POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Algoritmi di intelligenza artificiale per macchine industriali di taglio laser

Politecnico di Torino

Relatore
Prof. CERQUITELLI Tania

Correlatore:

Laureando
LONGO Samuele

Luglio 2025

Indice

Tabella dei contenuti

Elenco delle figure

Abstract

Negli scenari industriali contemporanei, l'adozione dell'Intelligenza Artificiale (AI) rappresenta un punto di svolta, in quanto consente l'analisi e l'ottimizzazione in tempo reale dei processi produttivi, favorendo la transizione verso sistemi intelligenti e adattivi, propri dell'Industria 4.0. In questo contesto, l'AI si configura come uno dei pilastri dell'innovazione, abilitando nuovi paradigmi di automazione e ottimizzazione, ed è ormai ampiamente utilizzata in settori ad alta competitività, quali quello automobilistico, aerospaziale e manifatturiero.
Tra i processi industriali più complessi e delicati vi è il taglio laser, una tecnica largamente adottata per la lavorazione di materiali con diverse caratteristiche e spessori. Il taglio laser offre numerosi vantaggi in termini di precisione, velocità e flessibilità, ma è anche influenzato da molteplici variabili, quali la qualità del materiale, la configurazione dei parametri di taglio, l'usura degli strumenti e le condizioni ambientali. Tecniche avanzate di intelligenza artificiale permettono oggi di affrontare queste criticità, abilitando l'adattamento in tempo reale ai cambiamenti di processo, la riduzione degli scarti, il rilevamento automatico dei difetti e il miglioramento della qualità del prodotto finale.

In questo contesto si inserisce la presente tesi, che si propone di contribuire alla ricerca sull'integrazione tra AI e automazione industriale, con un focus sull'ottimizzazione del processo di taglio laser mediante algoritmi di machine learning. L'obiettivo è lo sviluppo di un sistema intelligente in grado di classificare i difetti di taglio e suggerire correzioni ottimali dei parametri operativi. In caso di taglio difettoso, l'algoritmo riceve in input la configurazione attuale dei parametri e propone una nuova configurazione volta a ripristinare un taglio di qualità.
Una delle principali sfide affrontate nel progetto ha riguardato la scarsità e lo sbilanciamento dei dati reali provenienti dalle macchine di taglio laser, che hanno limitato l'efficacia dell'addestramento dei modelli predittivi. Per superare tale ostacolo, è stata adottata una strategia di generazione sintetica dei dati, supportata da tecniche di selezione automatica e pulizia del rumore, con l'obiettivo di costruire un dataset più denso e rappresentativo. Questo approccio ha consentito di migliorare la capacità di generalizzazione dei modelli e di ottimizzare l'intero processo di apprendimento degli algoritmi.

Capitolo 1

Introduzione

1.1 Contesto generale

Negli ultimi decenni, l'industria manifatturiera ha subito una profonda trasformazione grazie all'introduzione di tecnologie digitali avanzate e sistemi automatizzati. Questo processo, comunemente noto come Industria 4.0, si basa sull'integrazione di Internet of Things (IoT), Big Data, robotica avanzata e, soprattutto, Intelligenza Artificiale (AI). La combinazione di questi elementi consente di sviluppare sistemi produttivi automatizzati e digitalizzati, in cui le macchine sono in grado di comunicare tra loro, di monitorare lo stato dei processi in tempo reale e di adattarsi dinamicamente alle variazioni operative.
L'adozione dell'AI nei processi industriali è ormai un fattore chiave per mantenere competitività in un mercato globale sempre più esigente. Attraverso tecniche di machine learning e deep learning, è possibile estrarre valore dai dati prodotti continuamente dai sensori e dai sistemi di controllo, migliorando la qualità, riducendo gli sprechi, prevenendo guasti e ottimizzando i cicli produttivi. Inoltre, queste tecnologie permettono una maggiore personalizzazione dei prodotti, flessibilità nelle linee di produzione e una gestione proattiva della manutenzione.
L'applicazione dell'AI va ben oltre il semplice monitoraggio: si estende alla capacità di prendere decisioni autonome o semiautonome, supportando gli operatori umani e riducendo l'errore umano. Questi sistemi intelligenti sono fondamentali soprattutto in processi complessi e delicati, dove la qualità finale del prodotto dipende da un gran numero di parametri interagenti, spesso difficili da controllare manualmente.

Nonostante il potenziale, l'integrazione efficace dell'AI nei contesti industriali richiede un approccio multidisciplinare, che coinvolge non solo i data scientist ma anche gli esperti di processo, e gli operatori di linea. L'interazione tra dominio applicativo e tecnologie avanzate è fondamentale per sviluppare soluzioni robuste, efficienti e di reale impatto operativo.

1.2 AI nell'industria dei macchinari di taglio laser

l taglio laser rappresenta una delle tecniche di lavorazione più avanzate e precise disponibili nell'industria moderna. La sua versatilità lo rende adatto a diversi materiali, spessori e applicazioni, garantendo tolleranze dimensionali molto strette e ottime finiture superficiali. Tuttavia, la qualità del taglio è fortemente influenzata da molteplici fattori, tra cui la qualità del materiale, le condizioni ambientali, l'usura degli strumenti e la corretta configurazione dei parametri di taglio (potenza, velocità, pressione gas, distanza focale, tipo di ugello). L'integrazione di algoritmi di AI permette oggi di ottimizzare questi parametri in modo automatico, riducendo sensibilmente i difetti e gli scarti.
Ad esempio, TRUMPF [15] ha sviluppato un sistema chiamato "Cutting Assistant", che utilizza l'AI per ottimizzare i parametri di taglio e migliorare la qualità dei bordi tagliati. Questo sistema è stato implementato con successo su macchine TruLaser con potenza superiore a 6 kW, contribuendo a migliorare l'efficienza e la qualità del processo di taglio.
Anche Mitsubishi ha integrato l'AI nelle sue macchine laser della serie GX-F ADVANCED, consentendo regolazioni automatiche per migliorare la stabilità del processo e ottimizzare l'uso dei materiali [16].
A supporto di questi sviluppi, l'uso di sistemi intelligenti per il taglio laser si dimostra efficace anche in termini di sostenibilità e risparmio di materiale. Secondo ADH Machine Tool [17], l'integrazione dell'AI con algoritmi di nesting avanzati e sistemi di controllo adattivo consente una riduzione degli sprechi di materiale tra il 15% e il 30%, con risultati particolarmente rilevanti nei processi su acciaio inossidabile e alluminio. Questo non solo ottimizza il rendimento della materia prima, ma contribuisce anche a ridurre i costi operativi e l'impatto ambientale del processo produttivo.

1.3 Scarsità dei Dati nei Contesti Industriali

Nonostante i numerosi benefici offerti dall'intelligenza artificiale, una delle principali criticità nella sua applicazione in ambito industriale riguarda la disponibilità e la qualità dei dati. In particolare, nei processi produttivi complessi come il taglio laser, la raccolta di dati etichettati, completi e rappresentativi risulta spesso limitata a causa di diversi fattori:

Costi e tempi di acquisizione: l'impiego prolungato delle macchine per l'esecuzione di test finalizzati alla raccolta dati può causare rallentamenti significativi della produzione e generare costi elevati, riducendo l'efficienza operativa complessiva.
Assenza di una cultura del dato: in molte realtà industriali, soprattutto quelle meno digitalizzate o a gestione tradizionale, manca una cultura consolidata di storicizzazione e monitoraggio sistematico dei dati di processo.
Distribuzione sbilanciata delle classi: i dati raccolti tendono spesso a essere fortemente sbilanciati, con una predominanza di esempi appartenenti a classi frequenti e una scarsa rappresentazione delle condizioni anomale o difettose, rendendo difficile l'addestramento di modelli predittivi affidabili.

Queste limitazioni pongono una sfida concreta allo sviluppo di soluzioni basate su AI, rendendo necessario l'impiego di tecniche di preprocessing e generazione sintetica per compensare la scarsità informativa. Tali approcci verranno approfonditi nel presente lavoro di tesi.

1.4 Obiettivi

L'obiettivo principale di questo lavoro è lo sviluppo di una strategia che sfruttando degli algoritmi di machine learning è in grado di supportare gli operatori nella correzione automatica dei parametri di taglio laser, migliorando così la qualità del processo produttivo e riducendo il numero di difetti.

A tal fine, viene adottato un approccio basato su algoritmi di machine learning, articolato in due fasi principali:

- La classificazione dei difetti di taglio a partire dalla configurazioni parametriche
- Proposta di configurazioni parametriche correttive tramite modelli di regressione, dove ogni tipologia di difetto viene trattata separatamente, così da permettere l'applicazione di strategie di ottimizzazione specifiche e mirate. Questa scelta consente di affrontare in modo più preciso la variabilità del processo e la natura eterogenea dei problemi riscontrati nei diversi scenari di taglio.

A monte dell'addestramento dei modelli, viene condotta un'attenta fase di analisi e preparazione dei dati. In particolare, vengono affrontate criticità legate alla scarsità e allo sbilanciamento dei dati attraverso l'impiego di tecniche di generazione sintetica, finalizzate a rafforzare la rappresentatività delle classi più rare e delle combinazioni meno frequenti di parametri (come spessore, tipo di ugello e difetto). Per garantire l'affidabilità del dataset e mitigare l'effetto del rumore introdotto dalla generazione artificiale, vengono inoltre applicati metodi di selezione, pulizia e rilevamento degli outlier.
La presente tesi comprende quindi tutte le fasi chiave di un processo di apprendimento automatico applicato in ambito industriale: dalla pre-elaborazione dei dati, alla definizione, addestramento e validazione dei modelli predittivi, fino alla valutazione delle prestazioni nel contesto reale di produzione.

1.5 Organizzazione del documento

Nel capitolo successivo (Cap. 2) affronteremo alcuni aspetti introduttivi propri dell'ottimizzazione di parametri di configurazione o dell'identificazione di anomalie in ambito industriale, assieme all'analisi dello stato dell'arte, al fine di comprendere al meglio il punto di partenza del lavoro. Seguiranno i capitoli dedicati alla descrizione dell'architettura utilizzata e le varie strategie adottate per mitigare le problematiche presenti nel caso preso in esame (Cap. 3) e dei risultati ottenuti (Cap. 4), che valuteranno l'efficacia dell'approccio seguito. Infine, verrà analizzato il lavoro svolto e verranno forniti alcuni spunti sugli sviluppi futuri a valle dei risultati ottenuti (Cap. 5).

Capitolo 2

Stato dell'arte

In questo capitolo si analizza lo stato dell'arte relativo all'ottimizzazione dei parametri di configurazione nelle macchine a taglio laser, con l'obiettivo di ottenere una qualità di taglio ottimale in termini di precisione, regolarità e assenza di difetti. In particolare, vengono approfondite le principali metodologie e tecniche impiegate in letteratura per affrontare il problema, con un focus sui modelli predittivi, le tecniche di ottimizzazione (sia deterministiche che metaeuristiche), e l'utilizzo di algoritmi di apprendimento automatico. L'ottimizzazione dei parametri di taglio più utilizzati, come potenza del laser, velocità di avanzamento, distanza focale e pressione del gas ausiliario, rappresenta un problema complesso a causa delle numerose variabili in gioco e della loro interdipendenza non lineare. L'approccio tradizionale, spesso basato su prove sperimentali ed euristiche ingegneristiche, risulta dispendioso in termini di tempo e poco generalizzabile. Per questo motivo, negli ultimi anni si è assistito a una crescente adozione di tecniche data-driven e di intelligenza artificiale, che permettono di modellare e ottimizzare il processo in maniera più efficiente e scalabile. Per avere un'idea più chiara del contesto preso in esame è opportuno descrivere brevemente i parametri di processo e le tipologie di laser più diffuse nell'ambito del taglio laser applicato a materiali metallici.

2.1 Tecnologie di taglio laser e parametri di processo

Il taglio laser è una tecnologia ampiamente adottata nell'industria manifatturiera grazie alla sua precisione, flessibilità e velocità. Le sorgenti laser più utilizzate per il taglio di metalli sono i laser CO_2 e Nd:YAG (Neodimio:Ittrio-Alluminio-Garnet), ciascuno con caratteristiche ottiche e applicative differenti.
I laser possono operare in modalità continua (Continuous Wave – CW) o pulsata (Giant Pulsed - GP), con implicazioni diverse sulla qualità del taglio. In particolare, la modalità GP emette impulsi di energia molto brevi, singoli o periodici. Questi impulsi presentano potenze di picco estremamente elevate consentendo una consegna dell'energia precisa e localizzata, minimizzando la diffusione del calore e i danni termici ai materiali circostanti, consentendo un maggiore controllo sul trasferimento termico, risultando utile per materiali sensibili o geometrie complesse.

I principali parametri di processo che influenzano la qualità del taglio sono:

- Potenza del laser: È la quantità di energia emessa dal laser nell'unità di tempo, più alta è la potenza, più calore può essere trasferito al materiale per fonderlo, vaporizzarlo o bruciarlo. espressa in Watt [W]
- Velocità di avanzamento: È la velocità con cui la testa del laser si muove lungo la traiettoria di taglio, espressa in mm/min o m/min.
- Frequenza (solo in modalità pulsata): Quantità di impulsi del laser emessi al secondo
- Duty cycle (solo in modalità pulsata): Indica la percentuale di tempo durante la quale il laser è acceso (cioè, attivo) all'interno di ciascun ciclo di impulso.
- Lunghezza focale: Distanza tra la lente ottica del sistema laser e il punto in cui il raggio raggiunge la massima concentrazione dell'energia
- Tipo e pressione del gas assistente (es. azoto, ossigeno): Insieme al laser viene veicolato anche un gas che fonde o vaporizza il materiale rendendo il taglio più pulito
- Stand-off distance: Distanza tra ugello (convoglia il gas assistente) e superficie del materiale

Questi parametri interagiscono in modo non lineare, influenzando la larghezza del taglio (distanza tra i bordi della fessura creata dal taglio), la rugosità superficiale, la zona termicamente alterata (HAZ) e la presenza di scorie o striature che rappresentano le caratteristiche che accomunano i vari difetti di taglio [2][3][6].

2.2 Approcci fisici e analitici nell'ottimizzazione del processo

L'ottimizzazione dei parametri di taglio laser è da tempo oggetto di studio nella letteratura tecnico-scientifica, con un ampio utilizzo di modelli fisici, analisi statistiche e metodi empirici. Questi approcci sono stati fondamentali per comprendere le dinamiche del processo e per costruire basi teoriche solide da cui partire nella definizione di strategie di controllo.

2.2.1 Analisi statistiche e sperimentali

Una prima classe di approcci si basa su **esperimenti sistematici**, progettati secondo criteri scientifici e supportati da strumenti matematici e statistici avanzati, con l'obiettivo di modellare in modo quantitativo le relazioni tra i parametri di processo e la qualità del taglio. Tra le tecniche principali:

- **ANOVA (Analysis of Variance**: Può essere utilizzata per valutare l'influenza statistica dei singoli parametri di taglio come potenza del laser, velocità di avanzamento e pressione del gas assistente sulle principali metriche di qualità del processo, tra cui la rugosità superficiale (Ra), la larghezza del taglio (kerf) e la quantità di scorie residue. Il test ANOVA consente di confrontare le medie dei risultati ottenuti da diverse configurazioni di parametri, determinando se le variazioni osservate nelle metriche sono attribuibili a cambiamenti reali nei fattori sperimentali oppure a fluttuazioni casuali. Perché il test ANOVA dia risultati affidabili, ci sono alcune condizioni (ipotesi statistiche) che devono essere rispettate:

  1) Normalità dei residui: Gli errori (differenza tra valore osservato e medio) devono seguire una distribuzione normale, per evitare che vengano fatte assunzioni errate dovute alla presenza di valori anomali o distribuzioni fortemente asimmetriche.
  2) Omoscedasticità **(**varianze omogenee**)**: Tutti i gruppi (es. diversi livelli di potenza, velocità, ecc.) devono avere varianze simili nella variabile risposta (Ra, kerf, ecc.). Se un gruppo ha una varianza molto diversa dagli altri, l'ANOVA può attribuire erroneamente questa variabilità interna a differenze tra gruppi, falsando l'analisi.

  3) Indipendenza delle osservazioni: Ogni osservazione deve essere statisticamente indipendente dalle altre (nessun effetto "a catena" o "a blocchi") per evitare che il modello "peschi" correlazioni false nei dati (ad esempio in dati temporali o spaziali).

Tale test può essere eseguito in varie modalità definite come $n - vie$, dove n indica il numero di fattori per cui si valuta l'effetto sulla variabile di risposta.
Infine, per valutare l'affidabilità dei risultati ottenuti dal test è opportuno valutare due parametri fondamentali:
F-value: Rappresenta il rapporto tra la varianza tra gruppi distinti e la varianza interna

ad ogni gruppo. Se $\mathbf{F \approx 1}$ allora le differenze tra i gruppi sono comparabili a quelle dentro i gruppi → nessuna evidenza di effetto significativo,
mentre Se $\mathbf{F \gg 1}$ → le differenze tra i gruppi sono maggiori di quelle all'interno allora è probabile che almeno un gruppo sia diverso dagli altri (effetto significativo).

p-value**:** rappresenta la probabilità di osservare, sotto l'ipotesi nulla (cioè assumendo che non vi sia alcun effetto reale), differenze nei dati almeno così estreme come quelle rilevate. In altre parole, indica quanto sia compatibile il risultato ottenuto con l'ipotesi che non ci sia alcuna relazione tra le variabili analizzate. Un p-value inferiore alla soglia di significatività (tipicamente 0.05) suggerisce che l'effetto osservato è statisticamente significativo, ovvero che è improbabile che le differenze siano dovute al caso, e quindi il parametro analizzato ha probabilmente un effetto reale sulla variabile di risposta.

- **Modelli di regressione**: Permettono di costruire funzioni matematiche predittive che descrivono la relazione tra i parametri di processo (input) e la qualità del taglio (output), come rugosità, kerf, o presenza di scorie. Tra le tecniche più utilizzate vi sono la regressione lineare, polinomiale e le varianti multiple. Questi modelli, oltre a fornire stime quantitative, permettono anche di valutare l'importanza relativa di ciascun parametro e di identificare eventuali interazioni.I modelli di regressione possono essere ulteriormente estesi tramite la Response Surface Methodology (RSM), una tecnica statistica che consente non solo di modellare fenomeni complessi con interazioni non lineari, ma anche di visualizzare la risposta del sistema su superfici tridimensionali e determinare le condizioni ottimali dei parametri tramite grafici 3D o contour plot che mostrano come cambia la risposta al variare di due fattori, mantenendo gli altri fissi, e il calcolo è effettuato mediante l'utilizzo delle funzioni di regressione. L'RSM si rivela particolarmente utile nei casi in cui si voglia esplorare lo spazio dei parametri in modo efficiente, riducendo il numero di esperimenti necessari grazie a un disegno sperimentale ben strutturato. Nel lavoro [1], ad esempio, si osserva come la larghezza del taglio aumenti all'aumentare della potenza e della pressione del gas, mentre diminuisca con valori più elevati di velocità e frequenza. Tali comportamenti sono rappresentati tramite superfici di risposta che evidenziano la natura monotona e semi-empirica del fenomeno, suggerendo che i modelli di regressione forniscono una buona approssimazione locale ma possono necessitare di ulteriori affinamenti per generalizzare a tutto lo spazio operativo. Tuttavia, è importante ricordare che i modelli di regressione si basano su alcune ipotesi statistiche (come linearità, indipendenza degli errori, omoscedasticità e normalità dei residui), che vanno verificate e validate prima di procedere all'ottimizzazione.

Inoltre, in presenza di fenomeni altamente non lineari o rumore sperimentale elevato, tecniche più avanzate di machine learning possono risultare più efficaci.

2.2.2 Approcci empirici tradizionali

I primi studi sull'ottimizzazione del taglio laser adottavano un approccio più empirico e osservazionale, basato sull'esecuzione di numerosi test pratici per comprendere l'effetto dei parametri sulla qualità del taglio. Sebbene utili per ottenere intuizioni iniziali, questi metodi sono costosi e poco scalabili, specialmente in presenza di pochi dati o spazi sperimentali ampi. Per ovviare a tali limiti, sono stati introdotti strumenti statistici semplificati che aiutano a ridurre il numero di esperimenti mantenendo l'efficacia dell'analisi. Tecniche come il metodo Taguchi e ancora l'ANOVA vengono qui impiegate non tanto per modellare il sistema in senso matematico, quanto per identificare rapidamente i parametri più influenti. Nel lavoro [4], ad esempio, l'ANOVA è stata utilizzata per analizzare l'impatto della potenza, della velocità e della distanza tra ugello e materiale (stand-off) nel taglio di fogli in PMMA. I risultati mostrano come la potenza sia il parametro dominante, con un'influenza che tende a ridursi all'aumentare del suo valore.

2.3 Modelli matematici per l'ottimizzazione del taglio laser

Una parte significativa della letteratura si concentra sulla costruzione di modelli matematici, che descrivono il legame tra parametri di input e output del processo. Spesso vengono utilizzati metodi scientifici basati sulla definizione di equazioni che legano le variabili d'interesse come svolto da Chmelickova e Polak [5], che hanno derivato un'approssimazione per la velocità di taglio ottimale dell'acciaio inossidabile utilizzando un modello basato sull'equazione del bilancio termico ed energetico:

$$\boldsymbol{\rho\, V(c\ \Delta T\ +\ L_v + Q_1) =\ P(1\ -\ R)}$$

dove l'energia fornita dal laser è bilanciata dall'energia necessaria a vaporizzare il materiale lungo il percorso di taglio.
Altri metodi riguardano i modelli di regressione che abbiamo già visto in precedenza nel paragrafo 2.1.1, una tecnica matematica e statistica utilizzata per descrivere la

relazione tra una o più variabili indipendenti (input) e una variabile dipendente (output), al fine di predire il valore dell'output sulla base degli input forniti) utili per rappresentare la variazione di parametri target in funzione di altri. In seguito, vengono mostrati alcuni esempi di funzioni di regressione.

2.3.1 Regressione lineare

La regressione lineare si basa sull'assunzione che esista una relazione di tipo affine tra la variabile di input e la variabile di output. La formula generale della regressione lineare semplice è:

$$y = ax + b$$

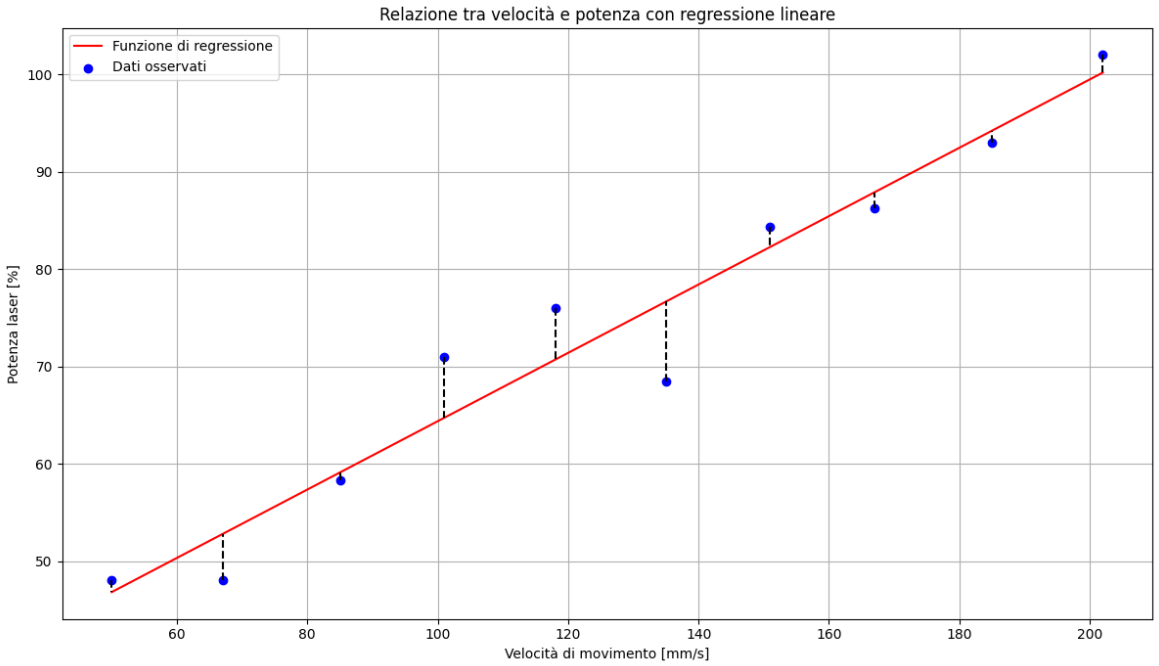


Figura 2.0: Regressione lineare

Nel contesto del taglio laser, questa metodologia viene spesso utilizzata per descrivere e prevedere il comportamento di parametri di processo sulla base di altri fattori noti come ad esempio:

$$Potenza = a \cdot Velocità + b$$

dove:
Potenza: rappresenta il parametro di output che si vuole prevedere o ottimizzare,
Velocità (velocità di avanzamento): è la variabile indipendente (input)
a: è il coefficiente angolare (pendenza della retta), che indica quanto varia la potenza al variare della velocità,
b: è il termine noto, ovvero il valore della potenza quando la velocità è zero.

