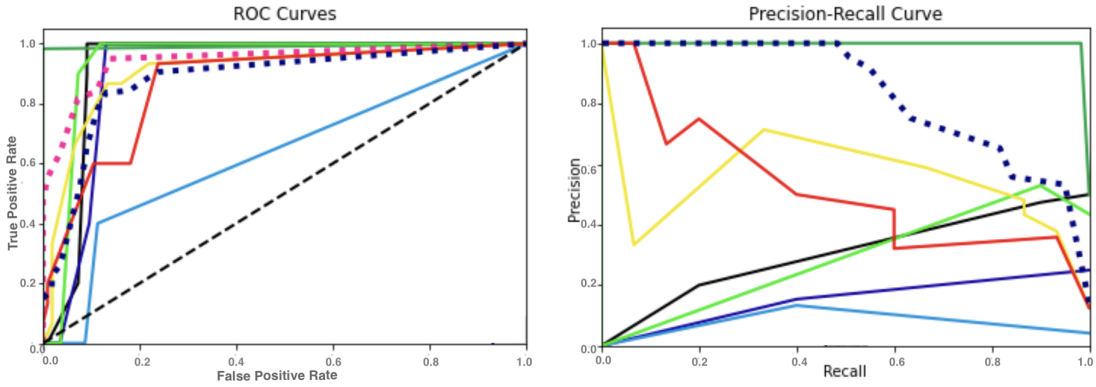


Figura 5.13: Curve ROC e Precision-Recall multiclass per k-NN.

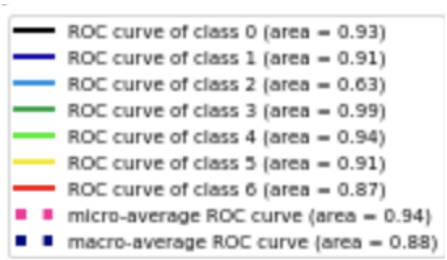


Figura 5.14: Legenda Curve-ROC per k-NN

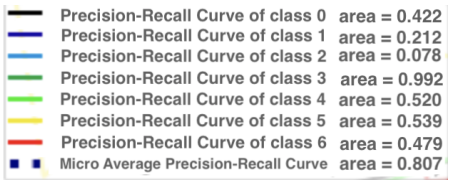


Figura 5.15: Legenda Curve-Precision-Recall per k-NN

Dai grafici sulle Curve ROC si può osservare che ad eccezione della classe 2 (che sarebbe la classe L_RB), i classificatori hanno una buona capacità di separabilità delle classi, ossia distinguono correttamente le classi. Difatti, per quanto riguarda la curva ROC del k-NN (figura 5.13), l'AUC è di 0.63 per la classe 2, il che significa che c'è il 63% di possibilità che il modello sia in grado di distinguere la corretta

classe dalle altre classi. Tutto sommato però, la Micro Average ROC ha un valore di AUC è di 0.94.

D'altra parte, osservando i grafici relativi alle curve Precision-Recall, si può dedurre che la relazione tra questi due parametri non è affatto buona per ogni classe analizzata, eccetto per la classe 3 (che corrisponde alla classe N). Nel complesso però, la Micro Average è discreta, raggiungendo il valore più alto di AUC corrispondente a 0.849, con l'algoritmo MLP.

In conclusione, gli algoritmi MLP e SVM hanno performance migliori in confronto al k-NN: il SVM ha un accuracy_score maggiore rispetto al MLP (75 % contro il 70.8 %); tuttavia il MLP ha la Micro Average migliore rispetto al SVM, con un AUC dello 0.97 contro lo 0.95 del SVM.

Al fine di aumentare le performance di questi algoritmi, si è costruito un dataset più ampio, costituito da 12000 features.

5.2 Risultati dataset finale

Il dataset con 12000 features è formato rispettivamente da 6000 campioni corrispondenti al target positivo (presenza dell'ictus) e 6000 campioni corrispondenti al target negativo (assenza dell'ictus).

