

# Tesina 3

## Analisi di Reti Neurali sul Dataset Wisconsin Breast Cancer

### Contesto del Problema

Un team di ricerca medica deve analizzare l'impatto dei diversi parametri delle Reti Neurali sulla loro performance di classificazione per la diagnosi del cancro al seno. Il dataset Wisconsin Breast Cancer contiene caratteristiche estratte da immagini di masse mammarie attraverso agoaspirato (FNA), offrendo un contesto reale e significativo per l'ottimizzazione delle reti neurali.

### Specifiche del Dataset

- Dataset Wisconsin Breast Cancer:
  - Features estratte da immagini di nuclei cellulari
  - Classificazione binaria (maligno/benigno)
  - 30 features (10 caratteristiche real-valued per ogni nucleo cellulare)
  - ~570 istanze
  - Dati numerici continui
  - Correlazioni complesse

### Obiettivi

1. Implementare una **Rete Neurale** base per classificazione usando scikit-learn
2. Studiare sistematicamente l'effetto dei parametri:
  - Convergenza del modello
  - Stabilità della classificazione
  - Tempi di training
  - Robustezza delle predizioni
3. Determinare configurazioni ottimali per:
  - Diverse architetture della rete
  - Funzioni di attivazione
  - Strategie di learning rate
  - Vincoli computazionali
  - Requisiti di accuratezza

### Vincoli

- Utilizzo stesso seed per confronti equi
- Minimo 30 run per configurazione
- Gestione appropriata della standardizzazione dei dati
- Cross-validation per ogni configurazione

### Fasi del Progetto

#### Fase 1: Implementazione Base

- Sviluppare rete neurale con:
  - Configurazione base MLPClassifier
  - Preprocessing dei dati
  - Train/test split appropriato
- Implementare logging dettagliato:
  - Accuracy per epoca
  - Tempi di training
  - Loss function

- Curve di apprendimento

## Fase 2: Analisi Parametrica

### 1. Scenario Architettura e Attivazione

- Test configurazioni layer:
  - Singolo layer: [(50,), (100,), (200,)]
  - Due layer: [(50,25), (100,50), (200,100)]
  - Tre layer: [(100,50,25), (200,100,50)]
- Funzioni di attivazione:
  - 'identity'
  - 'logistic'
  - 'tanh'
  - 'relu'
- Metriche:
  - Accuratezza
  - Convergenza training
  - Tempo di training

### 2. Scenario Learning Rate e Ottimizzazione

- Learning rates iniziali: [0.0001, 0.001, 0.01, 0.1]
- Learning rate policies:
  - 'constant'
  - 'invscaling'
  - 'adaptive'
- Solvers:
  - 'adam'
  - 'sgd'
  - 'lbfgs'
- Batch sizes: [32, 64, 128, 256]

### 3. Scenario Regolarizzazione

- Valori alpha: [0.0001, 0.001, 0.01, 0.1]
- Early stopping
- Validation split: [0.1, 0.2, 0.3]
- N\_iter\_no\_change: [5, 10, 20]

## Output Richiesti per Ogni Scenario

- Curve di apprendimento (training vs validation)
- Matrici di confusione
- Curve ROC
- Tempi di esecuzione
- Distribuzione degli score in cross-validation
- Box plot delle accurattezze
- Analisi della stabilità delle predizioni

## Codice Base di Riferimento

```
# Import base e configurazione default
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

```

import pandas as pd
import numpy as np

# Configurazione default
mlp_default = MLPClassifier(random_state=42)

# Esempio con parametri personalizzati completi
mlp_custom = MLPClassifier(
    hidden_layer_sizes=(100, 50),    # Due layer con 100 e 50 neuroni
    activation='tanh',               # Opzioni: 'identity', 'logistic', 'tanh', 'relu'
    solver='adam',                  # Opzioni: 'lbfgs', 'sgd', 'adam'
    alpha=0.0001,                   # Penalità L2
    batch_size=64,                  # Dimensione minibatch
    learning_rate='adaptive',        # Opzioni: 'constant', 'invscaling', 'adaptive'
    learning_rate_init=0.001,        # Learning rate iniziale
    max_iter=1000,                  # Numero massimo di iterazioni
    shuffle=True,                   # Shuffle dei campioni ad ogni iterazione
    random_state=42,                # Per riproducibilità
    early_stopping=True,             # Uso validation set per early stopping
    validation_fraction=0.2,        # Frazione di dati per validation
    n_iter_no_change=10,             # Numero iterazioni senza miglioramento
    verbose=True                     # Stampa messaggi di progresso
)

```

## Suggerimenti per l'Analisi

- Considerare l'impatto delle diverse funzioni di attivazione sulla velocità di convergenza
- Analizzare come le diverse strategie di learning rate influenzano la stabilità dell'apprendimento
- Valutare il trade-off tra complessità dell'architettura e performance
- Esaminare l'interazione tra funzione di attivazione e learning rate