2.3.2 Regressione polinomiale

La regressione polinomiale rappresenta un'estensione della regressione lineare, consentendo di modellare relazioni più complesse tra variabili tramite l'utilizzo di polinomi di grado superiore al primo. Questa tecnica è particolarmente utile quando la relazione tra la variabile indipendente e quella dipendente non è adeguatamente descritta da una retta, ma mostra andamenti curvilinei.
La regressione polinomiale di secondo grado, la più comune, assume la seguente forma:

$$y = ax^2 + bx + c$$

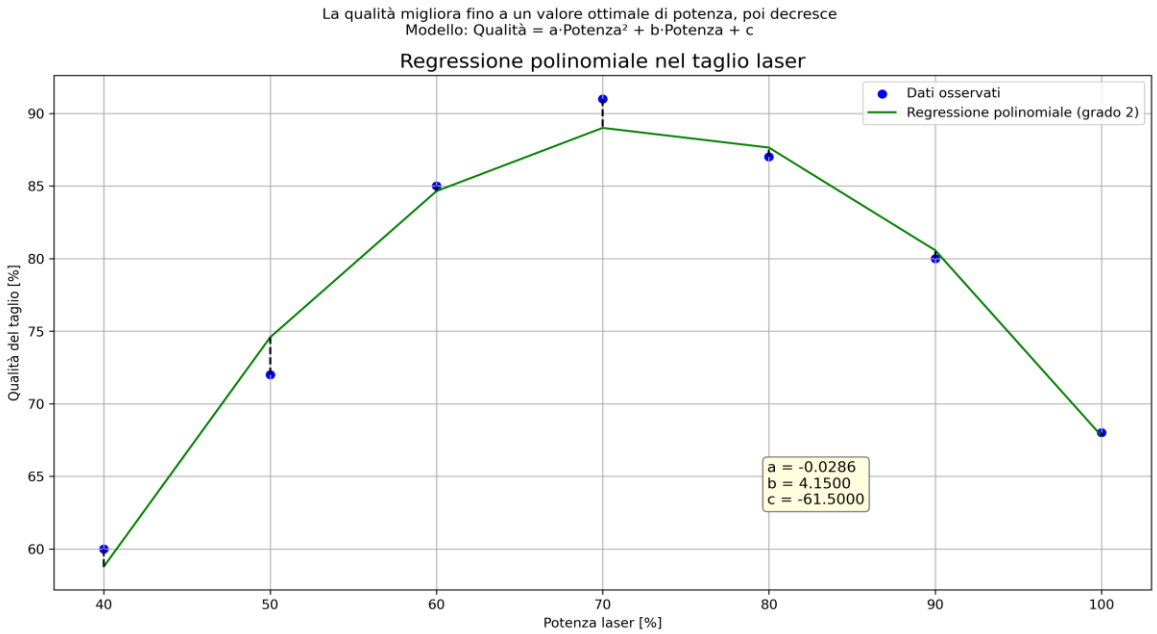


Figura 2.1: Regressione polinomiale

Nel taglio laser, la regressione polinomiale può essere utilizzata per descrivere fenomeni in cui, ad esempio, la qualità del taglio migliora fino a un certo valore ottimale di potenza, per poi peggiorare se questa viene ulteriormente aumentata. In questi casi la relazione tra input e output non è lineare, ma presenta un punto di ottimo che può essere individuato grazie al modello polinomiale:

$$Qualità = a \cdot Potenza^2 + b \cdot Potenza + c$$

$\underline{Potenza}$: È la variabile indipendente (input)
$\underline{Qualità}$: La variabile dipendente di cui si vuole stimare il valore ottimo (output)
$\underline{a}$, $\underline{b}$ e $\underline{c}$: Sono i parametri del modello, dove Se $\boldsymbol{a < 0}$ la parabola è rivolta verso il basso e ha un massimo (punto di ottimo), se $\boldsymbol{a > 0}$ la parabola è rivolta verso il basso e ha un minimo. Nell'esempio descritto la qualità che migliora fino a un valore ottimale di potenza e poi peggiorare se si va oltre.

2.3.3 Regressione esponenziale

La regressione esponenziale è una tecnica utilizzata per modellare relazioni in cui la variazione della variabile dipendente segue un andamento di crescita o decrescita esponenziale rispetto alla variabile indipendente. Questo tipo di modello è particolarmente adatto quando i dati mostrano un incremento (o una diminuzione) molto rapido che non può essere adeguatamente descritto da un modello lineare o polinomiale. La formula generale della regressione esponenziale è:

$$y = a \cdot e^{bx}$$

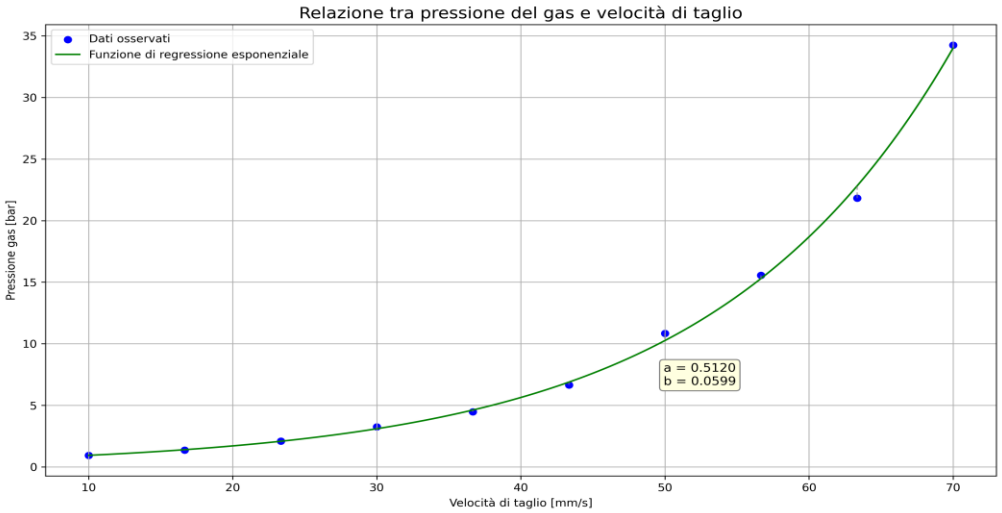


Figura 2.2: Regressione esponenziale

Nel contesto del taglio laser, la regressione esponenziale può essere utilizzata, ad esempio, per descrivere la relazione tra la pressione del gas e la velocità di taglio, in cui la pressione aumenta rapidamente all'aumentare della velocità fino a stabilizzarsi:

$$Pressione = a \cdot e^{b \cdot Velocità}$$

$\underline{Pressione}$: Rappresenta il parametro di output che si vuole prevedere o ottimizzare,
$\underline{Velocità}$: (velocità di avanzamento): È la variabile indipendente (input)
$\underline{a}$: Coefficiente di scala
$\underline{b}$: È il coefficiente di crescita (se positivo) o di decadimento (se negativo)
$\underline{e}$: È la base dei logaritmi naturali (circa 2,718)

2.3.4 Regressione logaritmica

La regressione logaritmica è una tecnica utilizzata per modellare relazioni in cui la variabile dipendente varia in modo rapido inizialmente, per poi rallentare e tendere ad appiattirsi al crescere della variabile indipendente. Questo modello è particolarmente adatto quando l'incremento della variabile di output si riduce progressivamente all'aumentare dell'input, fenomeno tipico in molti processi fisici e industriali.

$$y = a \cdot ln(x) + b$$

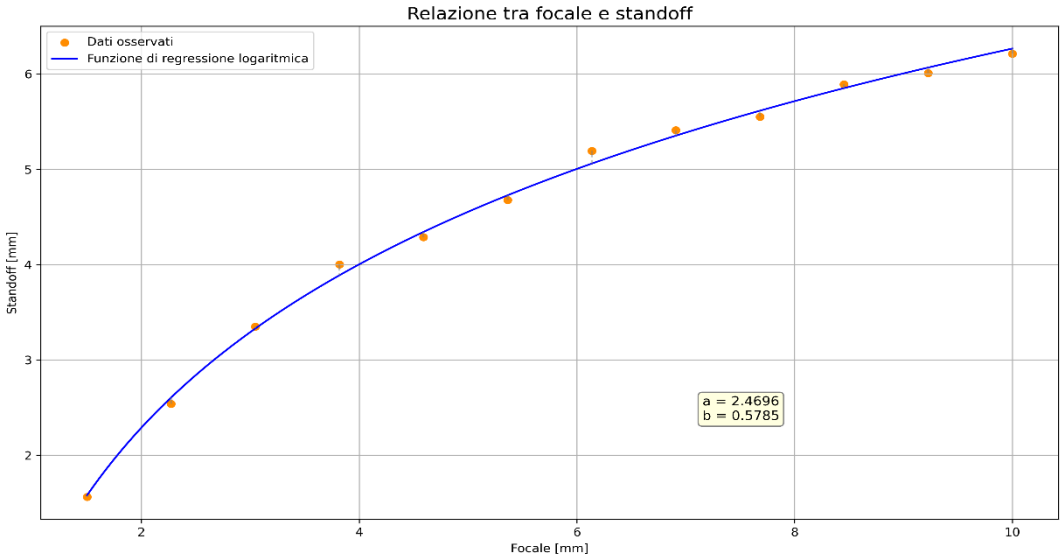


Figura 2.3: Regressione logaritmica

Nel contesto del taglio laser, la regressione logaritmica può essere utilizzata per descrivere la relazione tra lo standoff (distanza tra ugello e materiale) e la focale, quando l'aumento della focale produce un aumento rapido iniziale dello standoff che tende poi a stabilizzarsi:

$$Standoff = a \cdot \ln(Focale) + b$$

$\underline{Standoff}$: rappresenta il parametro di output che si vuole prevedere
$\underline{Focale}$: è la variabile indipendente (input),
$\underline{a}$: è il coefficiente che regola quanto rapidamente cresce lo $standoff$
$\underline{b}$: è il termine noto,
$\underline{ln}$: è il logaritmo naturale applicato alla variabile indipendente

Questi modelli possono essere facilmente ottenuti, ad esempio, con lo strumento di Curve Fitting di MATLAB, e utilizzati per rappresentare parametri di qualità del taglio come larghezza e profondità in funzione di variabili di processo quali potenza e velocità. La bontà del modello viene quindi validata attraverso metriche statistiche quali il coefficiente di determinazione R^2 e l'errore quadratico medio (RMSE), che valutano la qualità dell'adattamento della curva ai dati sperimentali.

2.4 Ottimizzazione tramite algoritmi metaeuristici

Le tecniche metaeuristiche includono algoritmi di ottimizzazione di alto livello progettati per affrontare problemi complessi, nei quali i metodi esatti risultano poco praticabili a causa di elevata dimensionalità, non linearità o presenza di molteplici minimi locali. Pur non garantendo la soluzione ottimale globale, queste tecniche forniscono soluzioni approssimate di alta qualità in tempi computazionalmente sostenibili. Ispirandosi a fenomeni naturali o comportamenti collettivi osservati in natura (come l'evoluzione biologica, la caccia dei predatori o il movimento degli sciami) [7][8], le metaeuristiche combinano due strategie fondamentali:

- Esplorazione: Per sondare aree nuove dello spazio delle soluzioni;
- Sfruttamento: Per affinare le soluzioni più promettenti già individuate.

Tra le tecniche più diffuse troviamo:

- **Algoritmi genetici (GA):**
  Gli algoritmi genetici sono ispirati al principio dell'evoluzione naturale. Partendo da una popolazione di possibili soluzioni a un problema; ciascuna soluzione viene valutata in base a quanto è "adatta" (fitness) rispetto all'obiettivo che vogliamo raggiungere. Le soluzioni migliori vengono "scelte" per riprodursi, combinando tra loro le proprie caratteristiche (crossover) e introducendo occasionalmente qualche cambiamento casuale (mutazione). Con questo meccanismo, generazione dopo generazione, la popolazione tende a evolversi, avvicinandosi sempre di più a una soluzione ottimale. La forza degli algoritmi genetici sta proprio nella loro capacità di esplorare ampi spazi di soluzioni, evitando di bloccarsi troppo presto in soluzioni non ottimali.

- **Particle Swarm Optimization (PSO):**
  Si ispira al comportamento collettivo degli stormi di uccelli o dei banchi di pesci, dove ogni possibile soluzione è rappresentata da una "particella" che si muove nello spazio delle soluzioni. Ogni particella è consapevole sia della posizione migliore che ha trovato personalmente, sia di quella trovata dal gruppo. Grazie a questa "memoria collettiva", le particelle tendono a muoversi verso le soluzioni migliori, bilanciando la ricerca di nuove aree con l'approfondimento delle zone promettenti già esplorate.

Questo approccio rende l'algoritmo molto efficace per trovare rapidamente soluzioni di buona qualità, specialmente in problemi con molti parametri da ottimizzare.

- **Whale Optimization Algorithm (WOA):**
  Il Whale Optimization Algorithm prende spunto dal comportamento di caccia delle megattere, in particolare dalla loro tecnica di "bubble-net feeding".
  In questa strategia, le balene creano spirali di bolle per intrappolare i pesci, avvicinandosi progressivamente alla preda. Allo stesso modo, le soluzioni candidate nell'algoritmo si muovono nello spazio delle soluzioni seguendo traiettorie spiraliformi, che permettono di esplorare in modo efficace sia le aree vicine che quelle più lontane. Alternando fasi di esplorazione e di sfruttamento, rendendolo in grado di adattarsi a problemi complessi e di evitare di bloccarsi in minimi locali.

- **Ant Lion Optimization (ALO):**
  L'Ant Lion Optimization è ispirato alla caccia degli antlioni, piccoli predatori che scavano trappole nella sabbia per catturare le formiche. Nell'algoritmo, le "formiche" rappresentano le soluzioni candidate, mentre gli "antlioni" rappresentano le forze che guidano la ricerca. Le formiche si muovono casualmente nello spazio delle soluzioni, ma vengono attirate verso gli antlioni migliori, che rappresentano le soluzioni più promettenti trovate fino a quel momento. Periodicamente, le posizioni degli antlioni vengono aggiornate, permettendo di affinare progressivamente la ricerca della soluzione ottimale. Questo meccanismo di attrazione e selezione rende ALO particolarmente efficace nell'evitare soluzioni subottimali.

Questi algoritmi sono stati applicati con successo all'ottimizzazione dei parametri di taglio laser, in particolare quando la funzione obiettivo è derivata da modelli di regressione complessi. Ad esempio, nel lavoro [4], vengono confrontate GA, PSO, ALO e WOA. I risultati mostrano che PSO ha raggiunto le migliori performance in termini di accuratezza e velocità di convergenza, con un errore relativo sulla profondità di taglio inferiore allo 0.2%. Anche per la larghezza del taglio, PSO ha ottenuto il miglior valore di costo, con un errore dello 0.19%. Inoltre, il confronto con un modello basato su rete neurale artificiale (ANN) ha evidenziato come l'errore relativo dei modelli metaeuristici (circa 3%) sia competitivo rispetto a metodi alternativi, tra cui appunto le reti neurali. Studi precedenti [5] avevano già impiegato

GA per minimizzare la rugosità superficiale o ottimizzare la geometria del taglio su materiali diversi. Tuttavia, l'integrazione di modelli di regressione accurati e la comparazione diretta tra più algoritmi evolutivi costituiscono il principale contributo innovativo del lavoro [4] per predire i parametri di taglio migliori.

2.5 Approcci basati su machine learning

Oltre all'impiego delle metaeuristiche, un approccio complementare ampiamente diffuso è rappresentato dalle reti neurali artificiali (ANN). Se nel paragrafo precedente le ANN venivano utilizzate come strumenti di validazione, in questa sezione se ne analizza l'impiego diretto come modello predittivo e di ottimizzazione.
Le ANN costituiscono un'alternativa flessibile ed efficace ai metodi classici per modellare il processo di taglio laser: addestrate su dati sperimentali, sono in grado di apprendere in modo non lineare la relazione tra i parametri di input (come potenza, velocità, pressione) e le metriche di qualità del taglio (rugosità, larghezza del kerf, presenza di bave, ecc.). Rispetto alle tecniche di regressione tradizionali, le reti neurali non richiedono l'esplicitazione della forma funzionale del modello, ma imparano direttamente dai dati, rendendole particolarmente adatte a descrivere fenomeni complessi e non lineari.
L'impiego delle ANN nel taglio laser si articola su più fronti:

- **Modellazione predittiva**: utilizzando strutture tipicamente feedforward, come le multilayer perceptron (MLP), è possibile prevedere con buona accuratezza gli output del processo a partire da configurazioni di input arbitrariamente complesse. Questi modelli sono stati adottati in numerosi studi per sostituire del tutto le funzioni di regressione, con errori medi spesso inferiori al 5%.
- **Ottimizzazione indiretta**: una volta addestrata, la rete può essere utilizzata come funzione obiettivo surrogata all'interno di algoritmi evolutivi, riducendo significativamente i tempi di valutazione delle soluzioni candidate.
- **Validazione delle soluzioni**: come già evidenziato nel lavoro [4], una rete neurale può servire a verificare la coerenza e la robustezza delle configurazioni ottimali proposte da algoritmi metaeuristici, offrendo una seconda stima indipendente basata sui dati.

Inoltre, recenti approcci ibridi combinano la potenza predittiva delle ANN con la capacità di esplorazione globale degli algoritmi metaeuristici, come nel caso delle combinazioni tra MLP e algoritmi genetici o swarm intelligence. In tali scenari, le reti neurali vengono utilizzate per ridurre drasticamente il numero di esperimenti

fisici o simulazioni numeriche richiesti, aumentando l'efficienza dell'intero processo di ottimizzazione.

2.5.1 Multi-Layer Perceptron (MLP)

Il Multi-Layer Perceptron (MLP) [18][19] è una rete neurale feedforward costituita da più livelli completamente connessi (fully-connected), utilizzata per modellare relazioni non lineari complesse tra input e output. È ispirato al funzionamento dei neuroni biologici, dove ogni nodo (neurone) riceve segnali, li elabora, e trasmette l'informazione ai livelli successivi.

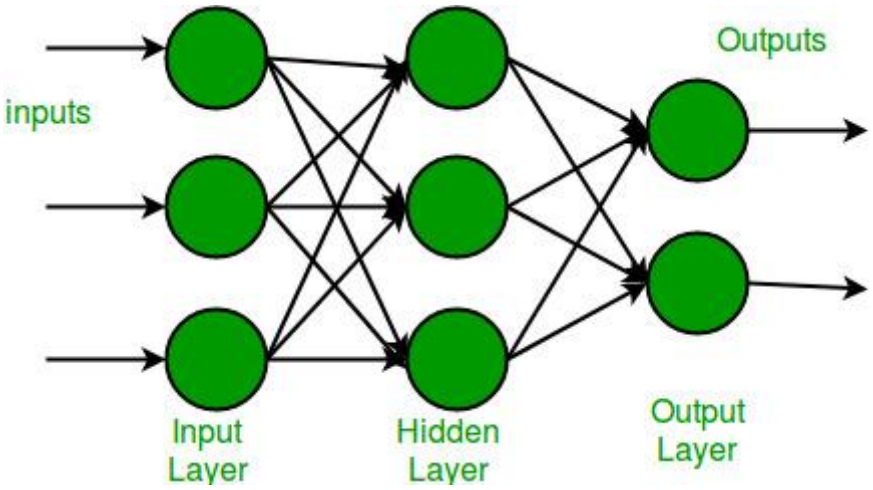


Figura 2.4: Multilayer Perceptron

La rete è composta da uno strato di input, uno o più strati nascosti e uno strato di output, dove:

- **Input Layer**: costituito da neuroni che ricevono direttamente le variabili di input del problema (ad esempio: potenza, pressione, velocità). Ogni neurone in questo strato rappresenta una caratteristica del dataset.
- **Hidden Layers**: questi strati rappresentano il cuore computazionale dell'MLP. Ogni layer nascosto è composto da neuroni che ricevono input dallo strato precedente e li trasformano tramite una combinazione lineare (somma pesata) seguita da una funzione di attivazione non lineare. È proprio questa sequenza di trasformazioni che permette alla rete di catturare relazioni complesse e non lineari nei dati. Gli hidden layer possono essere uno o molti, e il numero di

neuroni al loro interno può variare a seconda della complessità del problema. Ogni strato successivo consente alla rete di apprendere rappresentazioni sempre più astratte e informative dei dati iniziali

- **Output Layer**: genera la predizione finale del modello. Il numero di neuroni in questo strato dipende dal tipo di problema: uno per la regressione, o tanti quanti sono le classi nel caso di classificazione multiclasse.

Ogni connessione tra neuroni ha un peso e un bias, e l'output y di un neurone è calcolato in due fasi:

Somma pesata degli input:

$$z = \sum_{i=1}^{n} w_i x_i + b$$

La rete neurale calcola per ciascun neurone una combinazione lineare degli input ricevuti. Ogni input x_i è moltiplicato per un peso w_i, che rappresenta l'importanza assegnata a quell'informazione, e si somma un termine di bias b, che consente di spostare la funzione di attivazione. Il risultato è un valore intermedio z, che verrà poi trasformato dalla funzione di attivazione. Questo calcolo viene ripetuto in ogni layer nascosto, propagando l'informazione attraverso la rete dove ogni layer contribuisce così alla costruzione di una funzione via via più complessa, capace di descrivere in modo sempre più preciso la relazione tra input e output.

Applicazione della funzione di attivazione:

$$a = f(z)$$

Le funzioni di attivazione sono fondamentali nei nodi dei layer nascosti perché introducono non linearità nel modello, permettendo alla rete neurale di apprendere relazioni complesse. Le più comuni sono:

- **ReLU (Rectified Linear Unit):**

  $$f(z) = \max(0, z)$$

  Non attiva i neuroni con input negativi (output zero per $z < 0$).

- **Sigmoid:**

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Mappa i valori in un intervallo tra 0 e 1.
Usata in output layer per classificazione binaria, ma meno nei hidden layer perché può causare vanishing gradient.

- **Tanh (Tangente iperbolica):**

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Simile alla Sigmoid, ma con output tra -1 e 1.
Migliore della Sigmoid negli hidden layer, ma comunque soggetta a vanishing gradient per input estremi.

- **Softmax:**

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Usata nello strato di output per classificazione multiclasse.
Trasforma gli output grezzi in una distribuzione di probabilità.

Forward Propagation:

Durante la propagazione in avanti, i dati fluiscono dallo strato di input verso l'output attraversando gli strati nascosti, trasformandosi a ogni livello. Questo processo produce una predizione finale $\hat{y}$.

Funzione di perdita:

Per apprendere dai dati, l'MLP confronta $\hat{y}$. con il valore reale y usando una funzione di costo. Le più comuni sono:

Errore quadratico medio (MSE) per regressione:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Cross-Entropy per classificazione binaria:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right]$$

Backpropagation e ottimizzazione:

La backpropagation è il meccanismo attraverso cui una rete neurale apprende dai propri errori, aggiornando in modo iterativo i pesi che collegano i neuroni. Dopo aver ottenuto una predizione tramite il processo di forward propagation, la rete calcola quanto si è discostata dal risultato atteso tramite una funzione di perdita.
Il processo avviene in tre fasi principali:

1. **Calcolo del gradiente (derivata del loss rispetto ai pesi):**
   Usando la regola della catena del calcolo differenziale, si calcola l'effetto di ogni peso sul valore finale della perdita. Questo è ciò che consente di sapere "in che direzione" muovere ogni parametro per migliorare la predizione.