Sulla base dei ragionamenti fatti in precedenza in questo capitolo con il dataset ridotto, anche in questo caso, per prima cosa gli algoritmi sono stati testati suddividendo i dati in sole due classi (positiva e negativa), successivamente è stato affrontato il problema multiclass.
Questa procedura pertanto è stata utilizzata per verificare che gli algoritmi funzionino correttamente nella classificazione con i dati realizzati.

Dunque, si inizia con lo stabile è necessario il preprocessing oppure no; di conseguenza sono stati calcolati i seguenti parametri, tabella (5.6).

Massimo dei Massimi	500.101
Minimo dei Massimi	10.895
Massimo dei Minimi	-12.619
Minimo dei Minimi	-1050.21

Tabella 5.6: Valori massimi e minimi della distribuzione dei dati.

Analizzando i valori nella tabella (5.6), si può dedurre che la distribuzione dei dati ha una media non nulla con una notevole varianza, dunque, è stato necessario procedere

con il preprocessing con feature standardization. I nuovi dati standardizzati, avranno media nulla e varianza unitaria.
Successivamente, è stata effettuata la feature reduction con PCA per estrarre le componenti principali; da un numero di componenti pari a 1152 (poiché ogni valore dei 576 valori è un numero complesso formato da parte reale e da parte immaginaria, dunque suddividendo tali valori prendendo ogni valore a sé si avranno 576x2 valori), dopo la PCA il numero di componenti principali è stato pari a 30.

5.2.1 Risultati dataset finale con problema binario

Dopo aver suddiviso i dati in due classi e di conseguenza in due cartelle, i dati sono stati importati con le funzioni delle libreria di sklearn (*sklearn. datasets._loadfiles*). Successivamente, è stato effettuato il preprocessing.

Una volta ultimato il preprocessing, il passo successivo è stato quello della ricerca degli iperparametri ideali per ogni algoritmo. In particolare sono stati scelti i seguenti parametri per ogni algoritmo, elencati in tabella (5.7).

Hypermarameters MLP	
Activation	'relu'
Alpha	0.0001
Hidden_layer_sizes	[1000,500]
Learning_rate	'adaptive'
Solver	'adam'
Hypermarameters SVM	
Kernel	'linear'
C	1
Hypermarameters k-NN	
n_neighbors	2

Tabella 5.7: Iperparametri MLP, SVM, k-NN.

Con questi iperparametri è stata ottenuta l'accuratezza nella classificazione con i dati training e con quelli di test, elencate in tabella (5.8).

Accuracy_score		
Algoritmo	**Accuratezza Training Set**	**Accuratezza Test Set**
MLP	1	1
SVM	1	1
k-NN	1	1

Tabella 5.8: Accuracy_score MLP, SVM, k-NN per classe binaria.

Per visualizzare le performance di ogni algoritmo, è stata realizzata la Confusion Matrix; per tutt'e tre è stata ottenuta la stessa Confusion Matrix, figura (5.16).

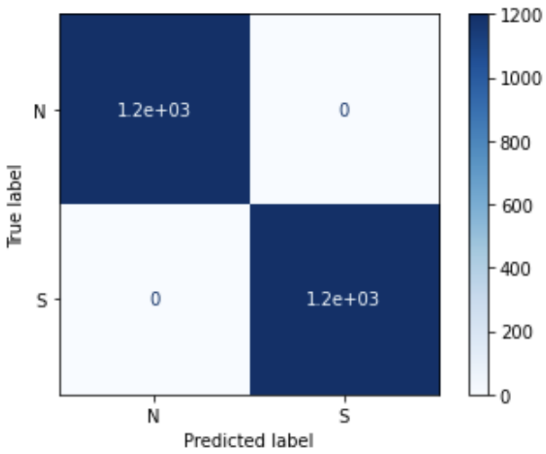


Figura 5.16: Confusion matrix binaria per MLP, SVM, k-NN.

Dalla figura (5.16), si può notare che viene riempita solamente la diagonale della matrice (dunque solamente VP e TN): la classificazione è ottima.
Di conseguenza, anche Precision e Recall per tutti e tre gli algoritmi sono al 100%.

Per quanto riguarda le curve ROC e Precision-Recall, per tutti e tre gli algoritmi, sono stati ottenuti gli stessi andamenti, figura (5.17).