2. **Propagazione dell'errore all'indietro:**
   L'errore viene trasmesso a ritroso attraverso i layer, partendo dall'output e risalendo fino agli strati nascosti e all'input.Ogni neurone contribuisce parzialmente all'errore totale e riceve un segnale su quanto deve "correggersi".

3. Aggiornamento dei pesi:

Infine, i pesi vengono aggiornati con una regola basata sul gradient descent. Per un peso w, la formula è:

$$w \leftarrow w - \eta \cdot \frac{\partial L}{\partial w}$$

dove:

- $\boldsymbol{w}$ è il peso da aggiornare,
- $\boldsymbol{\eta}$ è il learning rate, ovvero la velocità di apprendimento,
- $\frac{\partial L}{\partial w}$ è il gradiente della funzione di perdita L rispetto al peso.

Questo processo si ripete per molte epoche (iterazioni sull'intero dataset), finché la rete minimizza la funzione di perdita e migliora le sue predizioni.

Infine, stanno emergendo applicazioni più avanzate delle reti neurali profonde (deep learning), in particolare quando i dataset disponibili sono sufficientemente ampi.

Architetture più complesse come le convolutional neural networks (CNN) o le long short-term memory (LSTM) vengono esplorate per includere dati visivi (es. immagini dei bordi di taglio) o dinamici (es. variazioni temporali nei parametri del processo).

2.6 Limiti delle soluzioni esistenti e motivazioni del presente lavoro

Nonostante i risultati promettenti riportati da numerosi approcci presenti in letteratura, persistono alcune criticità rilevanti:

- Gran parte degli studi si concentra sull'ottimizzazione di una singola configurazione target, con l'obiettivo di ottenere precisi valori quantitativi in output (ad esempio: rugosità, larghezza del taglio, profondità).
- I metodi proposti richiedono spesso ampie campagne sperimentali e grandi volumi di dati, con combinazioni di parametri costruite ad hoc. Tali condizioni

risultano difficilmente replicabili in ambienti produttivi reali, soprattutto in presenza di dataset limitati.

Il presente lavoro propone un cambio di paradigma: affrontare il problema dell'ottimizzazione e della correzione automatica dei parametri di taglio laser a partire da configurazioni iniziali che generano tagli difettosi, trasformandole in configurazioni migliorative, capaci di garantire una qualità del taglio accettabile.
A differenza degli approcci tradizionali, che mirano a raggiungere valori numerici target ben definiti, l'obiettivo qui è di tipo qualitativo: si opera con una classificazione binaria del risultato ("buono" o "cattivo"), senza necessità di misurazioni metrologiche dettagliate (es. rugosità o profondità). Inoltre, il metodo è stato ideato per gestire scenari caratterizzati da una limitata disponibilità di dati, come spesso accade in contesti industriali reali, dove non è possibile disporre di dataset ampi né condurre numerosi test sperimentali. Pur senza ambire a prestazioni ottimali in senso assoluto, l'approccio consente di modellare il problema in forma semplificata, affrontando sottoproblemi specifici e contribuendo comunque a migliorare la configurazione di taglio, riducendo il numero di tentativi necessari a effettuare un taglio esente da difetti.

Capitolo 3

Architettura e implementazione

In questo capitolo viene descritta l'architettura del sistema sviluppato e le tecniche di intelligenza artificiale impiegate per correggere configurazioni di taglio laser errate, trasformandole in configurazioni ottimali, in grado di produrre tagli esenti da difetti, ripercorrendo la traccia di alcuni concetti visti nel capitolo dedicato allo stato dell'arte. L'obiettivo è quello di supportare l'operatore nella selezione automatica o semi-automatica dei parametri di processo, riducendo il numero di tentativi necessari aumentando la qualità complessiva del taglio.

Inoltre, i dati disponibili possono risultare rumorosi o non bilanciati, con una prevalenza di classi di difetti rispetto ad altre. Questa situazione può compromettere l'efficacia degli algoritmi utilizzati poiché i modelli tendono a sovraddestrarsi sulle classi maggioritarie e a mostrare una scarsa sensibilità nei confronti dei casi critici. Le tecniche che verranno descritte nei paragrafi successivi consentono tuttavia al sistema di mantenere una buona capacità predittiva anche in presenza di dataset limitati, ponendo le basi per una futura implementazione in un contesto operativo reale. In seguito viene mostrata la pipeline relativa alla parte di ottimizzazione del dataset:

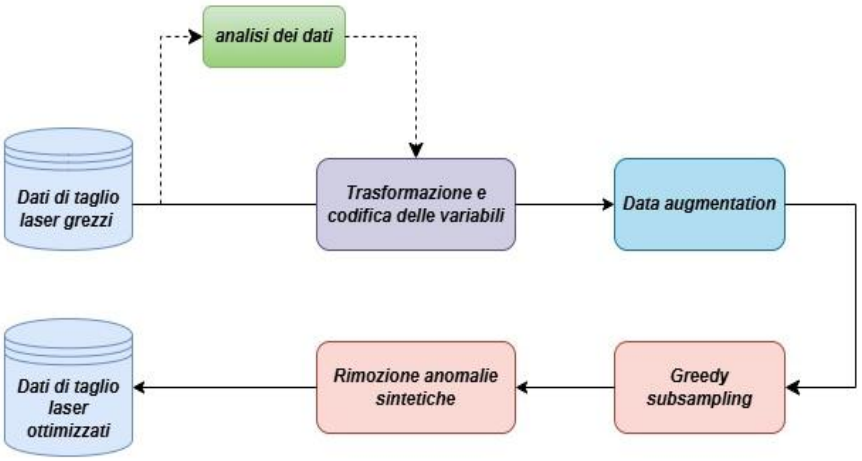


Figura 3.0: Pipeline di ottimizzazione del dataset

3.1 Analisi dei dati

Prima di passare alla fase di correzione delle configurazioni di taglio, è fondamentale comprendere in modo approfondito i dati a disposizione e le loro limitazioni. L'analisi esplorativa dei dati (*Exploratory Data Analysis*, EDA) consente di identificare tendenze, correlazioni, anomalie e pattern utili per una successiva modellazione più efficace e robusta. In particolare, verranno analizzate le variabili di input relative al processo di taglio laser, esaminando la loro distribuzione statistica, le interazioni reciproche e il loro legame con l'esito del taglio. Questa fase è cruciale per valutare la qualità del dataset, individuare eventuali problemi (come squilibri di classe o outliers) e stabilire quali caratteristiche risultino più influenti nel determinare la qualità finale del taglio. L'analisi sarà condotta attraverso strumenti statistici e grafici, tra cui matrici di correlazione, grafici di distribuzione e tecniche di riduzione dimensionale come la PCA, per ottenere una visione più chiara e sintetica della struttura dei dati.

3.1.1 Panoramica delle variabili in gioco

Le principali variabili coinvolte nel processo di taglio laser per il caso preso in esame comprendono:

Parametri di taglio:

- **Velocità di avanzamento**:
  Indica la velocità con cui la testa di taglio si muove lungo il percorso di taglio. È espressa in millimetri al minuto (mm/min) ed è rappresentata con l'alias "$CONTOUR\_SPEED\ [mm/min]$

- **Potenza del laser**:
  Rappresenta la quantità di energia fornita dal laser, espressa in Watt (W) ed è rappresentata con l'alias "$LASER\_POWER[W]$".

- **Pressione del gas assistente:**
  Indica la pressione del gas (aria, ossigeno o azoto) soffiato attraverso l'ugello durante il taglio, espressa in (bar). È rappresentata con l'alias "$CONTOUR\_GAS\_PRESSURE\ [bar]$ ".

- **Distanza focale**:
  È la distanza tra la lente focale del sistema ottico e il piano superiore del materiale, determina il dove il raggio laser è concentrato al massimo della sua potenza, espressa in millimetri (mm). È rappresentata con l'alias "$CONTOUR\_FOCAL\ [mm]$"

- **Standoff**:
  Rappresenta la distanza tra l'ugello di uscita del gas assistente e la superficie del materiale da tagliare, espressa in (bar). È rappresentata con l'alias "$CONTOUR\_NOZZLE\_DISTANCE\ [mm]$ "

- **Ugello**:
  È il componente terminale della testa di taglio laser, posto subito dopo la lente focale, Serve a convogliare e direzionare il gas assistente (come aria, ossigeno o azoto) sul punto di taglio ed è suddiviso in base alla tipologia e diametro. E' rappresentata con l'alias "$NOZZLE\_TYPE$"

- **Tipo di gas**:
  È la tipologia di gas impiegata per assistere il taglio. È rappresentata con l'alias "GAS_TYPE". Può essere N_2 , O_2 o un mix di gas.

- **Tipo di laser**:
  È la tipologia di laser impiegata per effettuare il taglio .È rappresentato con l'alias "$LASER\_TYPE$", in questo caso specifico abbiamo un laser in fibra all'itterbio (YLS) e un laser CO_2 (CF)

- **Modalità del laser**:
  indica il modo in cui il fascio laser viene emesso durante il taglio. È rappresentata con l'alias "$CONTOUR\_LASER\_MODE$ ". Il laser può essere emesso in modalità continua (CW) o pulsata (GP).

Parametri caratterizzanti del materiale:

- **Nome o tipo del materiale**:
  I materiali, in questo caso quelli metallici, sono espressi secondo lo standard Europeo EN che associa delle sigle ai diversi materiali come acciai, alluminio o rame, Come ad esempio: *S420MC* o *X5CrNi18-10* che rappresentano rispettivamente due sigle di acciai diversi. È rappresentato con l'alias "*MATERIAL_NAME_TULUS*"

- **Spessore**:
  Rappresenta lo spessore del materiale su cui effettuare il taglio, è espresso in (mm). È rappresentato con l'alias "$THICKNESS\_TULUS\ [mm]$"

Variabili target:

- **Tipo di difetto**:
  Rappresenta il tipo di difetto (se presente) riscontrato una volta effettuato il taglio. È rappresentato con l'alias "$DEFECT\_TYPE$".

- **Qualità del taglio**:
  Indica la qualità del taglio ottenuto. In assenza di difetti assume il valore 'Good' altrimenti 'Bad'. È rappresentato con l'alias "$QUALITY\_CUT$".

3.1.2 Descrizione del dataset

Il dataset utilizzato per l'addestramento e la valutazione dei modelli è costituito da dati sperimentali raccolti durante un numero limitato di prove di taglio laser eseguite su materiali metallici di diversa natura, spessore e configurazione di processo. Complessivamente, il dataset comprende circa 320 campioni, ciascuno dei quali è descritto attraverso parametri di processo, caratteristiche del materiale e variabili target, dove queste ultime rappresentano l'esito qualitativo del taglio e l'eventuale presenza di difetti.
I dati sono stati raccolti in un contesto industriale reale, con il supporto di un'azienda partner, in collaborazione con il Politecnico di Torino. Tuttavia, come spesso accade nei contesti produttivi, la disponibilità dei dati è limitata da diversi fattori, tra cui i costi elevati delle prove sperimentali, la difficoltà di replicare le condizioni operative e la scarsa disponibilità di etichette affidabili in presenza di difetti.
Il dataset presenta alcune criticità tipiche dei contesti reali, che hanno richiesto interventi specifici in fase di pre-processing:

1. **Squilibrio tra classi**:
   La variabile target *QUALITY_CUT* risulta fortemente sbilanciata, con una netta prevalenza di esempi etichettati come 'Good' rispetto a quelli 'Bad'. Questo potrebbe introdurre bias nei modelli di classificazione, penalizzando la capacità predittiva sui casi meno frequenti.

2. **Distribuzione disomogenea di alcune variabili categoriche**:
   Come il tipo di ugello o lo spessore del materiale, che risultano dominati da poche modalità particolarmente ricorrenti a scapito di altre scarsamente rappresentate.

3. **Presenza di valori duplicati**:
   In alcuni casi concentrati proprio nelle classi meno rappresentate, aggravando ulteriormente il problema dello sbilanciamento.

Tali problematiche hanno reso necessarie una fase accurata di pulizia, riequilibrio e trasformazione dei dati, che verrà approfondita nel paragrafo successivo.

3.1.3 Trasformazione e codifica delle variabili

Una volta identificate le principali variabili in gioco, è necessario trasformarle in una forma adatta all'analisi statistica e alla modellazione predittiva. In questa fase si interviene per rendere il dataset più uniforme, correggere eventuali incongruenze e

convertire le informazioni testuali o categoriali in un formato numerico interpretabile dagli algoritmi, rimuovere valori o categorie troppo singolari. Le variabili coinvolte nel contesto preso in esame e discusse nel paragrafo 3.1.1 vengono suddivise in continue e categoriali, dove le continue sono variabili rappresentate da quantità misurabili e possono assumere valori numerici, su scala appunto continua oppure discretizzata. D'altra parte, le variabili categoriali rappresentano gruppi o categorie non numerici, come ad esempio il colore, il sesso o una tipologia. Queste variabili possono essere qualitative (senza un ordine logico) o ordinali (con un ordine logico).

Tra le variabili continue abbiamo:

- $CONTOUR\_SPEED$ $[mm/min]$
- $LASER\_POWER$ $[W]$
- $CONTOUR\_GAS\_PRESSURE$ $[bar]$
- $CONTOUR\_FOCAL$ $[mm]$
- $CONTOUR\_NOZZLE\_DISTANCE$ $[mm]$
- $THICKNESS\_TULUS$ $[mm]$

Mentre le variabili categoriche sono:

- GAS_TYPE
- $LASER\_TYPE$
- $NOZZLE\_TYPE$
- $CONTOUR\_LASER\_MODE$
- $MATERIAL\_NAME\_TULUS$

Per meglio comprendere la distribuzione delle variabili analizzate, la Figura 3.1 mostra un insieme di istogrammi rappresentativi dei valori assunti da ciascuna feature del dataset, che comprende variabili continue e categoriali. I grafici evidenziano importanti caratteristiche statistiche, come la presenza di sbilanciamenti nelle categorie, valori dominanti e possibili outliers, che hanno guidato le successive scelte di preprocessing. Prima di affrontare un'analisi grafica è necessario gestire l'incongruenza nella formattazione delle variabili categoriche nominali, la gestione dei valori vuoti (o NaN) e oltretutto è stata aggiunta la nuova categoria di difetto '$No\ defects$' all'interno della feature $DEFECT\_TYPE$ per attribuire un valore a quest'ultimo quando il campo $QUALITY\_CUT$ assume il valore '$Good$'. Inoltre, l'aggiunta del valore '$No\ defects$' al campo $DEFECT\_TYPE$ sarà utile per la creazione di un algoritmo di classificazione dei difetti a partire dalle configurazioni di taglio.

Figura 3.1: Distribuzione delle variabili

Nel contesto dei modelli di machine learning, è noto che il rapporto tra il numero di feature e il numero di campioni rappresenta un elemento critico che può influenzare negativamente le prestazioni. Un numero eccessivo di variabili, soprattutto se poco informative o scarsamente rappresentate, tende infatti ad aumentare il rischio di overfitting, ridurre la capacità di generalizzazione del modello e peggiorare l'efficienza computazionale. All'interno del dataset analizzato, si osservano alcune situazioni simili ben evidenziate nella Figura 3.1. Alcune variabili categoriali, come ad esempio $NOZZLE\_TYPE$, includono un elevato numero di categorie distinte, ma solo poche di esse risultano effettivamente rappresentate da un numero significativo

di campioni. Alcuni valori, come '*SMT 3.5*' e '*ST 2.5*', compaiono in modo estremamente raro, rendendo il loro contributo trascurabile dal punto di vista statistico. Allo stesso modo, variabili come *TECHNOLOGY_GAS, LASER_TYPE* e *CONTOUR_LASER_MODE* risultano costanti o quasi costanti, cioè, assumono un solo valore oppure una distribuzione fortemente sbilanciata, e pertanto non forniscono alcuna informazione utile in ottica predittiva. In presenza di queste condizioni, si è reso necessario un intervento di riduzione della dimensionalità, volto a migliorare la qualità e la compattezza del dataset. Nello specifico, le variabili costanti o quasi costanti sono state eliminate, poiché ridondanti e prive di potere discriminante. Per le variabili categoriali con numerose modalità poco rappresentate, si è optato per la rimozione delle categorie più rare, come nel caso dei già citati tipi di ugello, al fine di evitare l'introduzione, tramite codifica one-hot, di colonne scarsamente popolate e statisticamente irrilevanti. Queste operazioni preliminari si sono rivelate fondamentali per garantire un corretto bilanciamento tra complessità del modello e significatività dell'informazione presente nei dati, e verranno ulteriormente approfondite nei paragrafi successivi.

3.1.4 Aggregazione semantica dei difetti

Un ulteriore passaggio fondamentale ha riguardato la semplificazione delle classi relative ai difetti di taglio. Come evidenziato nella fase esplorativa, la variabile *DEFECT_TYPE* presenta un'elevata granularità, con numerose sottocategorie che, sebbene rappresentino fenomeni fisicamente distinti, risultano poco rappresentate a livello statistico. Questa frammentazione eccessiva avrebbe compromesso la stabilità e l'affidabilità della fase di classificazione, oltre a rendere più complesso l'addestramento del modello, soprattutto in presenza di un dataset limitato in termini di numerosità complessiva. Per questo motivo, è stato scelto di accorpare alcune delle sottoclassi in macrocategorie più generali, che potessero mantenere il significato tecnico del difetto pur migliorando la coerenza e la distribuzione dei dati. Ad esempio, le classi "*short black burr*" e "*long black burr*" sono state unificate all'interno della categoria "*Burr*", così come le classi "*light plasma*" e "*intense plasma*" sono confluite in un'unica classe denominata "*Plasma*". La stessa logica è stata applicata ad altre tipologie di difetti, seguendo un criterio sia semantico che quantitativo, al fine di ottenere classi più robuste e informative per la fase di modellazione. La Figura 3.2 mostra graficamente la struttura dell'aggregazione effettuata:

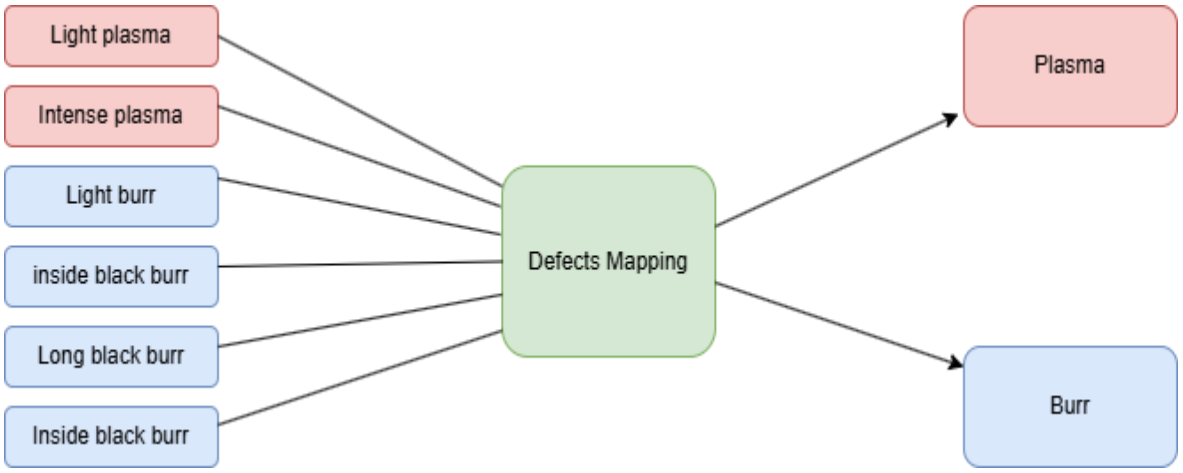


Figura 3.2: Aggregazione delle classi di difetti

Questa operazione di consolidamento non solo ha permesso di migliorare l'equilibrio tra le classi target, ma ha anche contribuito a ridurre la complessità del problema di classificazione, rendendolo più gestibile e adatto alla quantità di dati disponibili. Nella figura 3.4 è mostrato il risultato di questa fase di preprocessing dei dati:

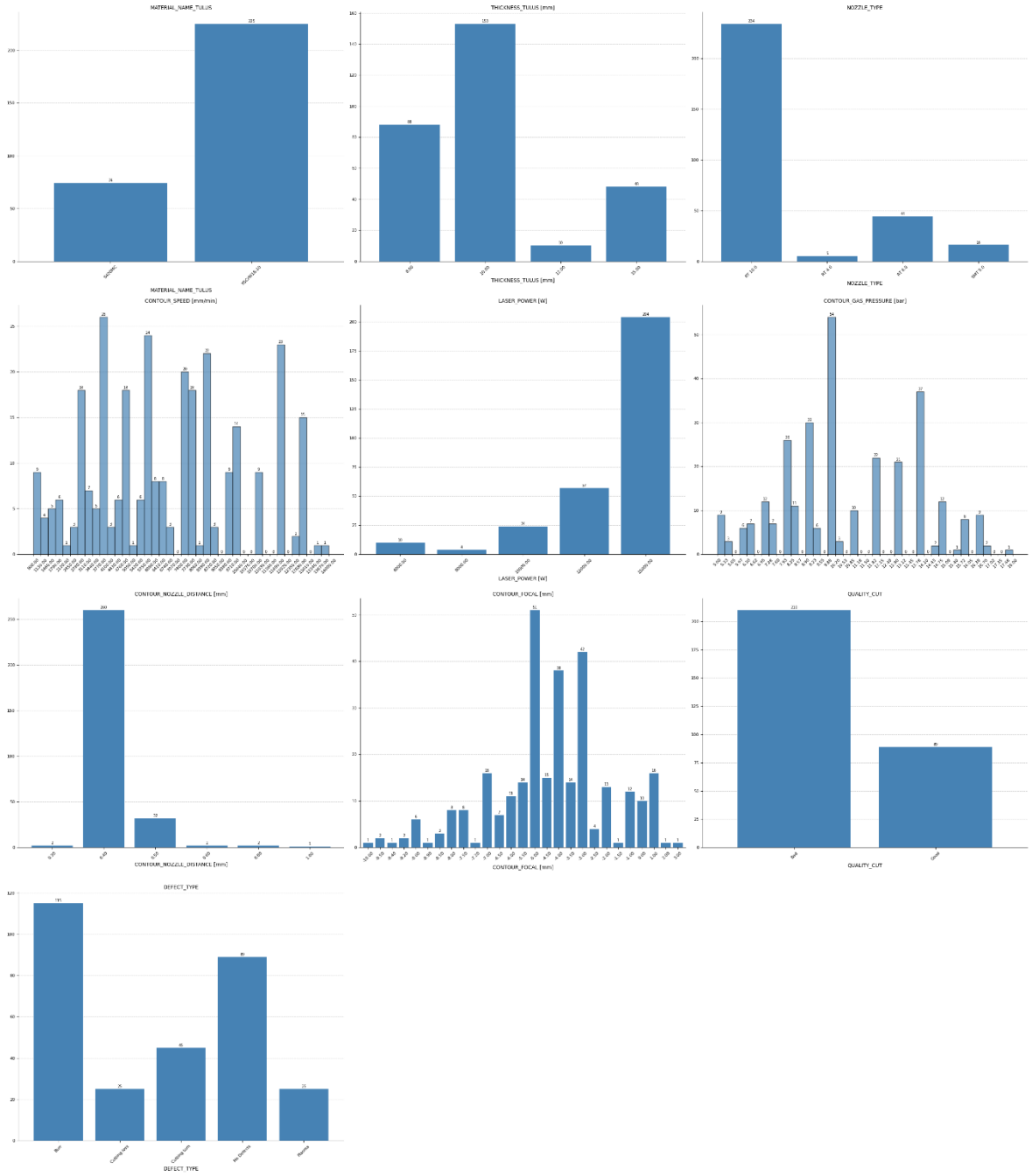


Figura 3.4: Distribuzione delle variabili post-processing

In Figura 3.4 è possibile osservare una versione più pulita del dataset, contenente solo le variabili selezionate per le fasi successive dell'analisi. Nei paragrafi successivi verrà condotta un'analisi grafica che evidenzia le correlazioni tra le diverse variabili.

3.1.5 Analisi bivariata delle variabili

Dopo aver effettuato una pulizia e trasformazione preliminare dei dati, è utile analizzare in modo più approfondito le relazioni tra le variabili, considerando le associazioni a coppie mediante un'analisi bivariata, consentendo di esplorare la presenza di correlazioni, dipendenze o pattern significativi tra due variabili, siano esse continue o categoriali. Questo tipo di analisi è fondamentale per comprendere meglio la struttura del dataset e per individuare relazioni potenzialmente informative ai fini della modellazione. Nel caso di variabili numeriche continue, si fa uso di scatterplot per osservare la forma della distribuzione congiunta, valutando la presenza di tendenze lineari, curve o pattern specifici. Nelle figure 3.5, 3.6, 3.7, 3.8 e 3.9 sono mostrati gli scatterplot delle variabili continue del dataset per ogni difetto.

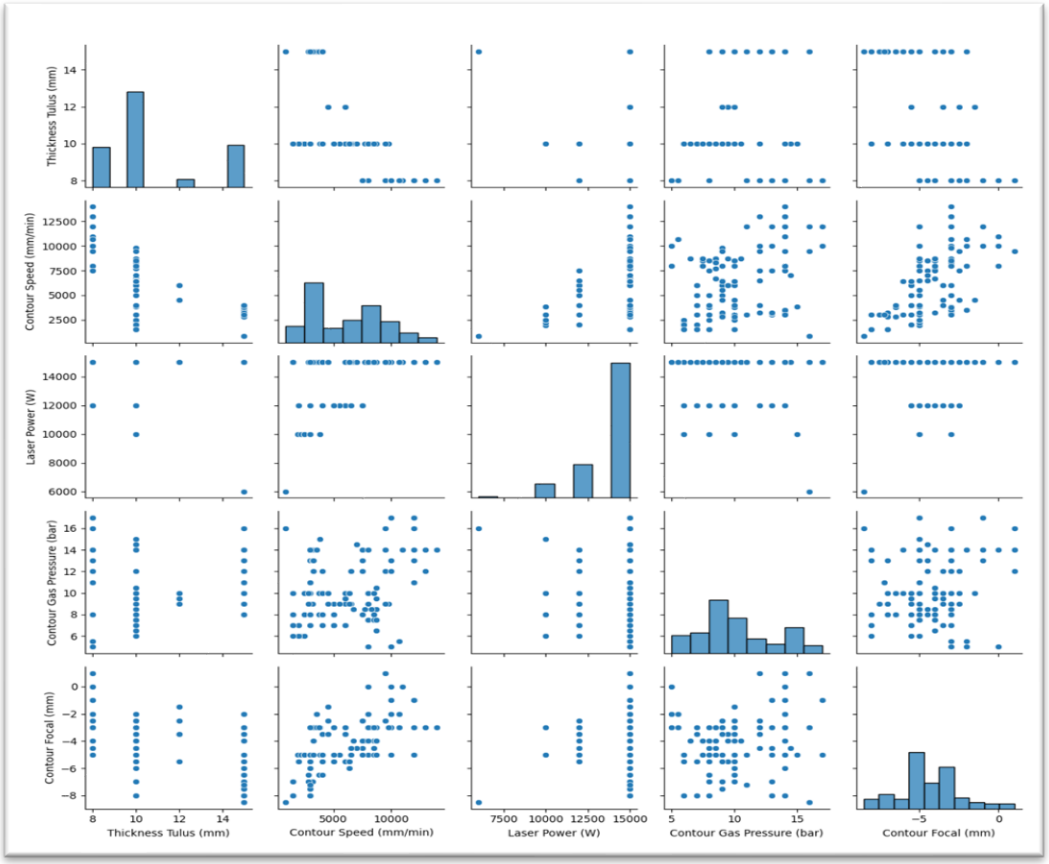


Figura 3.5: analisi bivariata delle variabili per il difetto Burr

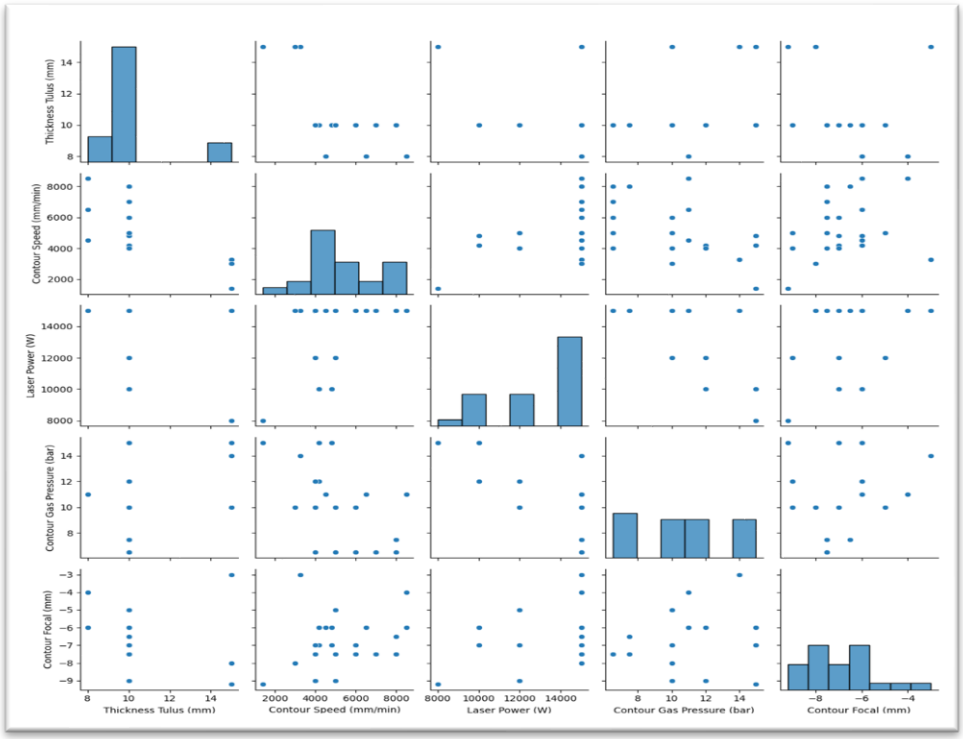


Figura 3.6: analisi bivariata delle variabili per il difetto Plasma

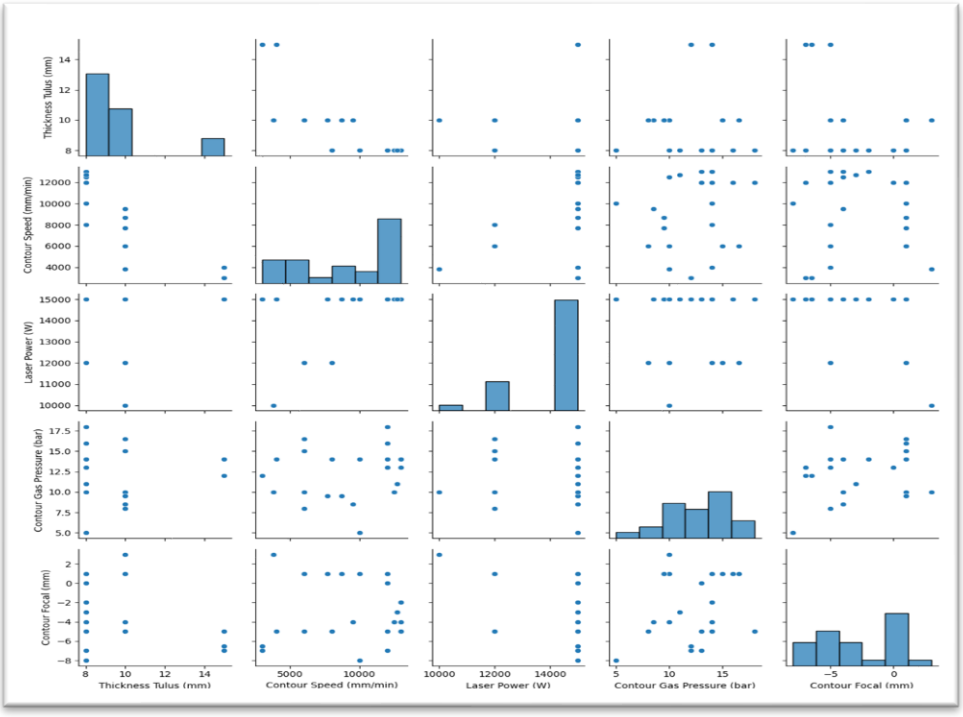


Figura 3.7: analisi bivariata delle variabili per il difetto Cutting loss

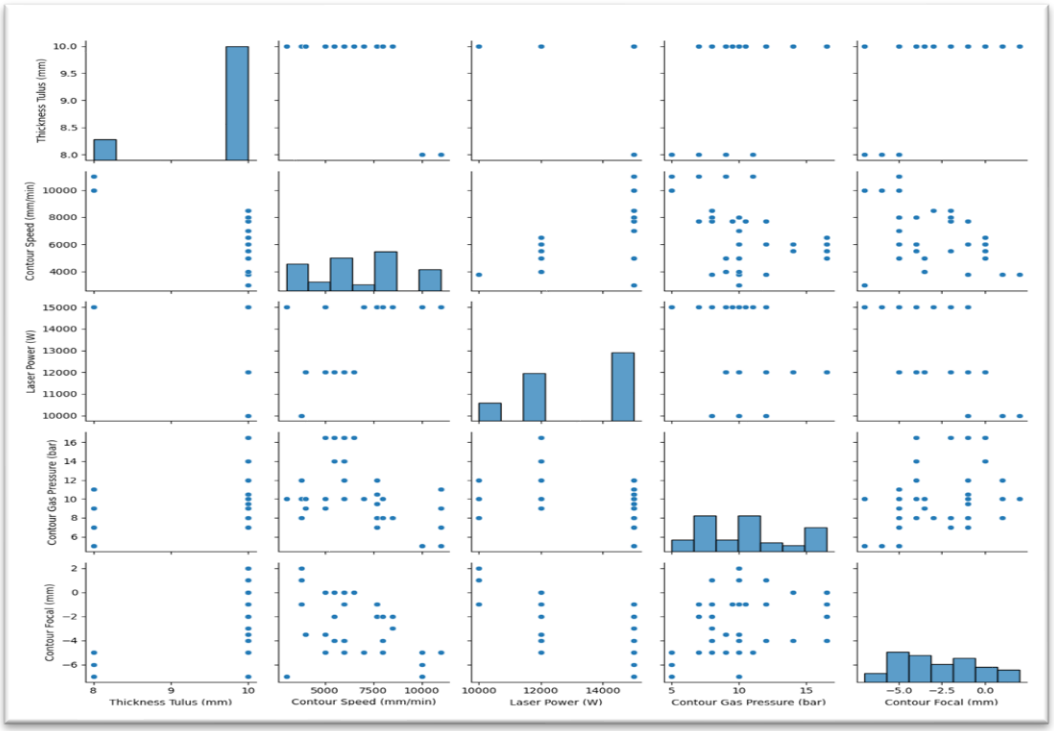
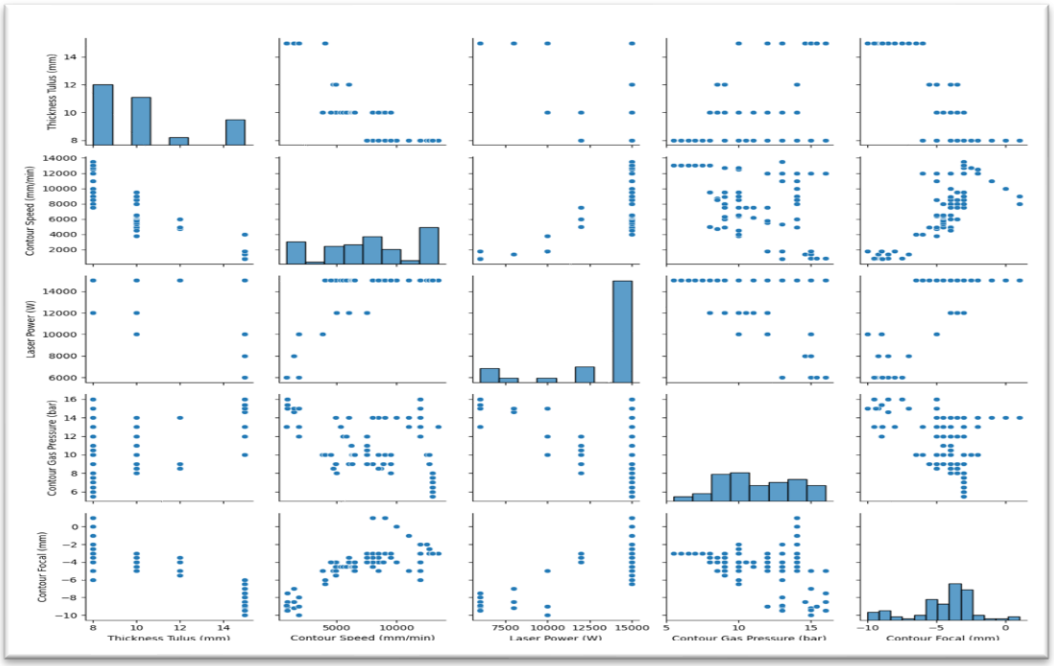


Figura 3.8: analisi bivariata delle variabili per il difetto Cutting torn



Dall'analisi bivariata delle variabili continue emerge che, per i difetti '$Plasma$', '$Cutting\ loss$' e '$Cutting\ torn$', non si osservano pattern evidenti, probabilmente a causa del numero ridotto di esempi disponibili. Al contrario, per i difetti '*Burr'* e '*No defects'*, più rappresentati nel dataset, è possibile identificare alcune relazioni tra coppie di variabili.
Nel caso del difetto '$Burr$', si nota che $CONTOUR\_SPEED$ e $THICKNESS\_TULUS$ formano nuvole di punti verticali, segno che a ogni spessore corrisponde una certa gamma di velocità, senza però una chiara correlazione lineare.
Più interessante è il legame tra $CONTOUR\_FOCAL$ e CONTOUR_SPEED, dove si osserva una relazione inversa: all'aumentare della distanza focale, la velocità tende a diminuire. Questo riflette dinamiche note del taglio laser, in cui una messa a fuoco più profonda impone parametri più cauti. Una tendenza simile, anche se meno netta, si riscontra tra $LASER\_POWER$ e $CONTOUR\_SPEED$, che appaiono moderatamente correlate in modo diretto.
Per la classe '*No defects'*, le variabili $THICKNESS\_TULUS$ e $LASER\_POWER$ mostrano valori ripetuti in modo discreto, segnale di un uso frequente di settaggi standard. Al contrario, $CONTOUR\_SPEED$ presenta una maggiore variabilità, con zone più densamente popolate, indice di maggiore flessibilità operativa. Anche $CONTOUR\_GAS\_PRESSURE$ e $CONTOUR\_FOCAL$ mostrano una certa dispersione, ma con picchi su valori ricorrenti, probabilmente per ragioni tecniche.
Infine, si conferma una relazione interessante tra $CONTOUR\_FOCAL$ e

3.1.6 Correlazione lineare mediante coefficiente di Pearson

Come ulteriore approfondimento, è possibile valutare la forza e la direzione delle relazioni lineari tra le variabili calcolando la matrice delle correlazioni secondo il coefficiente di Pearson. Questo tipo di analisi permette di quantificare il grado di dipendenza lineare tra coppie di variabili, fornendo un'indicazione numerica che va da -1 (correlazione negativa perfetta) a +1 (correlazione positiva perfetta), con 0 ad indicare assenza di correlazione lineare. A supporto dell'analisi numerica, la matrice delle correlazioni può essere visualizzata graficamente sotto forma di heatmap, facilitando l'individuazione immediata di relazioni forti (positive o negative) tra variabili.

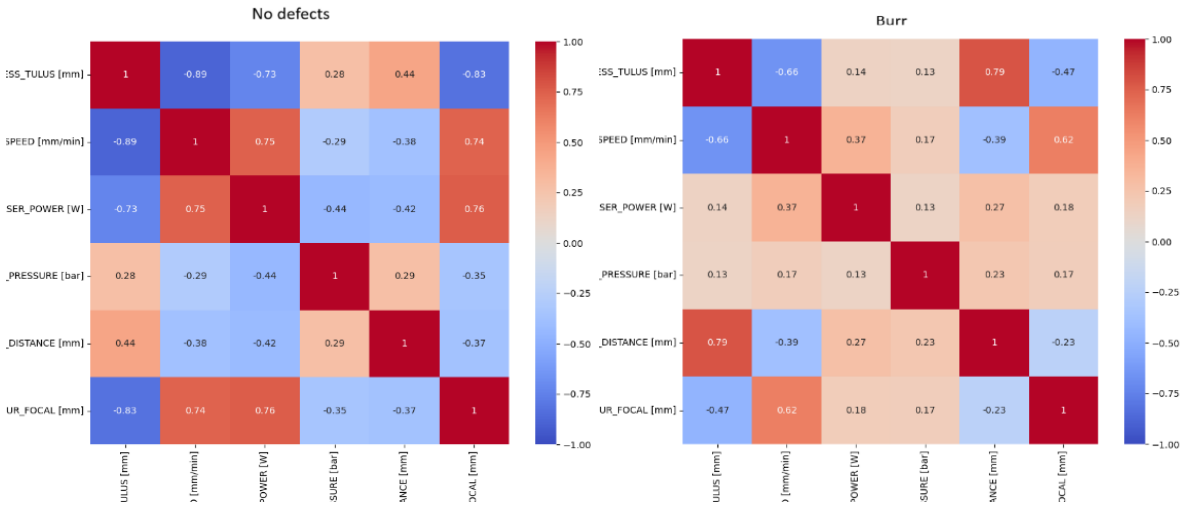


Figura 3.10: Matrici di correlazione per 'No defects' e 'Burr'

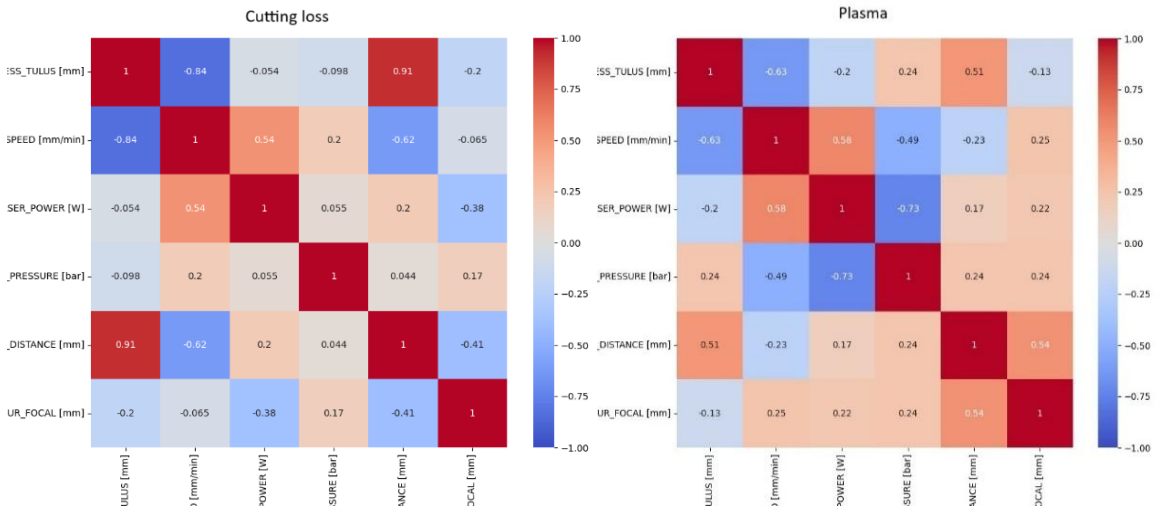


Figura 3.11: Matrici di correlazione per ‘Cutting loss’ e ‘Plasma’

Osservando i valori presenti nelle matrici di correlazione riportate in Figura 3.10, emerge come la classe ‘*No defects*’ mostri relazioni più definite e coerenti tra i diversi parametri di taglio, fornendo spunti interessanti sull’equilibrio dei settaggi nei casi in cui il risultato del processo è privo di difetti. Osservando la matrice di correlazione relativa alla classe ‘*No defects*’, si evidenziano alcune relazioni lineari piuttosto forti tra i parametri di taglio, che risultano particolarmente interessanti poiché riflettono le condizioni in cui il taglio avviene senza generare difetti. In particolare, si nota una marcata correlazione negativa tra lo spessore del materiale (*THICKNESS_TULUS*) e la velocità di taglio (*CONTOUR_SPEED*) pari a circa -0.89. Questo suggerisce che, nei casi in cui il taglio è qualitativamente buono, all’aumentare dello spessore è necessario ridurre la velocità per mantenere risultati ottimali. Un’altra correlazione negativa significativa riguarda la relazione tra lo spessore e il valore della focale (*CONTOUR_FOCAL*), con un coefficiente di circa -0.83. Anche in questo caso, il dato riflette una relazione coerente dal punto di vista fisico: spessori maggiori richiedono probabilmente una messa a fuoco più stretta del laser per penetrare efficacemente il materiale. Al contrario, nella classe ‘*Burr*’ tali relazioni risultano più deboli o meno strutturate, evidenziando una maggiore dispersione nei parametri associati a questo tipo di difetto. Questo rafforza l’idea che la bontà del taglio sia legata a un equilibrio preciso tra le variabili di processo, equilibrio che può essere rilevato anche attraverso una semplice analisi di correlazione lineare, che conferma in parte alcune relazioni viste nell’analisi bivariata.

3.2 Data augmentation mediante generazione sintetica di nuovi dati

Per far fronte alla scarsità di campioni, sia in termini di numerosità complessiva sia di squilibrio tra le classi di difetti, è stata introdotta una fase di data augmentation basata sulla generazione sintetica di nuovi campioni, con l'obiettivo di arricchire il dataset esistente mantenendo la coerenza statistica tra le variabili originali. In particolare, sono stati utilizzati due modelli generativi basati su tecniche di deep learning: un TVAE (Tabular Variational Autoencoder) [9] e un CTGAN (Conditional Tabular Generative Adversarial Network) [10].

3.1 TVAE (Tabular Variational Autoencoder)

Il Tabular Variational Autoencoder (TVAE) è un modello generativo basato sull'architettura dei Variational Autoencoder (VAE), appositamente progettato per gestire dati tabellari contenenti sia variabili numeriche che categoriali. Il TVAE mira a rappresentare i dati originali in uno spazio latente di dimensioni ridotte, apprendendo una distribuzione probabilistica da cui generare nuovi campioni sintetici coerenti con le caratteristiche del dataset reale.