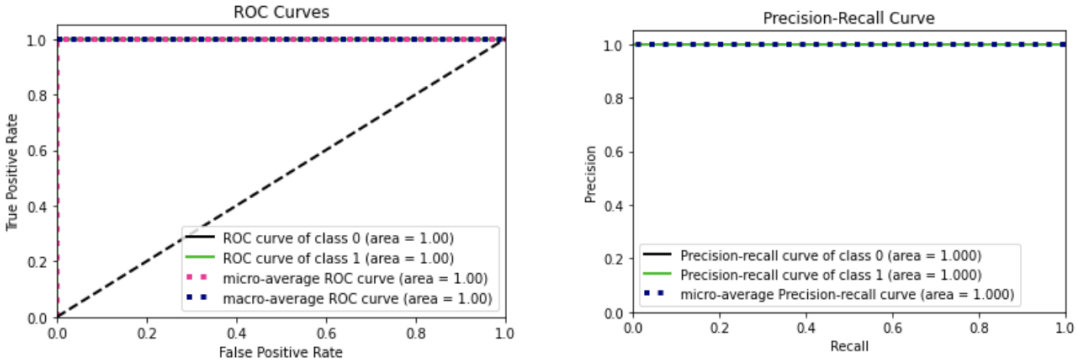


Figura 5.17: Curve ROC e Precision-Recall per classi binaria.

Come si può notare dal grafico (5.17), le curve ROC e Precision-Recall per ogni classe (insieme all'andamento Micro e Macro Average) hanno il comportamento ideale di un classificatore con Precision e Recall pari ad 1.
Di fatto, è stato ampiamente dimostrato che tutti e tre gli algoritmi funzionano abbastanza bene con un dataset con classi binarie. Il prossimo paragrafo illustrerà i risultati ottenuti con le classificazione multiclass.

5.2.2 Risultati dataset finale con problema multiclass

Innanzitutto, i dati vengono suddivisi in nove cartelle che corrispondono alle nove classi realizzate attraverso il caricamento dei dati con le funzioni delle libreria di Sklearn (*sklearn.datasets._loadfiles*), successivamente è stato effettuato il preprocessing dei dati.

Una volta ultimato il preprocessing, si è passati alla ricerca degli iperparametri ideali per ogni algoritmo mediante la Grid Search. In particolare, per ciascuno degli algoritmi, sono stati ottenuti i seguenti parametri elencati in tabella (5.9).

Hypermarameters MLP	
Activation	'relu'
Alpha	0.005
Hidden_layer_sizes	[1000,500,250]
Learning_rate	'constant'
Solver	'adam'
Hypermarameters SVM	
Kernel	'linear'
C	300
Hypermarameters k-NN	
n_neighbors	3

Tabella 5.9: Iperparametri MLP, SVM, k-NN per multiclass.

Con gli iperparametri ricavati dalla Grid Search, l'accuratezza nella classificazione con i dati di training e di test per ciascun algoritmo sono elencate in tabella (5.10).

Accuracy_score		
Algoritmo	**Accuratezza Training Set**	**Accuratezza Test Set**
MLP	0.998	0.997
SVM	1	0.999
k-NN	1	1

Tabella 5.10: Accuracy_score MLP, SVM, k-NN per multiclass.

Per visualizzare le performance di ogni algoritmo, sono state ottenute le seguenti Confusion Matrix, figure (5.18), (5.19), (5.20).