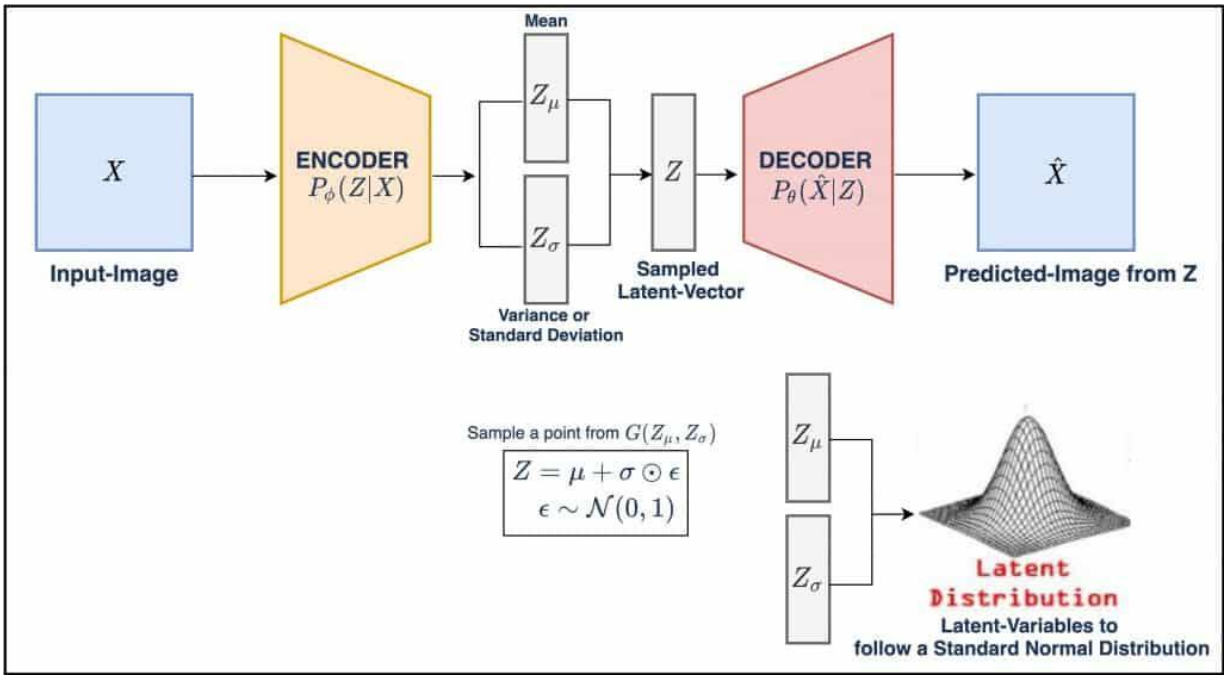


Figura 3.12: Vatiational Autoencoder [11]

L'architettura del TVAE si compone di tre parti fondamentali:

- **Encoder**:
  Un encoder che mappa un dato di input x in una distribuzione latente parametrizzata da media μ e deviazione standard σ, ovvero:

  $$q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x)I)$$

  dove z è il vettore latente, e ϕ rappresenta i parametri dell'encoder.

- **Sampling layer**:
  Esegue il campionamento dallo spazio latente.

- **Decoder**:
  Che ricostruisce i dati a partire da un campione $\boldsymbol{z}$ dallo spazio latente secondo la distribuzione:

  $$p_\theta(x|z)$$

  con θ i parametri del decoder.

In un Variational Autoencoder, si ipotizza che i dati osservabili $\boldsymbol{x}$ siano generati da una variabile latente non osservabile $\boldsymbol{z}$, attraverso una distribuzione condizionata

$$p_\theta(x|z)$$

dove la variabile latente $\boldsymbol{z}$ è modellata con una prior $p(z)$ tipicamente una normale standard $N(0, I)$. L'obiettivo è approssimare la distribuzione posteriore $p(z \mid x)$ tramite un encoder parametrizzato $q_\phi(z \mid x)$, con parametri ϕ. Il modello viene addestrato massimizzando l'Evidence Lower Bound (ELBO):

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\mathrm{KL}}(q_\phi(z|x) \parallel p(z))$$

dove:

$E_{q_\phi(z|x)}[log\, p_\theta(x \mid z)]$ *è* la loss di ricostruzione, che misura l'errore tra

dati originali x e quelli ricostruiti dal decoder, valutato in modo differente per variabili numeriche e categoriali (ad esempio MSE per le prime e cross-entropy per le seconde).

- D_{KL} è la divergenza di Kullback-Leibler (KL) tra la distribuzione latente appresa $q_\phi(z \mid x)$ e la distribuzione prior $p(z)$, solitamente una normale standard *N(0,I)*. Questo termine regolarizza il modello, evitando che si adatti eccessivamente ai dati di training.

Questo approccio consente al TVAE di generare nuovi campioni realistici campionando $z \sim N(0, I)$ e passando z nel decoder per ottenere:

$$\hat{x} = p_\theta(x|z)$$

3.2 CTGAN (Conditional Tabular Generative Adversarial Networks)

Per affrontare la sfida della generazione di dati tabellari contenenti sia variabili numeriche che categoriche, è stato utilizzato anche il modello CTGAN (Conditional Tabular GAN). A differenza dei GAN tradizionali, che faticano a modellare efficacemente dati misti e distribuzioni sbilanciate, il CTGAN è stato progettato appositamente per il dominio tabellare e si è dimostrato molto efficace nel riprodurre fedelmente le relazioni tra le variabili nei dati originali.

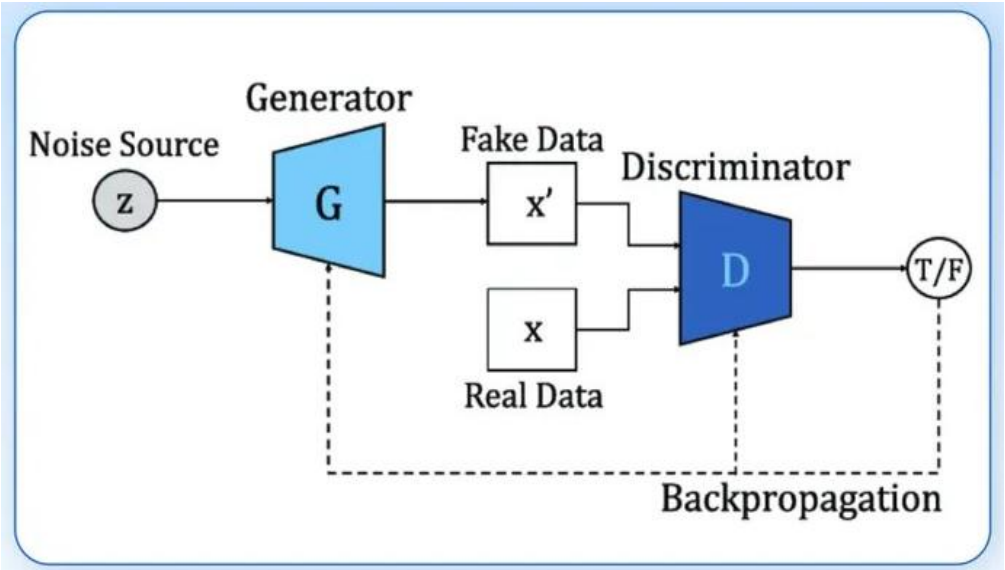


Figura 3.13: GAN (Generative Adversarial Networks)

L'architettura del CTGAN si basa sullo schema classico di una Generative Adversarial Network, composta da due reti neurali principali:

- un Generatore G, che impara a produrre dati sintetici realistici prendendo in input un vettore casuale $z \sim p(z)$ e genera un campione sintetico $\hat{x} = G(z \mid c)$
- un Discriminatore D, che cerca di distinguere tra dati reali e sintetici.

Queste due reti si sfidano in un gioco a somma zero, formalizzato come:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

Nel caso del CTGAN, viene introdotta una condizione che guida il generatore nella produzione di campioni appartenenti a specifiche categorie.

Questo è fondamentale per affrontare il problema dello sbilanciamento tra classi, ad esempio quando alcune classi di difetti sono scarsamente rappresentate. La condizione viene aggiunta sia all'input del generatore che a quello del discriminatore, estendendo la funzione obiettivo come segue:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x \mid c)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z \mid c)))]$$

Dove il generatore cerca di minimizzare:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim p(z)}[\log D(G(z \mid c) \mid c)]$$

Mentre il discriminatore cerca di massimizzare:

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{data}}[\log D(x \mid c)] - \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z \mid c) \mid c))]$$

dove:

- x è un dato reale,
- $G(z \mid c)$ è un dato sintetico generato dal generatore,
- $D(\cdot \mid c)$ è il discriminatore condizionato sulla categoria ccc,
- $p(z)$ è la distribuzione del rumore, tipicamente una distribuzione normale o uniforme.

3.3 Confronto dei dati sintetici generati

Entrambi i modelli sono progettati per generare dati sintetici realistici, anche in presenza di feature categoriche e numeriche eterogenee, come nel caso del dataset in esame. La scelta finale del generatore è stata effettuata confrontando i dati sintetici prodotti da ciascun modello con i dati reali, in termini di preservazione della struttura statistica originaria, con particolare attenzione alla matrice di correlazione tra le variabili. Tale analisi si è rivelata necessaria per valutare il rispetto delle distribuzioni originali delle feature numeriche e categoriche, come evidenziato dai grafici esplorativi, nonché il potenziale dei modelli nel compensare classi minoritarie, migliorando la rappresentatività di condizioni di taglio meno frequenti. Per avere dei

valori più robusti sovra-campioneremo rispetto alla quantità necessaria (circa 10 volte la dimensione della classe) per via della randomicità presente nelle generazioni. Nelle figure seguenti vengono riportate le matrici di correlazione ottenute a partire dai dati sintetici generati con ciascun modello. Il confronto visivo con la matrice calcolata sui dati originali consente di evidenziare quale tra i due approcci riesca a ricostruire meglio le relazioni presenti nel dataset reale e quindi di selezionare il modello più adatto per la successiva fase di analisi e modellazione.

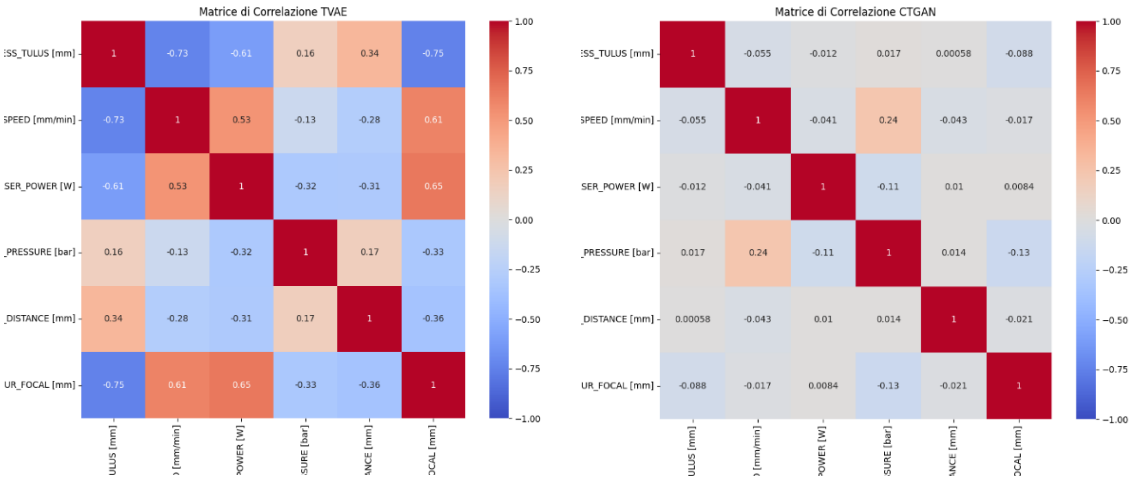


Figura 3.14: Matrici di correlazione dati sintetici per il difetto '*No defects*'

la matrice di correlazione del CTGAN mostra valori quasi nulli tra la maggior parte delle coppie di variabili, segnalando un'evidente perdita di struttura statistica. Il modello sembra aver generato dati quasi indipendenti tra loro, fallendo nella riproduzione delle relazioni lineari. Al contrario, nel TVAE le correlazioni tra variabili risultano coerenti e ben distribuite, con valori significativi sia positivi che negativi che riflettono relazioni realistiche tra i parametri del processo.

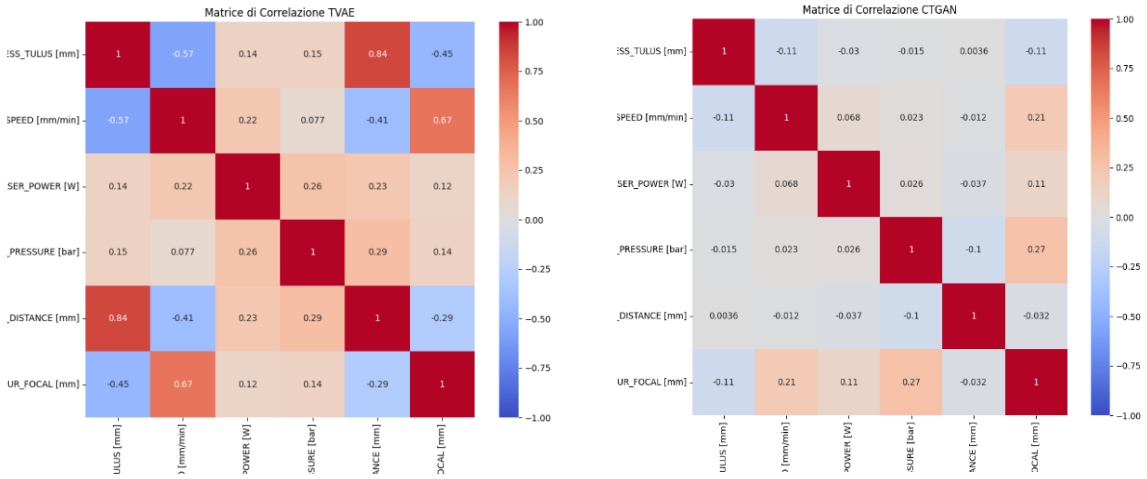


Figura 3.15: Matrici di correlazione dati sintetici per il difetto '*Burr*'

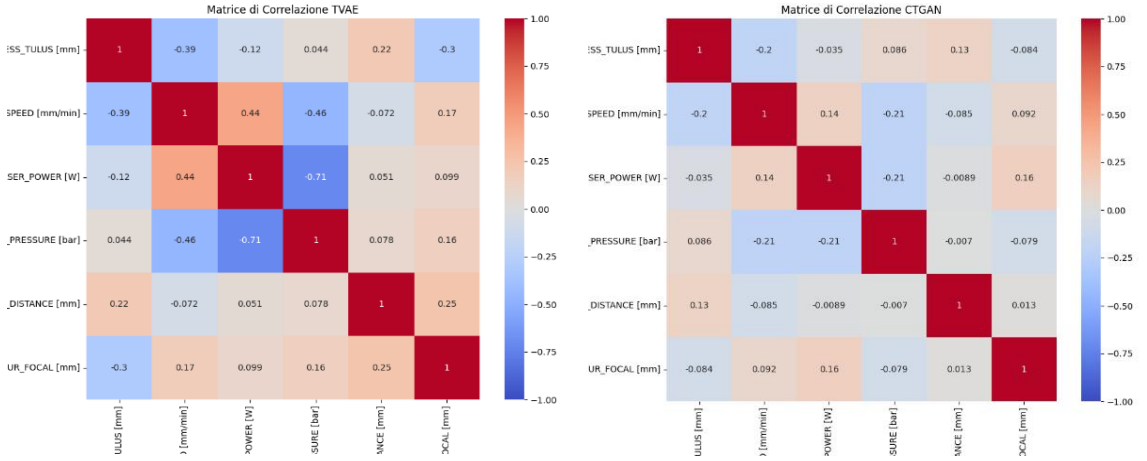


Figura 3.16: Matrici di correlazione dati sintetici per il difetto '*Plasma*'

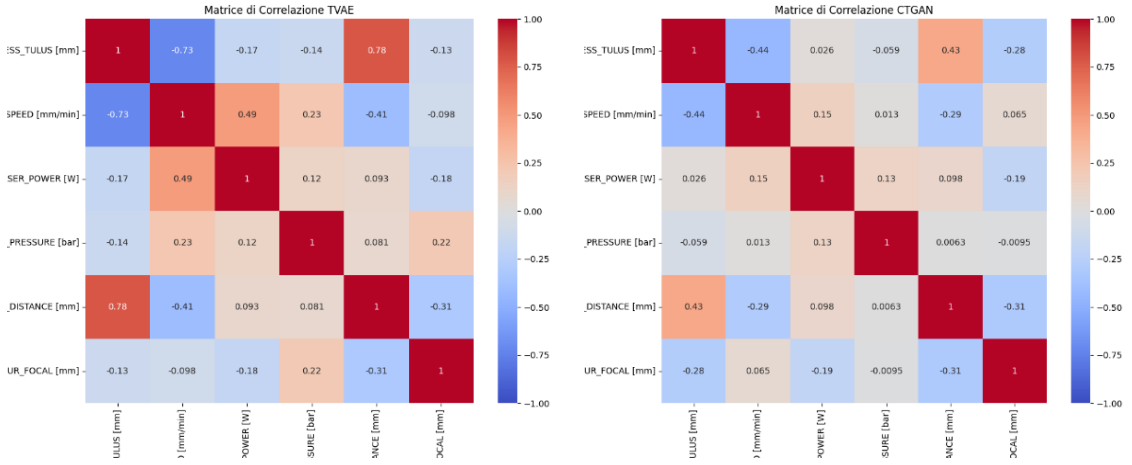


Figura 3.17: Matrici di correlazione dati sintetici per il difetto '*Cutting loss*'

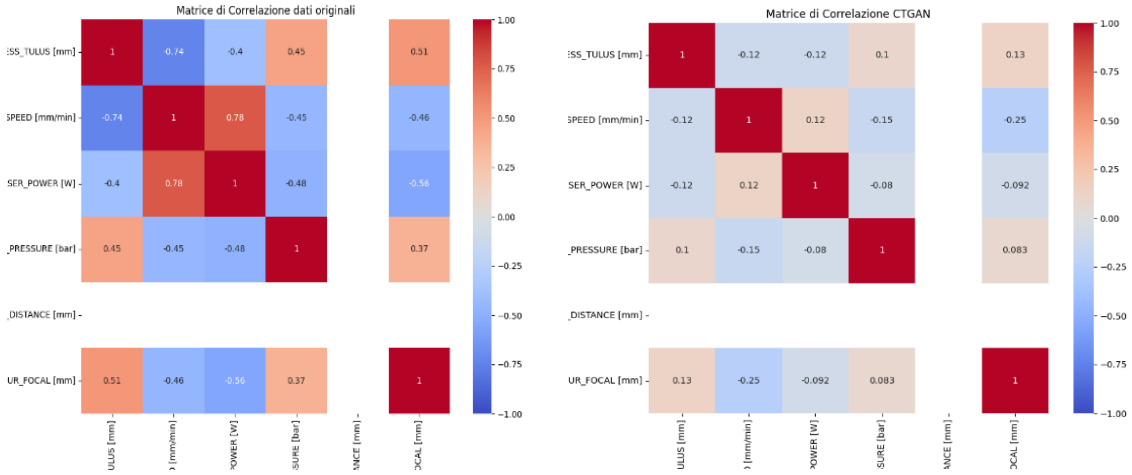


Figura 3.18: Matrici di correlazione dati sintetici per il difetto '*Cutting torn*'

Effettuando una valutazione grafica delle matrici di correlazione mostrate dalla Figura 3.14 alla figura 3.18 è possibile notare come il modello TVAE risulti essere molto più robusto e conservativo dal punto di vista della generazione di dati sintetici. In particolare, la matrice di correlazione ottenuta con il TVAE mostra una struttura di correlazioni che rispecchia in modo più fedele quella tipica dei dati reali, mantenendo la coerenza tra le variabili originali. Si osservano valori di correlazione più netti e ben distribuiti sia nelle relazioni positive che in quelle negative tra le diverse variabili, a testimonianza della capacità del TVAE di preservare la dipendenza statistica presente nei dati reali.

Ad esempio, si notano coefficienti di correlazione elevati tra alcune variabili chiave come $CONTOUR\_SPEED$ e $LASER\_POWER$ oppure tra $THICKNESS\_TULUS$ e $LASER\_POWER$, riflettendo pattern già individuati nell'analisi dei dati originali.
Al contrario, il modello CTGAN appare meno efficace nel riprodurre fedelmente la struttura di correlazione delle variabili.
Dalla matrice relativa, infatti, si rileva una maggiore tendenza all'appiattimento dei valori di correlazione, con molte relazioni che risultano attenuate o prossime allo zero. Questo comportamento suggerisce che il CTGAN, pur essendo in grado di generare dati sintetici realistici dal punto di vista delle distribuzioni univariate, fatica maggiormente a cogliere e preservare le interazioni multivariate tra le feature del dataset. In particolare, le relazioni tra $CONTOUR\_FOCAL$ e le altre variabili risultano molto deboli, così come le correlazioni tra $CONTOUR\_GAS\_PRESSURE$ e $CONTOUR\_SPEED$ che nei dati sintetici prodotti dal CTGAN appaiono meno rappresentate rispetto al modello TVAE. Complessivamente, l'analisi delle matrici di correlazione conferma la maggiore affidabilità del TVAE nel mantenere la struttura statistica del dataset originale. Questo aspetto è fondamentale ai fini di un data augmentation efficace, poiché garantisce che i dati sintetici non solo arricchiscano numericamente il dataset, ma contribuiscano anche a preservare le relazioni tra variabili necessarie per un'adeguata modellazione dei processi.
Tuttavia, per avere un'idea chiara e verificare che i modelli abbiano effettivamente preservato le caratteristiche delle singole variabili, è necessario effettuare un'analisi delle distribuzioni univariate dei dati sintetici. In seguito vengono mostrate le distribuzioni ottenute dai due modelli generativi.

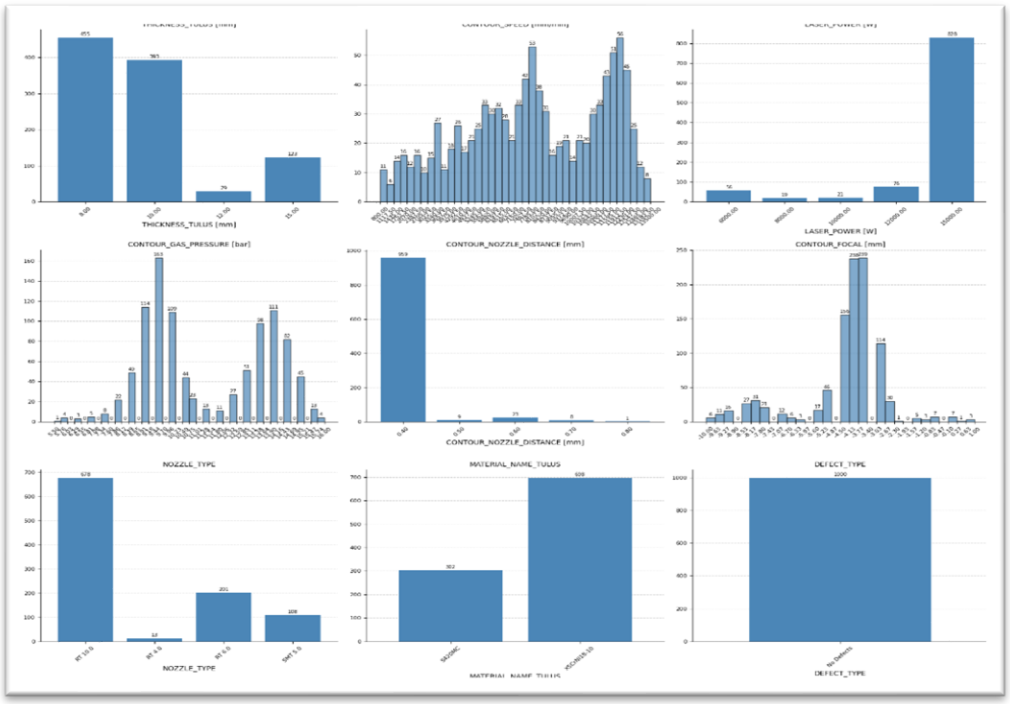


Figura 3.19: Distribuzioni variabili TVAE per il difetto '*No defects*'

Figura 3.20: Distribuzioni variabili CTGAN per il difetto '*No defects*'

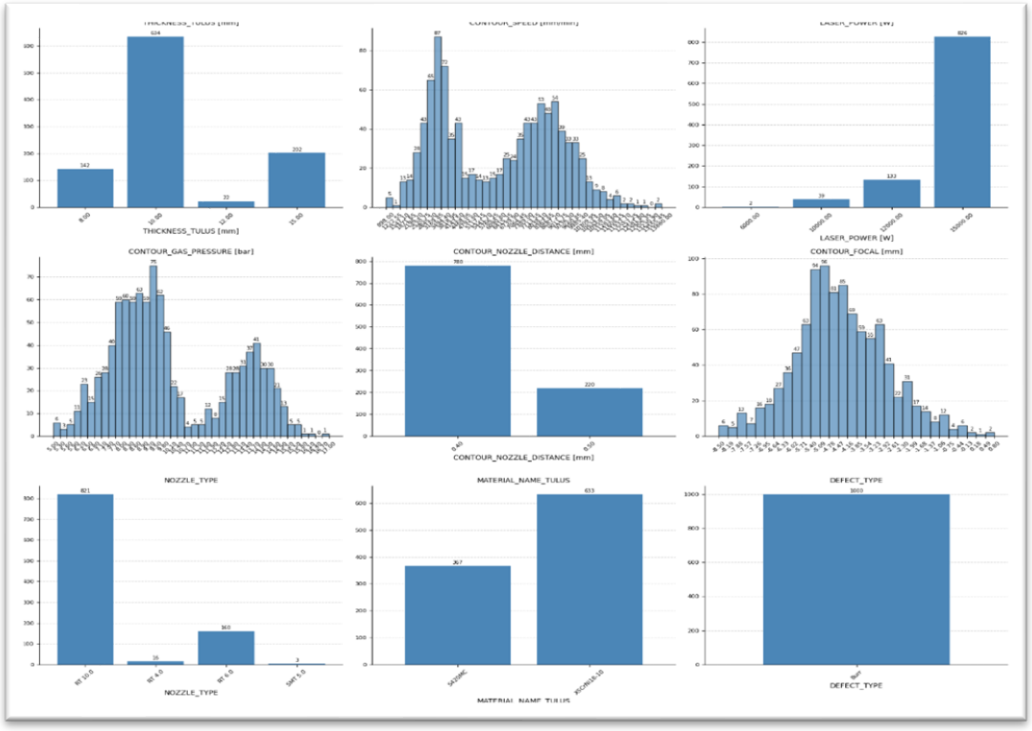


Figura 3.21: Distribuzioni variabili TVAE per il difetto '*Burr*'

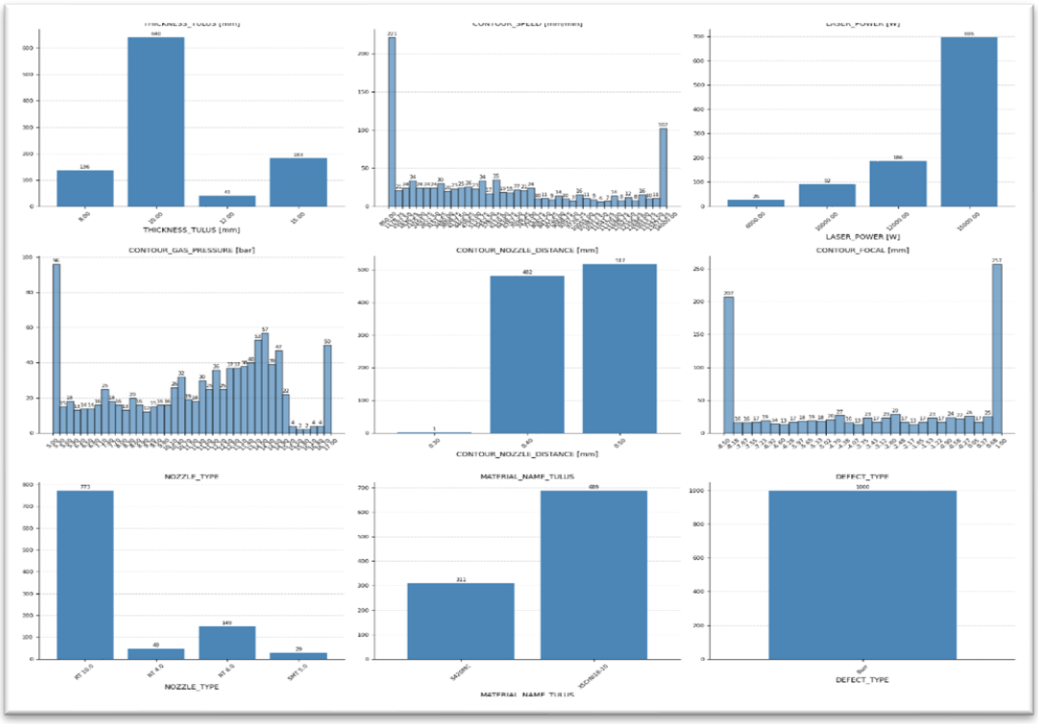


Figura 3.22: Distribuzioni variabili CTGAN per il difetto '*Burr*'

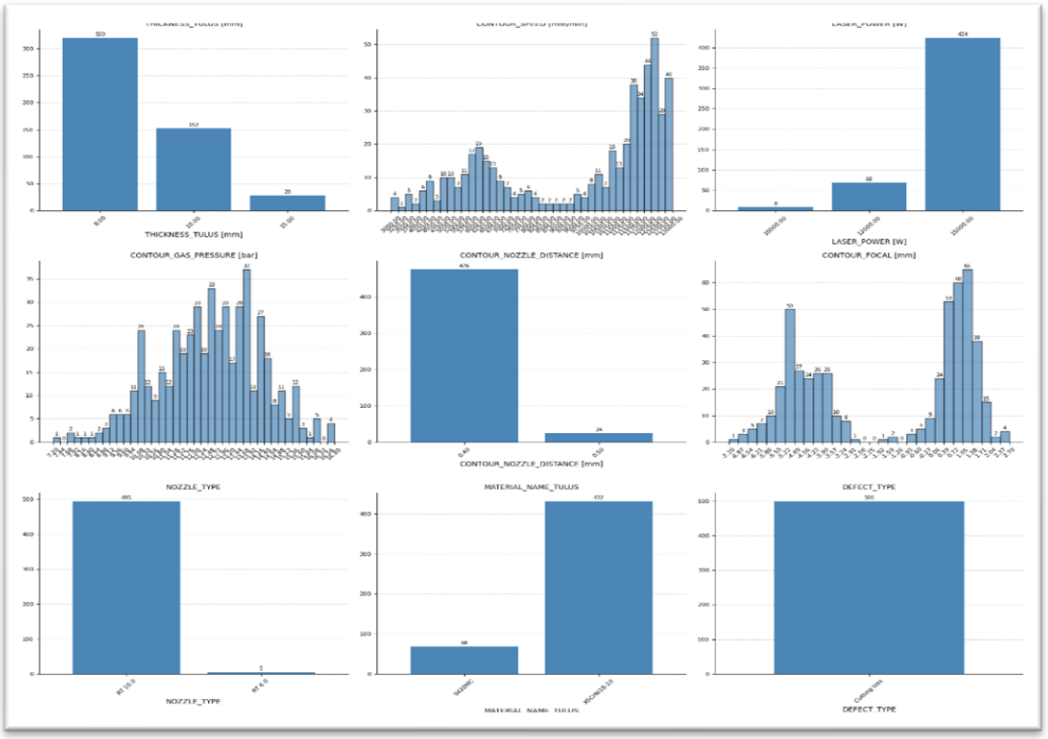


Figura 3.23: Distribuzioni variabili CTGAN per il difetto '*Cutting loss*'

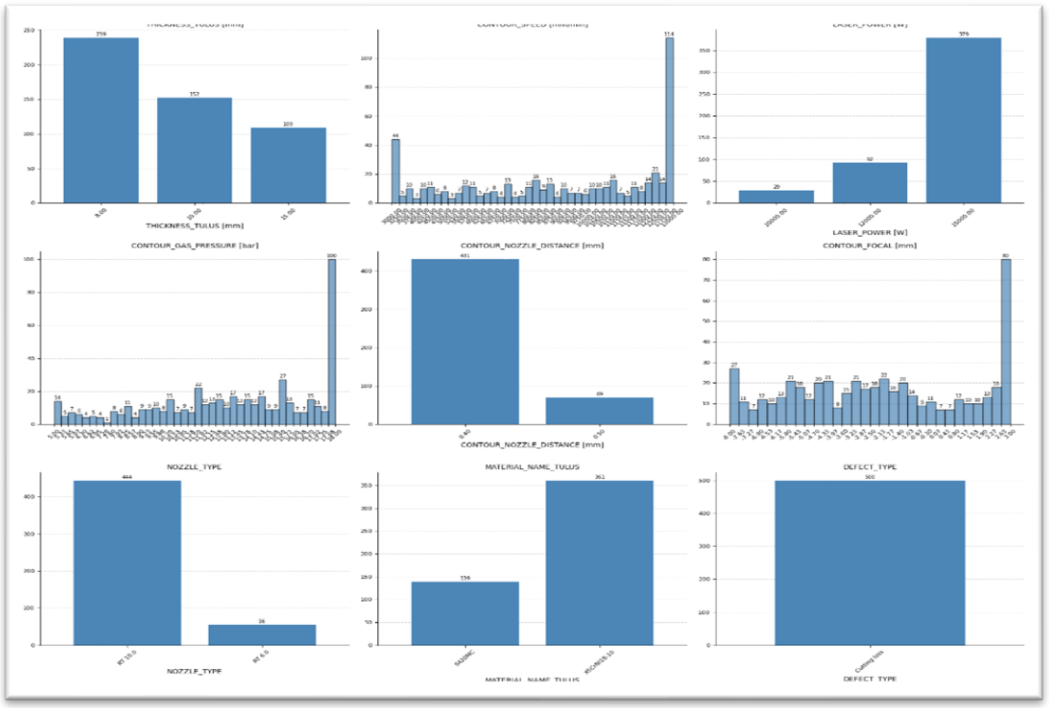


Figura 3.24: Distribuzioni variabili CTGAN per il difetto '*Cutting loss*'

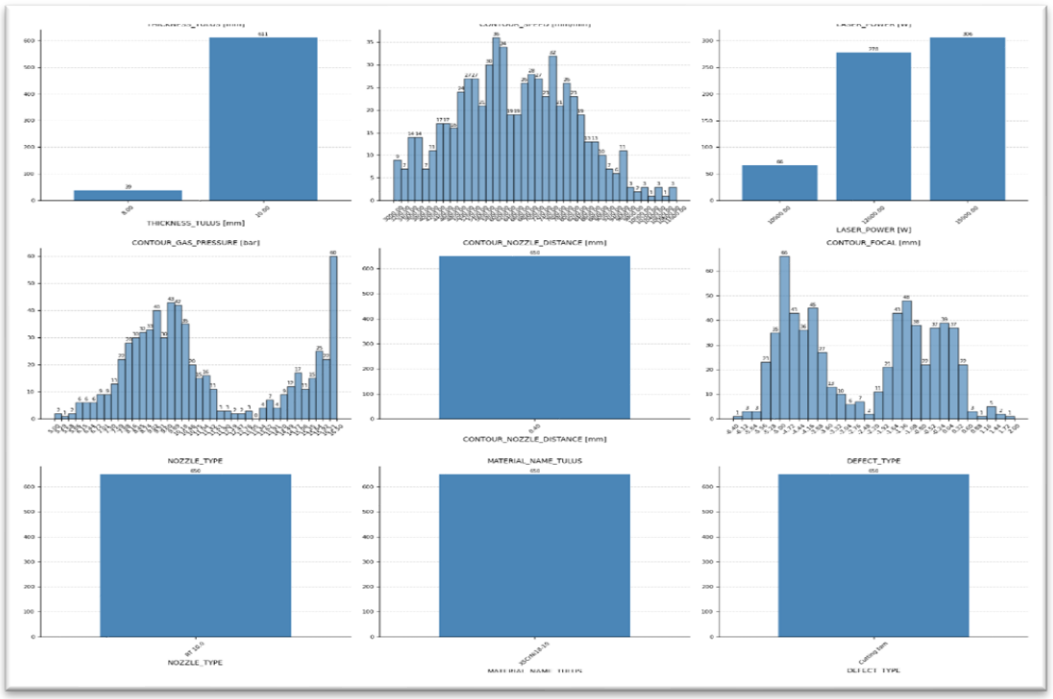


Figura 3.25: Distribuzioni variabili TVAE per il difetto '*Cutting torn*'

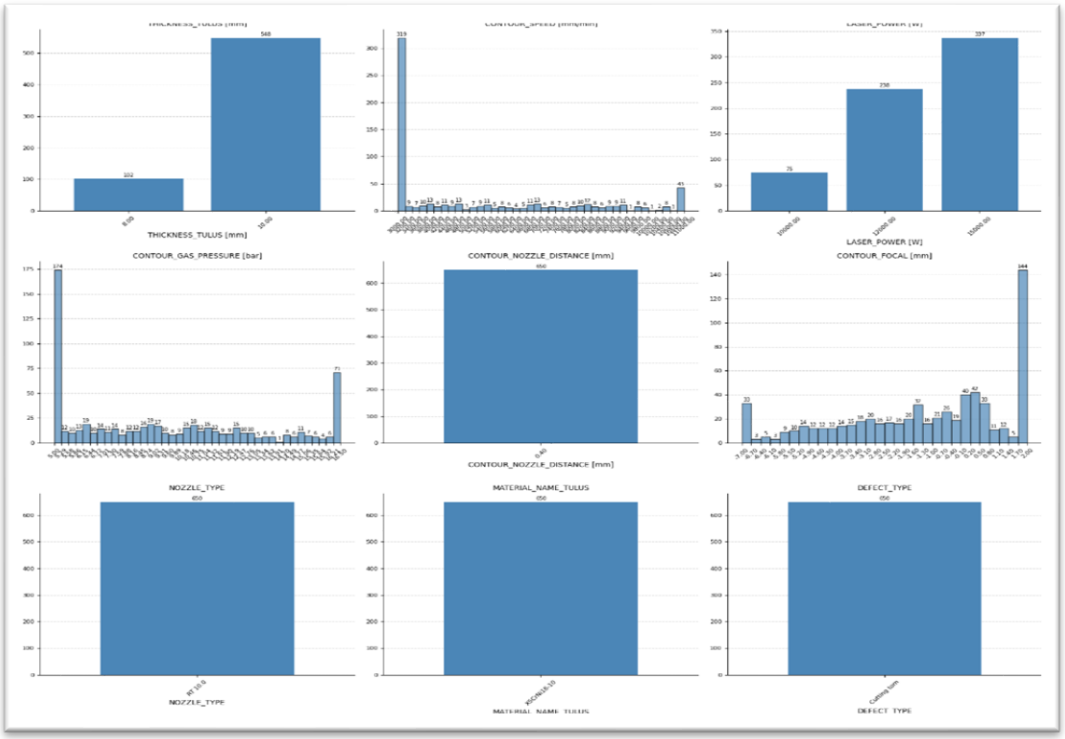


Figura 3.26: Distribuzioni variabili CTGAN per il difetto '*Cutting torn*'

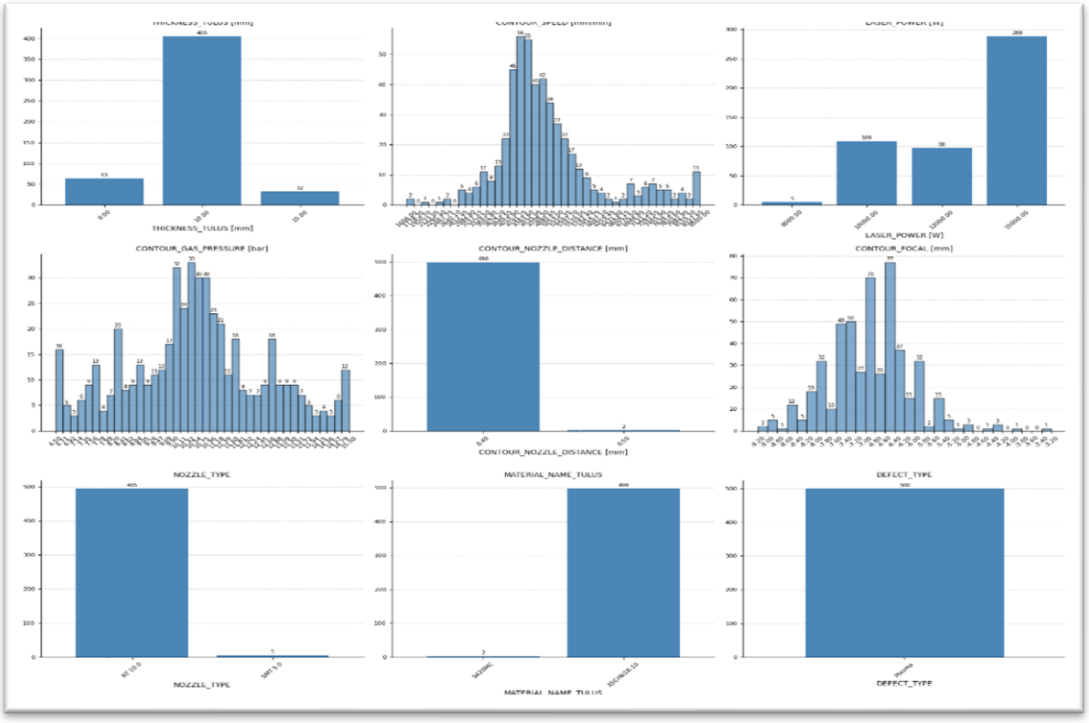


Figura 3.27: Distribuzioni variabili TVAE per il difetto '*Plasma*'

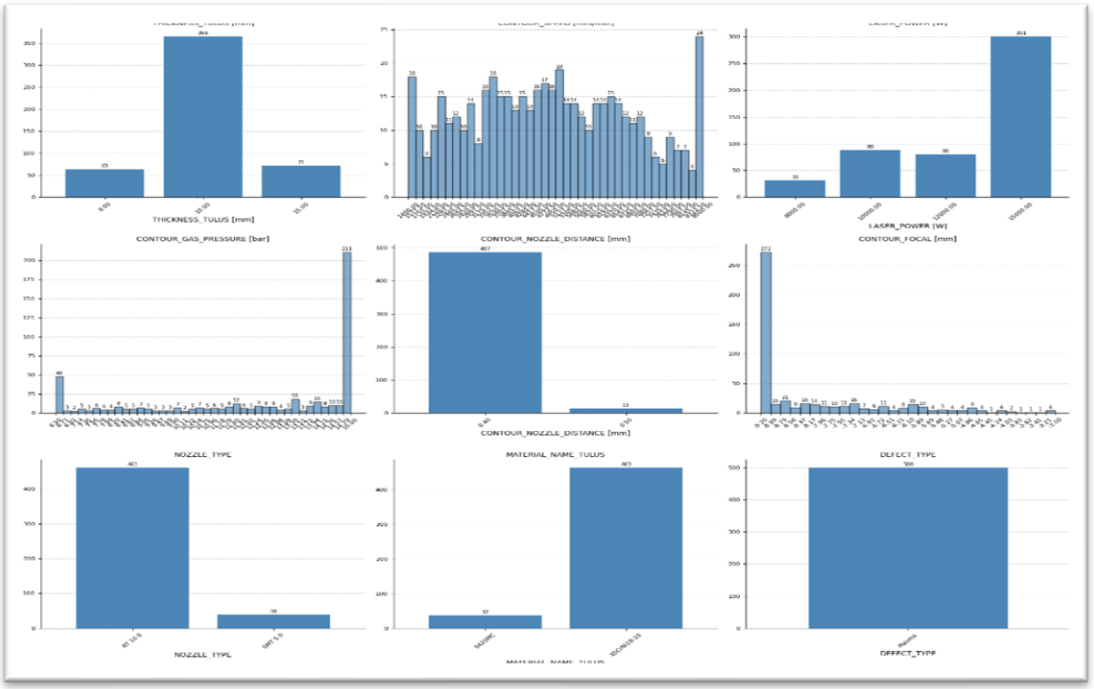


Figura 3.28: Distribuzioni variabili CTGAN per il difetto '*Plasma*'

Oltre a quanto già osservato, un'analisi più dettagliata dei grafici di Figura 3.19 permette di evidenziare ulteriori differenze qualitative tra i due modelli generativi. In particolare, il TVAE si distingue per la capacità di riprodurre fedelmente non solo la forma delle distribuzioni delle variabili continue come la simmetria, la presenza di code, i picchi e i valori modali, ma anche la presenza di eventuali multimodalità o irregolarità tipiche dei dati reali. Questo aspetto è particolarmente importante in ottica di data augmentation, perché garantisce che i dati sintetici non introducano distorsioni nelle relazioni tra le variabili, mantenendo quindi la validità statistica del dataset ampliato. Nel caso del CTGAN, invece, si nota una tendenza a una certa "piattezza" nelle distribuzioni delle variabili continue e una minore capacità di cogliere le caratteristiche più fini dei dati reali, come la presenza di sottopopolazioni o l'asimmetria. Tuttavia, come già accennato, il punto di forza di questo modello risiede nella gestione delle variabili categoriche, dove il condizionamento sull'input consente di garantire una maggiore rappresentatività delle classi meno frequenti. Questo si traduce in una migliore copertura delle categorie rare, il che può essere particolarmente utile per mitigare problemi di sbilanciamento e migliorare la performance dei modelli predittivi sulle classi minoritarie.
Un ulteriore aspetto da sottolineare riguarda la coerenza tra la struttura delle distribuzioni e le matrici di correlazione analizzate in precedenza: i risultati ottenuti confermano che il TVAE è in grado di preservare sia le correlazioni tra le variabili che le statistiche univariate, mentre il CTGAN, pur mostrando buone performance sulle variabili categoriche, potrebbe non essere altrettanto efficace nel mantenere le dipendenze tra le variabili continue. Alla luce di questi risultati, il TVAE si configura come la scelta preferenziale per la generazione di nuovi campioni sintetici nel contesto specifico analizzato, risultando particolarmente adatto a supportare le successive fasi dell'implementazione.

3.4 Sottocampionamento dei dati sintetici mediante greedy subsampling

Nel nostro workflow, per ciascuna delle classi di difetto considerate, è stato generato un ampio set di dati sintetici utilizzando il modello TVAE, addestrato separatamente su ciascun difetto. Tuttavia, l'intero insieme di dati sintetici prodotti per ogni difetto risulta in genere molto più grande rispetto al corrispondente dataset reale e presenta una significativa ridondanza. Inoltre, includere indiscriminatamente tutti i campioni sintetici potrebbe introdurre distorsioni nelle distribuzioni e nelle correlazioni tra feature, compromettendo la validità statistica del dataset complessivo.

Per affrontare questo problema, è stato implementato un meccanismo di sottocampionamento guidato con l'obiettivo di selezionare, per ogni difetto, un sottoinsieme dei dati sintetici che riproduca il più fedelmente possibile la struttura di correlazione osservata nei dati reali. In particolare, ci si è focalizzati sulle feature continue, calcolando per ciascun set la relativa matrice di correlazione e calcolando la distanza di Frobenius.

3.4.1Distanza di Frobenius

la distanza di Frobenius è una misura largamente adottata per valutare la distanza tra due matrici. Data una matrice $A \in R^{m \times n}$, la norma di Frobenius si definisce come:

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$$

In altre parole, essa rappresenta la radice quadrata della somma dei quadrati di tutti gli elementi della matrice. Quando si vuole quantificare quanto due matrici $\boldsymbol{A}$ e $\boldsymbol{B}$ siano simili, si considera la norma di Frobenius della loro differenza:

$$\|A - B\|_F$$

Una distanza Frobenius più bassa indica una maggiore somiglianza tra le strutture di correlazione. Questa metrica è particolarmente utile perché è facile da calcolare, invariante per trasposizione e sensibile alle variazioni globali nella matrice, risultando quindi efficace come criterio di confronto per selezionare il miglior sottoinsieme sintetico da mantenere nel processo di generazione.

3.4.2 Greedy Selection basata su distanza di Frobenius

La procedura adottata si articola in due fasi principali:

- Nella prima fase, denominata Tournament Selection, vengono estratti casualmente più campioni sintetici della stessa dimensione del dataset reale. Per ciascun campione si calcola la distanza di Frobenius tra la matrice di correlazione delle feature continue e quella calcolata sui dati reali. Viene selezionato il sottoinsieme di campioni con la distanza di Frobenius minima come base iniziale del sottoinsieme sintetico.

- Nella seconda fase, l'algoritmo **greedy** espande progressivamente questo sottoinsieme iniziale fino ad arrivare alla dimensione desiderata. A ogni iterazione, viene aggiunto il campione sintetico che comporta la minore variazione (o la massima riduzione) della distanza di Frobenius rispetto alla matrice di correlazione reale.