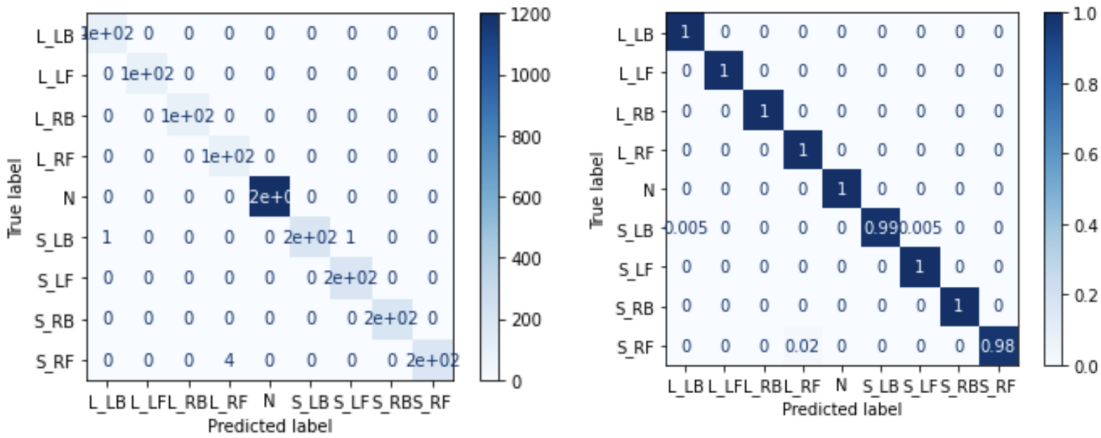


Figura 5.18: Confusion matrix multiclass per MLP.

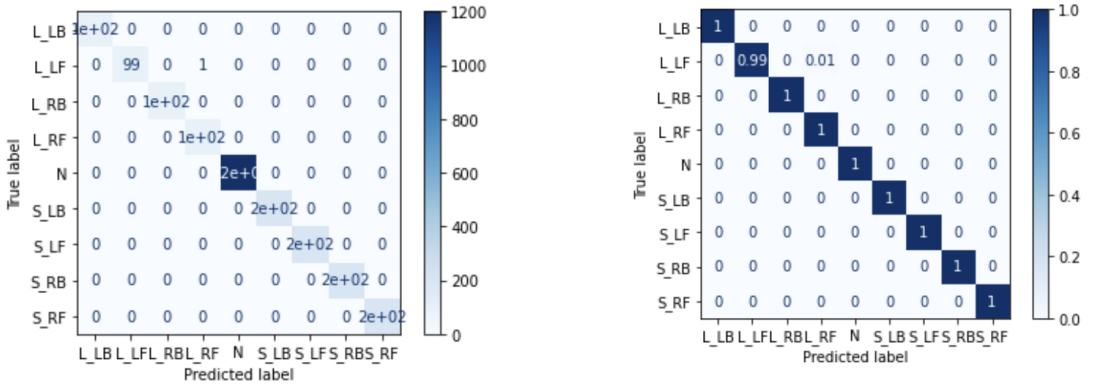


Figura 5.19: Confusion matrix multiclass per SVM.

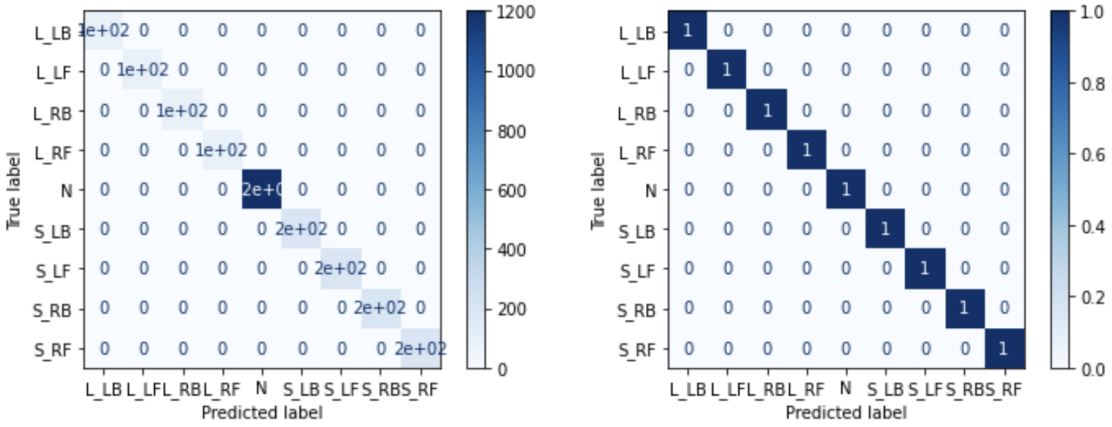


Figura 5.20: Confusion matrix multiclass per k-NN.

Dalla precedenti figure, si può osservare che per tutti e tre gli algoritmi avviene un ottima classificazione, difatti i valori sono esclusivamente sulla diagonale della matrice.