Questo metodo consente di selezionare, per ciascuna classe di difetto, un sottoinsieme di dati sintetici statisticamente più simile ai dati reali, pur mantenendo una certa varietà nei campioni. I risultati di questa procedura sono stati poi analizzati sia visivamente (mediante pairplot e confronto diretto delle matrici di correlazione) sia quantitativamente, al fine di verificarne l'efficacia nella conservazione della struttura multivariata dei dati reali. In seguito viene mostrato un coinfronto tra la matrice di correlazione associata ai dati reali (destra) con quella ottenuta tramite l'operazione di sottocampionamento con l'algoritmo greedy (sinistra):

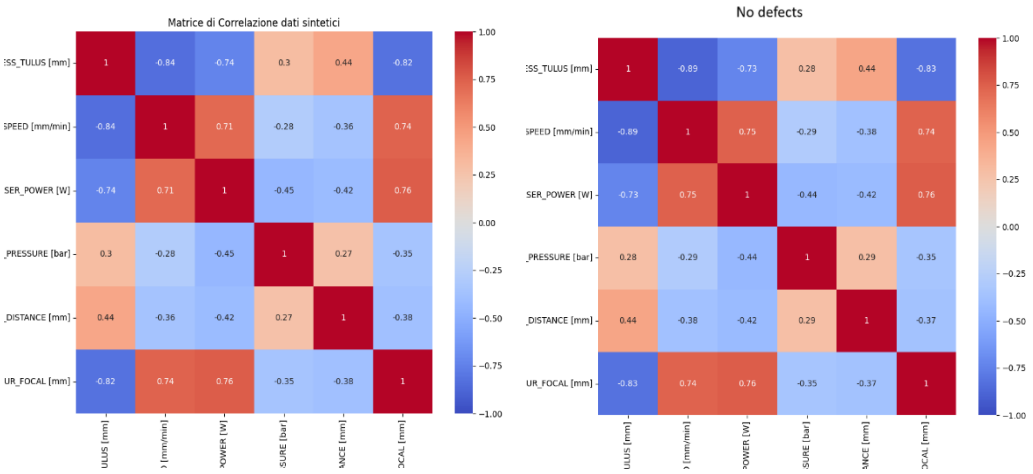


Figura 3.29: Confronto delle matrici di correlazione per il difetto '*No defects*'

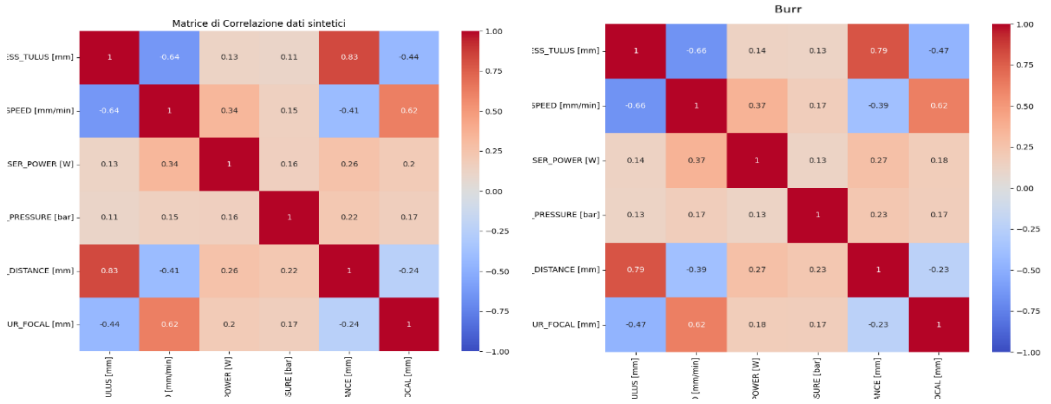


Figura 3.30: Confronto delle matrici di correlazione per il difetto '*Burr*'

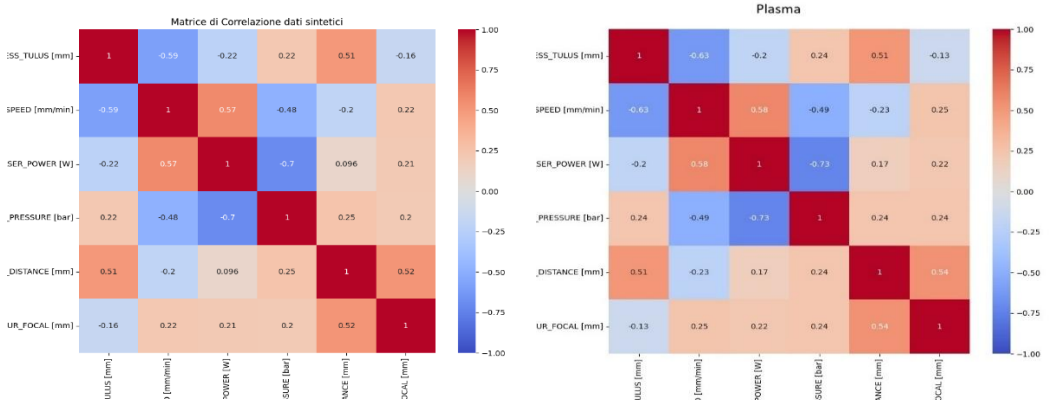


Figura 3.31: Confronto delle matrici di correlazione per il difetto '*Plasma*'

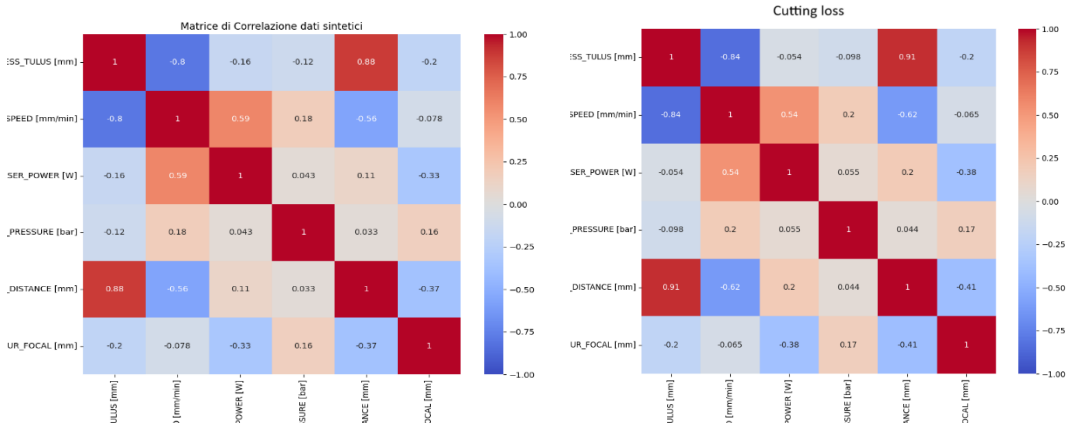


Figura 3.32: Confronto delle matrici di correlazione per il difetto '*Cutting loss*'

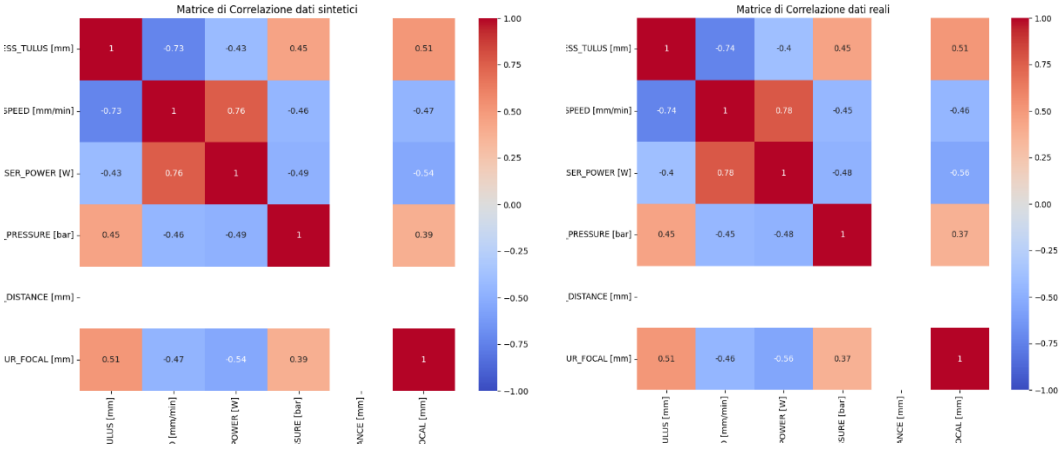


Figura 3.33: Confronto delle matrici di correlazione per il difetto '*Cutting torn*'

Il confronto tra le matrici di correlazione calcolate sui dati reali e quelle ottenute dopo il subsampling, riportato dalla figura 3.24 alla figura 3.28 evidenzia una notevole somiglianza tra le strutture di correlazione tra i dati reali e i dati generati post pulizia (le righe/colonne vuote sono associate a valori costanti della variabile). In particolare, si osserva come i pattern di relazione tra le diverse variabili vengano fedelmente riprodotti nei dati sintetici, sia in termini di intensità che di segno delle correlazioni. Il risultato è particolarmente significativo poiché conferma la capacità del TVAE non solo di generare dati sintetici numericamente simili agli originali, ma anche di preservare la struttura multivariata complessa che caratterizza il dataset reale, il quale rappresenta un aspetto fondamentale per garantire la validità statistica e la rappresentatività dei dati aumentati nelle successive fasi di analisi e modellazione.

3.4.3 Confronto distribuzioni bivariate

Ora confronteremo le distribuzioni bivariate tra i dati reali (destra) e i dati generati (sinistra) per vedere se le tendenze delle relazioni osservate nel capitolo 3.1.3 sono stati mantenuti:

Figura 3.34: Confronto degli scatter plot per il difetto '$No\ defects$'

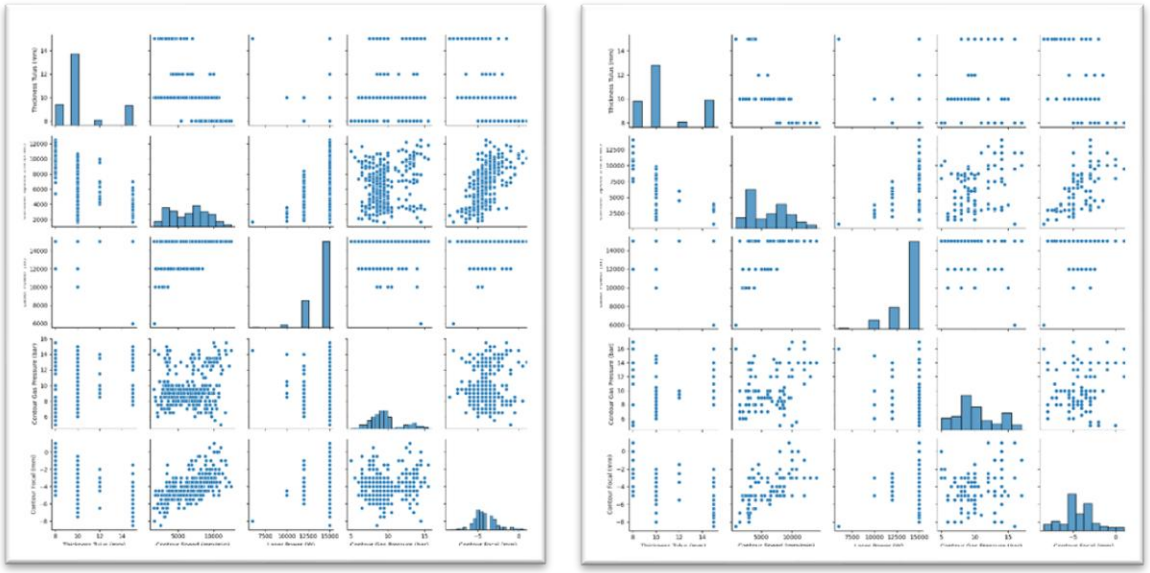


Figura 3.35: Confronto degli scatter plot per il difetto '$Burr$'

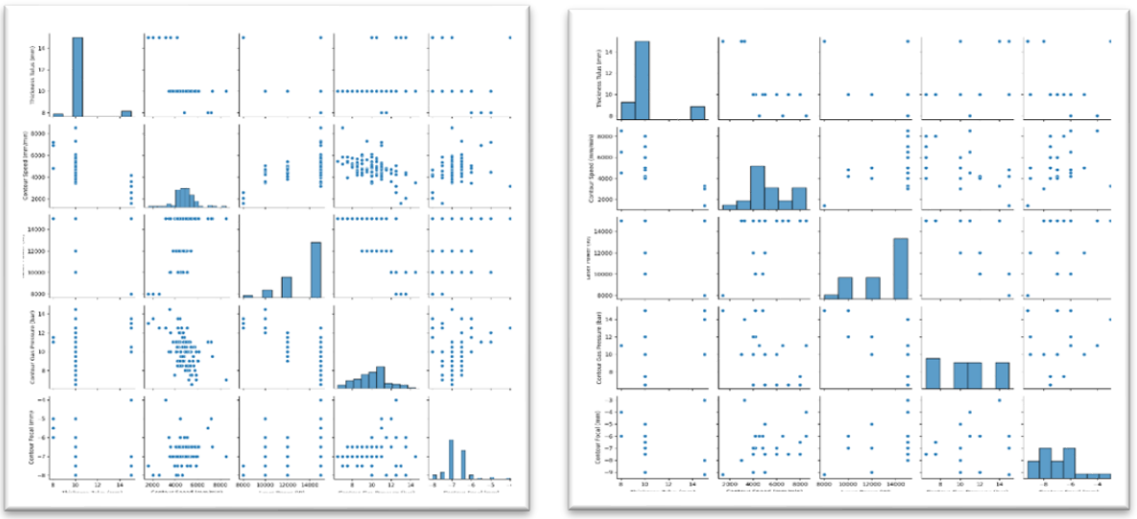


Figura 3.36: Confronto degli scatter plot per il difetto '*Plasma*'

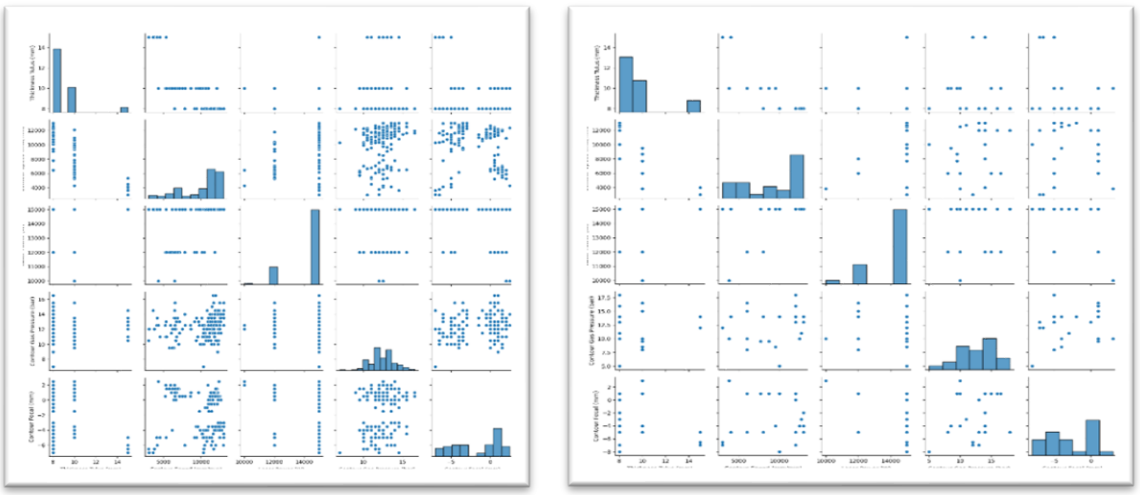


Figura 3.37: Confronto degli scatter plot per il difetto '*Cutting loss*'

La generazione e il subsampling dei dati sintetici hanno permesso di ampliare sensibilmente la numerosità dei dati associati a ciascun difetto, fino a raggiungere un incremento di circa sei volte rispetto alla situazione iniziale. Questo risultato rappresenta un importante supporto nella gestione dell'asimmetria e della scarsità di dati, in particolare per le classi meno rappresentate.

Difetti / Campioni	No defect	Burr	Plasma	Cutting loss	Cutting torn
Campioni iniziali	89	115	25	25	45
Campioni attuali	356	460	125	175	225

Figura 3.38: Dataset post generazione sintetica

Tuttavia, tale operazione comporta anche dei rischi potenziali: l'aumento della densità dei dati, se non accompagnato da una fedele riproduzione delle relazioni originarie, può introdurre correlazioni spurie o alterare la struttura del dataset. L'analisi visiva degli scatter plot conferma che, nel complesso, le principali relazioni tra le variabili sono state mantenute anche dopo l'augmentation: le nuvole di punti risultano più dense e le strutture già presenti nei dati reali appaiono rafforzate, a testimonianza della buona qualità dei dati sintetici generati. La maggiore densità facilita l'identificazione di pattern e tendenze già latenti nel dataset originale, migliorando la robustezza delle successive analisi statistiche e modellistiche. Tuttavia, si rileva anche la comparsa di alcuni outlier, un fenomeno fisiologico nei processi di generazione sintetica, specialmente quando si cerca di ampliare fortemente lo spazio delle condizioni sperimentalia presenza di outlier richiede particolare attenzione nelle fasi successive, dove verranno adottati specifici algoritmi di rilevamento degli outlier al fine di effettuare una pulizia dei dati. Questo passaggio risulta fondamentale per evitare che tali anomalie possano influire negativamente sulle performance dei modelli, sulla capacità di generalizzazione degli algoritmi di correzione dei parametri e sull'efficacia del classificatore che verrà discusso nei capitoli successivi.

3.5 Identificazione e rimozione delle anomalie

A seguito della generazione e dell'arricchimento del dataset tramite dati sintetici, un passaggio fondamentale per garantire la qualità e l'affidabilità dei dati è rappresentato dall'identificazione e dalla rimozione delle anomalie, comunemente noti come outlier. La presenza di outlier, infatti, può derivare sia da errori sperimentali nei dati reali, sia da imperfezioni nei modelli generativi che, durante il processo di data augmentation, possono occasionalmente produrre campioni non pienamente coerenti con la distribuzione originaria delle variabili. Questi punti anomali rischiano di introdurre rumore e distorsioni nelle fasi successive di analisi, influenzando

negativamente sia le performance dei modelli di regressione e classificazione, sia la loro capacità di generalizzare a nuovi dati.
Per affrontare questa problematica, vengono applicati algoritmi di rilevamento degli outlier, come l'Isolation Forest, il Local Outlier Factor (LOF) o tecniche basate su clustering come l'HDBSCAN , in grado di individuare i campioni che si discostano significativamente dalla struttura statistica del dataset. Tali metodi operano sfruttando la natura multidimensionale dei dati, valutando simultaneamente tutte le variabili disponibili per identificare osservazioni potenzialmente anomale.

3.5.1 Isolation Forest

L'Isolation Forest introdotto da Liu et al. nel 2008 [12], è un algoritmo di anomaly detection non supervisionato, sviluppato per identificare efficacemente outlier all'interno di grandi dataset, anche ad alta dimensionalità. L'idea fondamentale che lo caratterizza è che le anomalie, essendo rare e distanti dalla maggior parte dei dati, risultano più semplici da isolare rispetto ai punti normali. L'algoritmo costruisce una foresta di alberi binari ("isolation trees") in cui, a ogni nodo, una feature viene scelta casualmente e viene effettuato uno split su un valore casuale compreso nel range osservato. Questo processo si ripete ricorsivamente, fino a isolare ogni punto in una foglia dell'albero.
Il principio alla base è che un outlier richiede, mediamente, un numero di suddivisioni minore (profondità più bassa) per essere isolato rispetto ai punti normali.

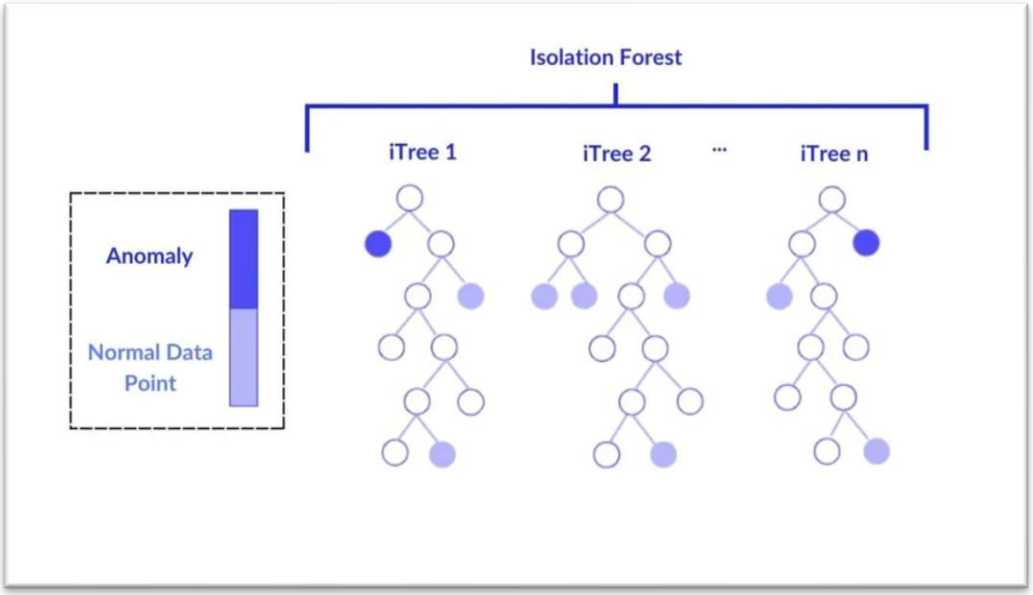


Figura 3.39: Isolation Forest

La profondità media necessaria per isolare un punto viene quindi utilizzata per calcolare il suo anomaly score, secondo la seguente formula:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

dove:

- $s(x,n)$ rappresenta l'anomaly score del punto x in un dataset di n campioni,
- E(h(x)) rappresenta la lunghezza media del percorso (numero di nodi attraversati fino all'isolamento) per x, calcolata mediando su tutti gli alberi della foresta (detti estimators),
- c(n) è il valore atteso della lunghezza del percorso in un albero binario, approssimato da:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n}$$

  dove H(i) è l'armonica di x ($H(x) \approx ln(i) + 0.5772$, costante di Eulero-Mascheroni).

Valori dello score s(x,n) maggiori di 0.5, secondo la teoria, implicano che il punto in questione sia da considerarsi un outlier, altrimenti è un punto normale. Tuttavia, nella pratica, la soglia di 0.5 può non essere ottimale per tutti i dataset, soprattutto quando la proporzione di outlier attesa è diversa da quella presupposta teoricamente. Per semplificare e rendere più flessibile l'identificazione degli outlier, l'implementazione pratica prevede l'inserimento di un parametro $contamination$ compreso tra 0 e 0.5, che rappresenta l'aspettativa sulla presenza di outlier nei dati, ad esempio, ottenuta tramite osservazione grafica o conoscenza pregressa del dominio.
Più precisamente, dopo il calcolo dei punteggi di anomalia (anomaly scores) per ciascun campione, l'algoritmo ordina i dati in base a questi valori e considera come outlier la frazione di punti corrispondente al valore di contamination impostato, stabilendo così una soglia operativa che può discostarsi dal valore teorico di 0.5 e adattarsi meglio al contesto specifico dell'analisi.

3.5.2 HDBSCAN

HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [14] è un algoritmo di clustering basato sulla densità, particolarmente efficace in presenza di dati non sferici, distribuzioni eterogenee e rumore. A differenza del classico DBSCAN [13], che richiede la definizione di un singolo parametro di distanza (ε), HDBSCAN costruisce una struttura gerarchica dei cluster, determinando automaticamente il numero e la forma dei gruppi, e isolando gli outlier come punti che non appartengono a nessun cluster significativo. L'algoritmo si basa su una metrica detta distanza di raggiungibilità reciproca (mutual reachability distance), che estende la distanza euclidea includendo informazioni sulla densità locale. Per ogni punto x, si definisce la distanza core come:

$$\text{core\_dist}_k(x) = \text{distanza tra } x \text{ e il suo } k\text{-esimo vicino più vicino}$$

Dove la metrica di distanza dipende dalla complessità del problema e dal caso specifico, come ad esempio la presenza di variabili eterogenee.
Tra le più utilizzate abbiamo:

- Euclidea
- Manhattan
- Mahalanobis
- Coseno

Il concetto chiave su cui si fonda HDBSCAN è la mutual reachability distance, una metrica che estende la distanza ordinaria (Euclidea, Manhattan, ecc.) integrando informazioni sulla densità locale. Dato un valore di k (numero di vicini considerati), la distanza di mutual reachability tra due punti a e b è definita come:

$$d_{\text{mreach}}(a, b) = \max\left(\text{core}_k(a), \text{core}_k(b), d(a, b)\right)$$

dove:

- $d(a, b)$ è la distanza tra i punti a e b secondo la metrica scelta
- $core_k(a)$ è la core distance del punto a, ovvero la distanza tra a e il suo k-esimo vicino più prossimo.