Dunque i valori di Precision e Recall sono pari al 100 %.

Per quanto riguarda le curve ROC e Precision-Recall, sono state ottenuti gli stessi risultati per tutti gli algoritmi, figura (5.21).

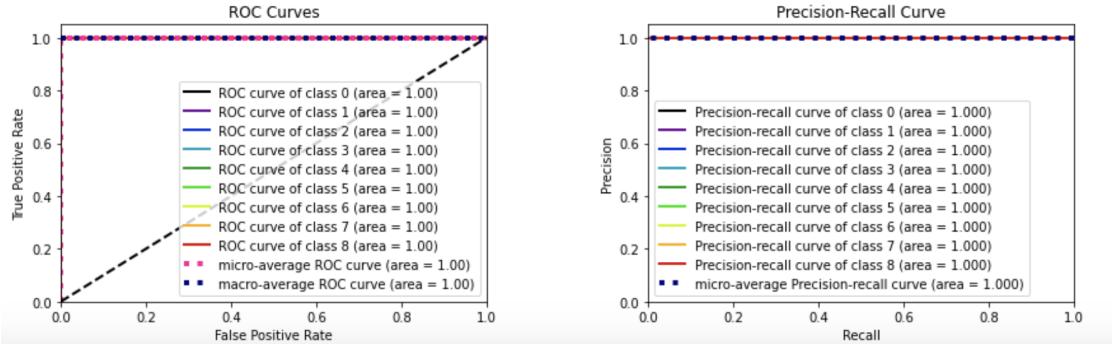


Figura 5.21: Curve ROC e Precision-Recall multiclass per MLP, SVM e k-NN.

Si può affermare che la relazione tra questi due parametri è ottima per ogni classe analizzata.

In conclusione, tutti e tre gli algoritmi hanno delle ottime performance, che ci permettono di ottenere dei classificatori quasi "perfetti", il che rende utile ed opportuno l'uso dell'intelligenza artificiale per futuri scopi.

Capitolo 6

Conclusioni e Sviluppi Futuri

In questa tesi è stato presentato un caso di studio sulla valutazione sperimentale iniziale di un sistema a microonde che si basa sulla rilevazione e classificazione dell'ictus cerebrale con gli algoritmi di ML. Il sistema a microonde proposto è un potenziale candidato per superare le limitazioni imposte dalle tecniche tradizionali attualmente in uso. Grazie all'utilizzo di tale gamma di frequenze, le radiazioni elettromagnetiche sono non ionizzanti, pertanto sono particolarmente adatte per effettuare un monitoraggio continuo, poiché non sono dannose per il paziente. I segnali elettromagnetici una volta che interagiscono con i tessuti cerebrali avranno delle differenze di proprietà dielettriche che permetteranno la rilevazione di anomalie.

La rilevazione e la discriminazione dell'ictus cerebrale nelle sue differenti dimensioni e localizzazioni sono state affrontate creando tantissime scene attraverso la simulazione di dati di scttering elettromagnetico, che grazie alla raccolta di questi dati hanno permesso l'utilizzo degli algoritmi di ML. Questo approccio è stato validato attraverso dei procedimenti inversi, sfruttandone l'algoritmo TSVD. Gli algoritmi di ML dunque, sono stati in grado di distinguere tra target che imitano l'ictus in base alla localizzazione e alla dimensione dello stesso.

Gli algoritmi hanno fornito un'ottima precisione di classificazione anche durante l'elaborazione di set di dati di addestramento con piccole dimensioni.

L'utilizzo della strategia MWI-ML sembra essere praticabile come strumento alternativo a quelli già esistenti e utile per la diagnosi clinica dell'ictus cerebrale. Il successo dell'impiego dell'apprendimento automatico sui dati clinici ottenuti potrebbe aiutare il medico nel monitoraggio post ictus cerebrale, e nella loro diagnosi.

Da una prospettiva clinica, è chiaro che una migliore accuratezza diagnostica preospedaliera potrebbe portare a una riduzione dei tempi di trattamento, aumentando così la sopravvivenza e la mitigazione delle lesioni.

In futuro si cercherà di ricostruire i dati in immagini, ed è per questo motivo che entra in gioco l'apprendimento automatico. In confronto ai programmi molto costosi dal punto di vista computazionale che la MWI richiede per ricostruire un'immagine delle proprietà dielettriche di un dato volume, il ML è più veloce ed economico dal punto di vista computazionale. Futuri studi si concentreranno sulla validazione dell'approccio MWI-ML proposto verso lo sviluppo di uno scanner portatile progettato per il rilevamento dell'ictus, magari generando immagini 3D che ne rispecchieranno forma e dimensione, ma soprattutto tipologia dell'ictus ischemico o emorragico. Il sistema così sarà in grado di ricostruire il bersaglio all'interno di un phantom cerebrale.

Bibliografia

[1] Mikael Persson, Andreas Fhager, Hana Dobšíček Trefná, Yinan Yu, Tomas McKelvey, Göran Pegenius, Jan-Erik Karlsson e Mikael Elam. «Microwave-based stroke diagnosis making global pre-hospital thrombolytic treatment possible». In: (nov. 2015), p. 1 (cit. alle pp. i, 6).

[2] R. Scapaticci, L. Di Donato, I. Catapano e L. Crocco. «A Feasibility Study on microwave imaging for Brain Stroke Monitoring». In: *Progress In Electromagnetics Research B* 40 (2012), pp. 1-2-3-4-5-6-7–8 (cit. alle pp. i, 6, 16, 18).

[3] Imran Sarwar, Giovanna Turvani, Mario R. Casu, Jorge A. Tobon, Francesca Vipiana, Rosa Scapaticci e Lorenzo Crocco. «Low-Cost Low-Power Acceleration of a Microwave Imaging Algorithm for Brain Stroke Monitoring». In: (nov. 2018), pp. 1–2 (cit. alle pp. i, 20, 22, 24, 25).

[4] Rosa Scapaticci, Jorge Tobon, Francesca Vipiana Gennaro Bellizzi, Senior Member e Lorenzo Crocco. «Design and Numerical Characterization of a Low-Complexity Microwave Device for Brain Stroke Monitoring». In: (dic. 2018), p. 7 (cit. alle pp. i, 20, 24, 25).

[5] Jorge A. Tobon Vasquez et al. «A Prototype Microwave System for 3D Brain Stroke Imaging». In: (mag. 2020), pp. 2-3-4–5 (cit. alle pp. ii, 7, 21–26).

[6] Salim S. Virani, Alvaro Alonso, Emelia J. Benjamin, Marcio S. Bittencourt, Clifton W. Callaway, April P.Carson, M.S. Heart Disease e Stroke Statistics. «Heart Disease and Stroke Statistics—2020 Update: A Report From the American Heart Association». In: *AHA Journals - Circulation* 141.9 (2020), e139–e596 (cit. a p. 1).

[7] Ministero della salute e Istituto superiore di sanità. *Gestione Sanitaria del Paziente con Emorragia SubAracnoidea (ESA) per rottura di Aneurisma Intracranico*. Relazione. 2009, p. 10 (cit. a p. 1).

[8] *Ictus Cerebrale*. URL: `https://www.my-personaltrainer.it/salute/ictus.html` (cit. alle pp. 1, 2).

[9] *Spettro elettromagnetico*. URL: http://glossario.oa-cagliari.inaf.it/spettro2.html (cit. a p. 2).

[10] eresa Popolizio e Tommaso Scarabino. «Ictus ischemico». In: () (cit. alle pp. 4, 5).

[11] Rosa Scapaticci, Mina Bjelogrlic, Jorge A. Tobon Vasquez, Francesca Vipiana, Michael Mattes e Lorenzo Crocco. «Microwave Technology for Brain Imaging and Monitoring: Physical Foundations, Potential and Limitations». In: *Nature* (mar. 2018), p. 8 (cit. alle pp. 5, 6, 14–16, 18).

[12] Andreas Fhager, Stefan Candefjord, Mikael Elam e Mikael Persson. «Microwave Diagnostics Ahead». In: *IEE Microwave Magazine* (mag. 2018), pp. 78-79-80–81 (cit. alle pp. 5, 6, 14, 15, 27).

[13] A. Von Hippel. *Dielectrics and Waves*. New York, 1999 (cit. a p. 8).

[14] C.Gabriel, S.Gabriel e E.Corthout. *The dielectric properties of biological tissues: I. Literature survey*. Phys. Med. Biol, 1996, pp. 2231–2249 (cit. a p. 8).

[15] R.W.Lau, S.Gabriel e E.Corthout. *The dielectric properties of biological tissues: II. Measurements in the frequency range 10 Hz to 20 GHz*. Phys. Med. Biol, 1996, pp. 2251–2269 (cit. a p. 9).

[16] R.W.Lau, S.Gabriel e E.Corthout. *The dielectric properties of biological tissues: III. Parametric models for the dielectric spectrum of tissues*. Phys. Med. Biol, 1996, pp. 2271–2293 (cit. a p. 10).

[17] Renato Orta. *Lecture Notes on Transmission Line Theory*. Dipartimento di Elettronica e Telecomunicazione, Politenico di Torino, set. 2017 (cit. a p. 11).

[18] Kai Chang. *Encyclopedia of RF and Microwave Engineering*. Wiley - Interscience, 2005, pp. 3693–3711 (cit. a p. 14).

[19] Catapano, L. Crocco, L. Di Donato, G. Angiulli, T. Isernia, A. F. Morabito, S. Tringali, O. M. Bucci e R. Massa. «Guidelines for effective microwave breast imaging: An accuratenumerical assessment against 3D anthropomorphic phantoms». In: (apr. 2009) (cit. a p. 15).

[20] *An Internet resource for the calculation of the Dielectric Properties of Body Tissues in the frequency range 10 Hz - 100 GHz*. URL: http://niremf.ifac.cnr.it/tissprop/ (cit. alle pp. 17, 76, 84).

[21] Jorge A. Tobon Vasquez et al. «Design and Experimental Assessment of a 2D Microwave ImagingSystem for Brain Stroke Monitoring». In: (mag. 2019), pp. 2-3-4-5–6 (cit. alle pp. 19, 20, 24, 25).

[22] Nadine Joachimowicz, Bernard Duchêne, Christophe Conessa e Olivier Meyer. «Anthropomorphic Breast and Head Phantoms for Microwave Imaging». In: (dic. 2018) (cit. a p. 19).

[23] Rowayda A. Sadek. «SVD Based Image Processing Applications: State of The Art, Contributions and Research Challenges». In: (2012) (cit. alle pp. 24, 25).

[24] Tyson Reimer, Jorge Sacristan e Stephen Pistorius. «Improving the Diagnostic Capability of Microwave Radar Imaging Systems using Machine Learning». In: () (cit. a p. 27).

[25] Raquel C. Conceiçao, Hugo Medeiros, Daniela M. Godinho, Martin O'Halloran, Diego Rodriguez Herrera, Daniel Flores Tapia e Stephen Pistorius. «Classification of breast tumor models with a prototype microwave imaging system». In: (mag. 2019) (cit. a p. 27).

[26] Marco Salucci, Angelo Gelmini, Jan Vrba, Ilja Merunka, Giacomo Oliveri e Paolo Rocca. «Instantaneous brain stroke classification and localization from real scattering data». In: (lug. 2018) (cit. a p. 27).

[27] A. Géron. «Hands-On Machine Learning with Scikit-Learn, Keras e TensorFlow. Concepts, Tools, and Techniques to Build Intelligent Systems». In: (set. 2019), pp. 3–32, 72–79, 90–102 (cit. alle pp. 28, 30–33, 35–39, 41, 46).

[28] Bernardita Stitic. «A multiclass neural network model for contaminant detection in hazelnut-cocoa spread jars». Apr. 2020 (cit. alle pp. 28–39, 41, 42, 45, 46, 51, 55, 56, 58).

[29] URL: https://www.kdnuggets.com/2018/10/mitchell-paradigm-concise-explanation-learning-algorithms.html (cit. a p. 28).

[30] *sklearn.modelselection.GridSearchCV*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (cit. a p. 31).

[31] URL: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76 (cit. a p. 36).

[32] *A. Ng. Machine Learning*. URL: https://www.coursera.org/learn/machine-learning (cit. alle pp. 37, 38).

[33] *sklearn.preprocessing.StandardScaler*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html (cit. a p. 39).

[34] *Grid Search for model tuning*. URL: https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e (cit. a p. 41).

[35] *Metrics to Evaluate your Machine Learning Algorithm*. URL: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234 (cit. a p. 42).

[36] *Confusion Matrix*. URL: https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html (cit. a p. 45).

[37] *Multi-class Classification: Extracting Performance Metrics From The Confusion Matrix*. URL: https://towardsdatascience.com/multi-class-classification-extracting-performance-metrics-from-the-confusion-matrix-b379b427a872 (cit. alle pp. 45–47).

[38] *Precision and recall*. URL: https://en.wikipedia.org/wiki/Precision_and_recall (cit. a p. 46).

[39] *Understanding AUC - ROC Curve*. URL: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (cit. alle pp. 48–50).

[40] *Perceptron*. URL: https://www.hackevolve.com/perceptron/ (cit. a p. 52).

[41] *How to Create a Multilayer Perceptron Neural Network in Python*. URL: https://www.allaboutcircuits.com/technical-articles/how-to-perform-classification-using-a-neural-network-introducing-the-perceptron/ (cit. alle pp. 52–54).

[42] Gianmarco Sabbatini. «Algoritmi di Machine Learning per classificare il comportamento di drosophhile in presenza di mutazioni genetiche». 2018 (cit. alle pp. 54, 57, 59, 64–66).

[43] Marina Routolo. «Analisi di Support Vector Machines per la classificazione automatica». Università degli Studi di Padova Facoltà di ingegneria Corso di Laurea in Ingegneria dell'Informazione, 2012 (cit. alle pp. 59–62).

[44] URL: https://www.researchgate.net/figure/2D-input-space-mapping-into-3D-feature-space-to-separate-data-linearly_fig2_282920922 (cit. a p. 59).

[45] *Multi-Class C-Support Vector Machines*. 2013. URL: http://www.dis.uniroma1.it/~or/gestionale/svm/multi-class.pdf (cit. a p. 63).

[46] *k-Nearest Neighbors*. URL: https://it.wikipedia.org/wiki/K-nearest_neighbors (cit. alle pp. 64, 65).

[47] URL: https://ii.uni.wroc.pl/~lipinski/DM2017/uczenie_nadzorowane.pdf (cit. a p. 66).

Ringraziamenti

A conclusione di questo percorso vorrei ringraziare tutte le persone che hanno permesso il raggiungimento di questo traguardo.

Ringrazio la professoressa Vipiana, che mi ha permesso di svolgere la tesi su degli argomenti che mi hanno sempre affascinato.

Ringrazio il professor Casu, per avermi aiutato tanto nella stesura della tesi, e soprattutto per avermi dato tanta fiducia.

Ringrazio il professor Tobon, per la sua costante presenza e disponibilità.

Ringrazio i miei colleghi del corso: Claudia, Massimo e Rosa, senza di loro seguire le lezioni sarebbe stato monotono.

Ringrazio i miei amici, nonché i miei figli: Agnese, Antonino, Camilla, Claudio, Erika, Giovanni, Pietro, Rachele e Vincenzo.

Ringrazio Annalisa, Carmelo, Federica, Federico, Fabio ed Ilenia, per tutti i pranzi e le cene che abbiamo fatto insieme come una vera famiglia.

Ringrazio i miei zii, i miei cugini, e i parenti tutti.

Ringrazio mia madrina Cesina e mio padrino Daniele.

Ringrazio Angela, Lorenzo e Gianluca, per essere stati i miei fratelli maggiori nei miei primi anni a Torino.

Quindi un ringraziamento particolare va alla città di Torino, per avermi cullato durante tutti questi anni e fatto crescere sotto tutti i punti di vista.

Infine ringrazio tutti i miei amici, quelli del mio paese, Elia e Leo, che mi hanno permesso di staccare la spina quando ho avuto bisogno.

Questo sia solo l'inizio di una carriera piena di successi e di soddisfazioni.

Grazie a tutti,
Riccardo