Questa metrica garantisce che la distanza tra due punti rifletta anche la difficoltà di raggiungersi in aree meno dense. In particolare, essa penalizza i collegamenti tra punti in regioni sparse, migliorando la robustezza del clustering e la capacità di rilevare outlier. Infatti, HDBSCAN classifica automaticamente come outlier quei punti che non risultano appartenere ad alcun cluster stabile nella gerarchia. Ogni punto riceve un outlier score, basato sulla sua persistenza nei cluster durante il processo di creazione della struttura gerarchica. Più un punto è isolato, maggiore sarà il suo outlier score. Questi punteggi possono essere utilizzati per selezionare soglie e identificare anomalie in modo flessibile.

$$\text{outlier\_score}(x) = 1 - \lambda(x)$$

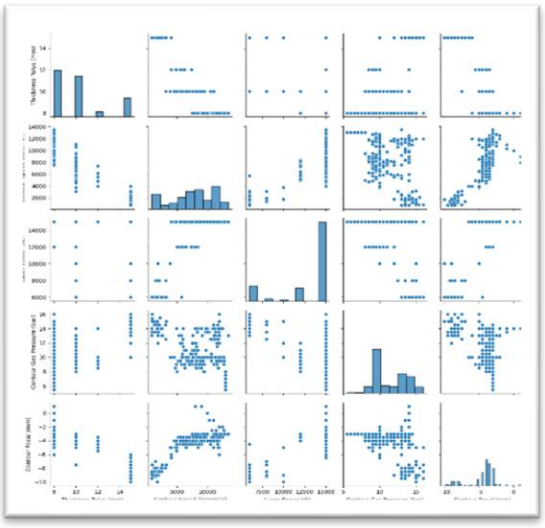
dove $\lambda(x)$ rappresenta la densità alla quale il punto si unisce definitivamente a un cluster. Valori vicini a 1 indicano alta probabilità che il punto sia un'anomalia.

3.5.3 Rimozione outliers con Isolation Forest

Per il processo di eliminazione degli outlier è stato impostato un livello di contaminazione proporzionale alla numerosità di ciascuna classe di difetto, con l'obiettivo di rimuovere un numero maggiore di punti dalle classi più dense.Questo approccio mira a riequilibrare la distribuzione delle classi, rendendole più comparabili tra loro.

L'eliminazione è stata effettuata all'interno di gruppi omogenei di dati, definiti sulla base della combinazione di tre attributi: materiale spessore e tipo di ugello. Questa scelta è stata motivata dalla necessità di preservare la rappresentatività di categorie rare: una segmentazione meno granulare avrebbe infatti potuto causare la rimozione eccessiva di dati provenienti da combinazioni poco frequenti, compromettendo la varietà del dataset. A supporto di questo processo, nelle sezioni seguenti sono riportati gli scatter plot dell'intero insieme di dati, comprensivo sia dei campioni reali che di quelli generati. Queste visualizzazioni evidenziano l'effetto dell'eliminazione degli outlier all'interno dei diversi sottogruppi. Rispetto ai grafici presentati nella fase preliminare di analisi esplorativa, quelli ottenuti dopo la generazione e pulizia risultano più densi e strutturati.

No defects



Burr

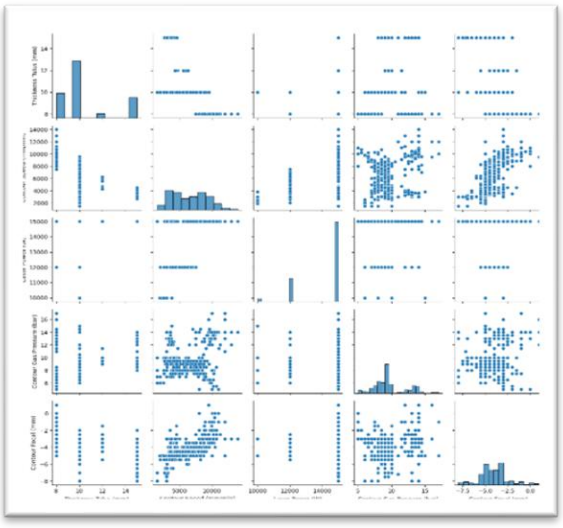
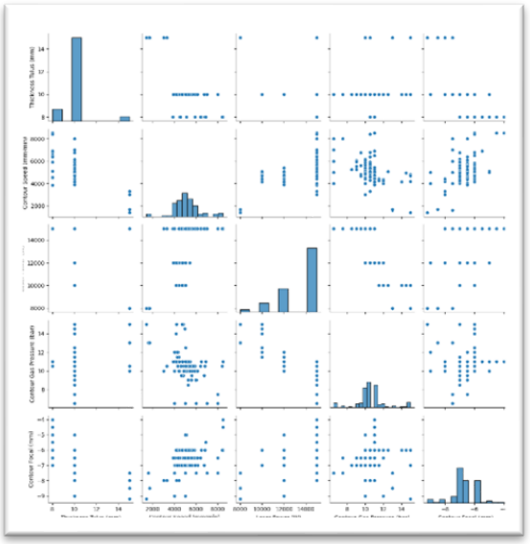


Figura 3.40: Distribuzione bivariata dei dati completi per ‘No Defect’ e ‘Burr’

Plasma



Cutting torn

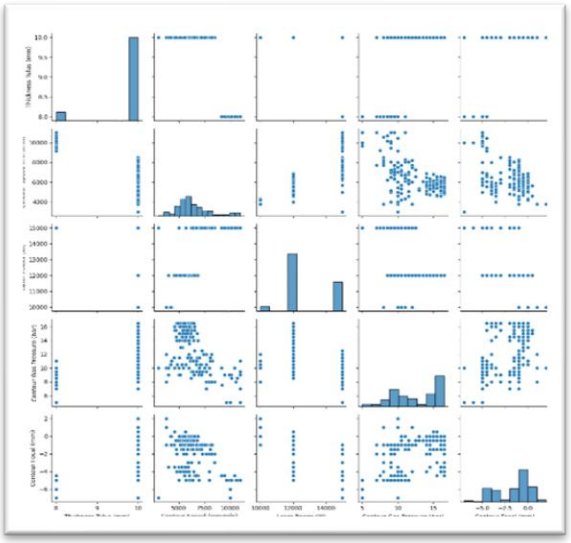


Figura 3.41: Distribuzione bivariata dei dati completi per ‘Plasma e ‘Cutting torn’

Cutting loss

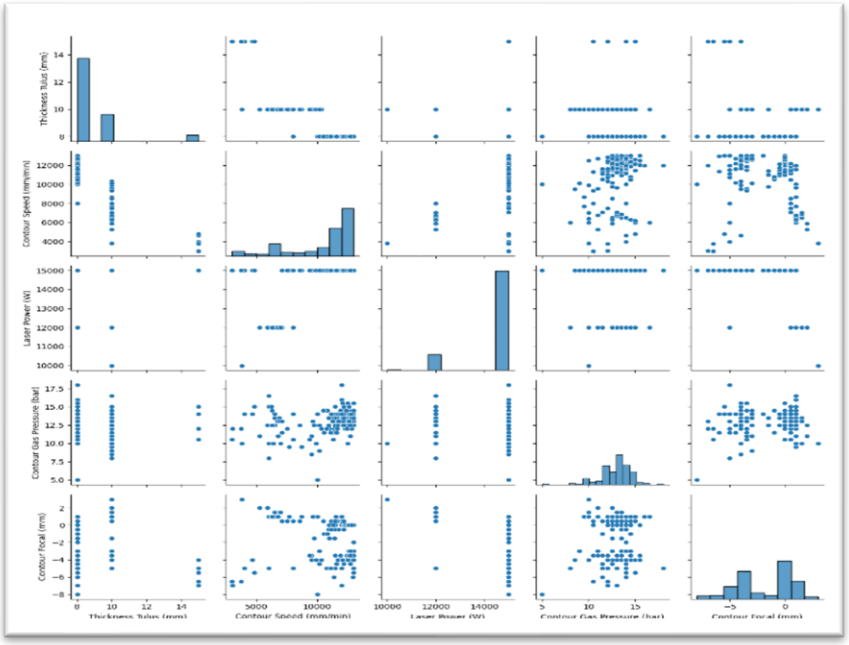


Figura 3.42: Distribuzione bivariata dei dati completi per 'Cutting loss'

3.6 Pipeline per la correzione parametri di taglio

Dopo l'attenta fase di ottimizzazione dei dati, volta a popolare il dataset e a rafforzare le relazioni tra le variabili, si procede con la fase finale: la correzione automatica dei parametri di taglio. A tal fine è stato addestrato un modello di classificazione dei difetti, capace di riconoscere il tipo di difetto (o se semplicemente classifica se la configurazione di taglio è buona o cattiva) a partire dalla configurazione dei parametri di processo. Il problema si presenta complesso per via di due aspetti principali: da un lato, la distribuzione fortemente sbilanciata delle classi; dall'altro, la natura categorica non ordinale della variabile target, che rende difficile modellare il legame tra input e output mediante approcci parametrici tradizionali. Inoltre, la disponibilità limitata di dati impedisce l'applicazione di tecniche statistiche sistematiche, come l'ANOVA, i test di sensibilità o le regressioni multiple, che richiederebbero una maggiore copertura e varietà nei campioni sperimentali.

In questo contesto, l'uso di classificatori basati su algoritmi di machine learning risulta particolarmente efficace: questi modelli apprendono direttamente dai dati la relazione (potenzialmente non lineare e complessa) che lega i parametri di processo

al tipo di difetto osservato, inclusa la distinzione tra condizioni di taglio corrette e difettose. Il classificatore assume così un ruolo di supporto al modello di regressione, il cui obiettivo è proporre configurazioni alternative che correggano il difetto riscontrato. La validità della configurazione proposta viene poi verificata nuovamente dal classificatore: se il difetto è risolto, il processo termina; altrimenti, si passa a una nuova iterazione di regressione e classificazione.
La figura seguente illustra l'architettura implementata per la correzione automatica dei parametri di taglio:

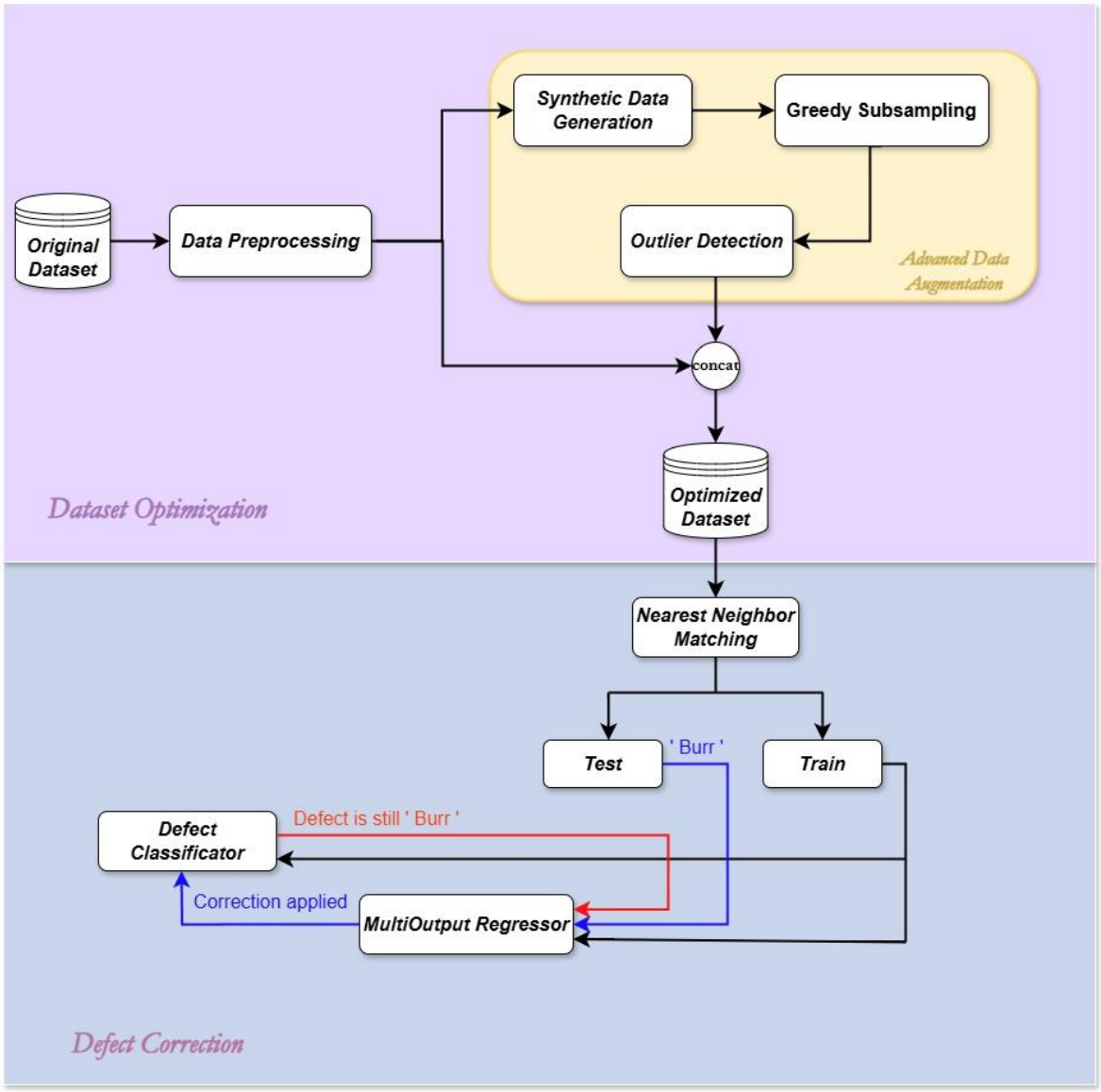


Figura 3.43: Architettura del modello per la correzione dei parametri

3.7 Algoritmi per la classificazione dei difetti

Per affrontare il problema della classificazione dei difetti, è stata valutata l'applicazione di diversi algoritmi di machine learning supervisionato. L'obiettivo era identificare un modello in grado di associare efficacemente una specifica configurazione di parametri di taglio al difetto osservato sul pezzo.

A tal fine, sono stati considerati sia modelli lineari che non lineari, al fine di confrontarne le prestazioni in termini di accuratezza, capacità di generalizzazione e robustezza rispetto a rumore o squilibri nei dati. Sono stati testati diversi classificatori ampiamente utilizzati nella letteratura scientifica, ognuno con caratteristiche peculiari in termini di interpretabilità, complessità computazionale e adattabilità al problema. Di seguito ne viene presentata una breve descrizione introduttiva.

3.7.1 Decision-Tree

I Decision Tree (alberi decisionali) sono modelli predittivi ampiamente utilizzati sia per problemi di classificazione che di regressione.

Essi operano suddividendo ricorsivamente il dataset in sottoinsiemi sempre più omogenei, sulla base di condizioni logiche applicate alle feature di input.
Ogni nodo dell'albero rappresenta una variabile su cui viene eseguito uno split, ovvero una decisione binaria o multiclasse, mentre le foglie corrispondono a classi di output (nel caso di classificazione) o a valori numerici (per la regressione).

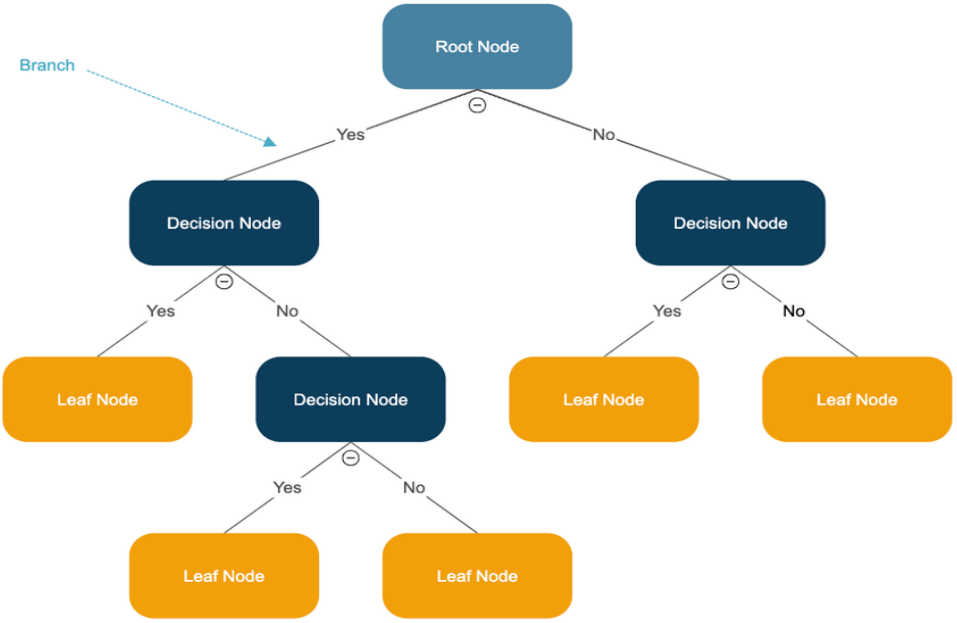


Figura 3.44: Decision Tree

La scelta della variabile e della soglia su cui effettuare la partizione è guidata da metriche che misurano la purezza del sottoinsieme risultante. Le metriche più comunemente utilizzate nei problemi di classificazione sono:

- *Entropia*:

  Misura il grado di incertezza o disordine in un insieme di dati. Se tutte le istanze appartengono alla stessa classe, l'entropia è nulla; viceversa, l'entropia cresce all'aumentare della mescolanza tra classi. Un buon split è quindi quello che produce sottogruppi a bassa entropia.

  Si calcola come:

$$H(S) = -\sum_{i=1}^{C} p_i \log_2(p_i)$$

  Dove p_i è la proporzione di elementi della classe i nel sottoinsieme S, e C è il numero totale di classi. L'obiettivo è ridurre l'entropia, ovvero ottenere insiemi più "puri" in termini di classe.

- *Information Gain*:

  L'Information Gain (guadagno informativo) misura la riduzione di entropia ottenuta dopo una partizione. Indica quanto "ordine" viene guadagnato scegliendo una certa variabile come criterio di suddivisione. Maggiore è il guadagno informativo, più efficace è lo split.

  Espressa come:

$$IG(S, A) = H(S) - \sum_{v \in \text{Val}(A)} \frac{|S_v|}{|S|} H(S_v)$$

  dove A è la variabile candidata per lo split, $Val(A)$ è l'insieme dei suoi possibili valori, e $\boldsymbol{S_v}$ è il sottoinsieme dei dati in cui $A = v$.

- *Gini Index*:

  Quantifica la probabilità che due campioni scelti a caso da un insieme

appartengano a classi diverse. Anche in questo caso, valori bassi dell'indice indicano alta purezza del nodo. L'indice di Gini è meno sensibile rispetto all'entropia e viene spesso preferito per motivi computazionali.

Si calcola come:

$$Gini(S) = 1 - \sum_{i=1}^{C} p_i^2$$

Viene spesso utilizzato per la sua semplicità computazionale e tende a penalizzare meno fortemente le partizioni imperfette rispetto all'entropia.

3.7.2Random Forest

ll Random Forest [20] è un algoritmo di apprendimento supervisionato che si basa sull'idea di combinare più alberi decisionali per ottenere un modello più robusto e preciso. L'approccio di base è quello dell'ensemble learning, in cui un insieme di modelli deboli (in questo caso, alberi) viene aggregato per costruire un predittore forte. A differenza di un singolo decision tree, che può soffrire di overfitting, il Random Forest riduce significativamente la varianza grazie all'introduzione della casualità sia nella selezione dei dati sia delle caratteristiche durante la costruzione degli alberi. Durante l'addestramento, vengono generati numerosi alberi decisionali decisionali su sottoinsiemi casuali del dataset ottenuti mediante bootstrap sampling (campionamento con ripetizione).

Inoltre, a ogni nodo, la scelta della variabile su cui dividere non avviene sull'intero set di feature, ma solo su un sottoinsieme casuale di queste. Questo introduce diversità tra gli alberi, riducendo il rischio che si concentrino tutti sugli stessi pattern o rumori del dataset.In fase di previsione, ogni albero restituisce una predizione indipendente, e il risultato finale è ottenuto tramite una votazione di maggioranza (per problemi di classificazione) o una media (per problemi di regressione). Questo meccanismo rende il Random Forest particolarmente efficace in contesti complessi e con dati rumorosi o incompleti.

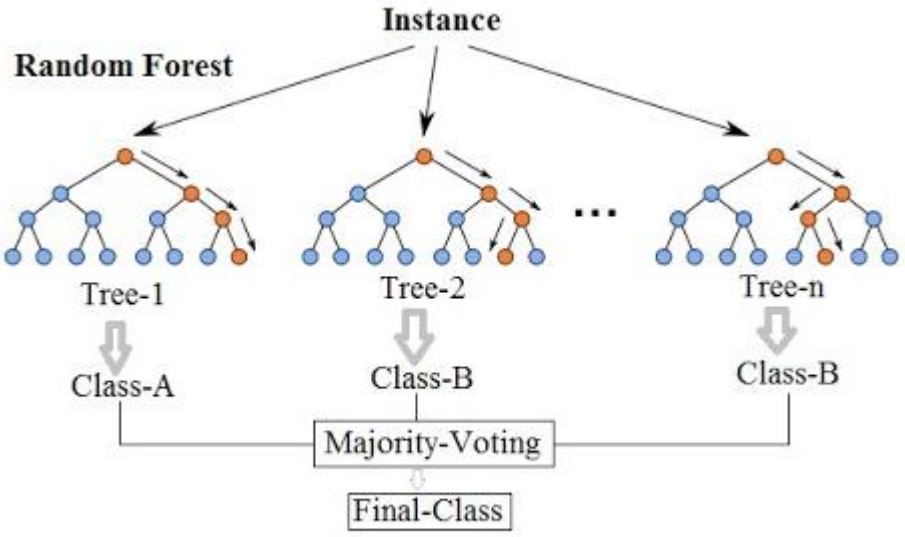


Figura 3.45: Random Forest

Le metriche più comuni utilizzate per valutare la qualità delle suddivisioni nei singoli alberi sono l'entropia, l'Information Gain e l'indice di Gini, già introdotti nel paragrafo precedente. Durante la costruzione del Random Forest, queste metriche continuano a guidare le scelte locali nei singoli alberi.

Un ulteriore vantaggio del Random Forest è la capacità di valutare l'importanza delle variabili, calcolata come la riduzione media dell'impurità (ad esempio Gini) attribuibile a ogni variabile lungo tutti gli alberi. Questo fornisce un utile strumento interpretativo anche in modelli ad alta complessità.

3.7.3 Support Vector Machine (SVM)

l Support Vector Machine (SVM) è un algoritmo supervisionato introdotto da Cortes e Vapnik (1995) [21] che si è affermato come una delle tecniche più efficaci per la classificazione binaria, con ottime prestazioni anche in contesti complessi e con dati non linearmente separabili. L'obiettivo dell'SVM è quello di trovare un iperpiano di separazione ottimale che massimizzi il margine, ovvero la distanza tra i punti di dati appartenenti a classi differenti e il confine di decisione. I punti che si trovano più vicini all'iperpiano e che determinano tale margine sono chiamati support vectors, da cui il nome dell'algoritmo. Formalmente, dato un dataset (x_i, y_i) con $x_i \in R^n$ e $y_i \in \{-1,1\}$, l'SVM cerca il vettore w e il bias b tali che l'iperpiano $w^T x + b = 0$ separi le due classi con il massimo margine.

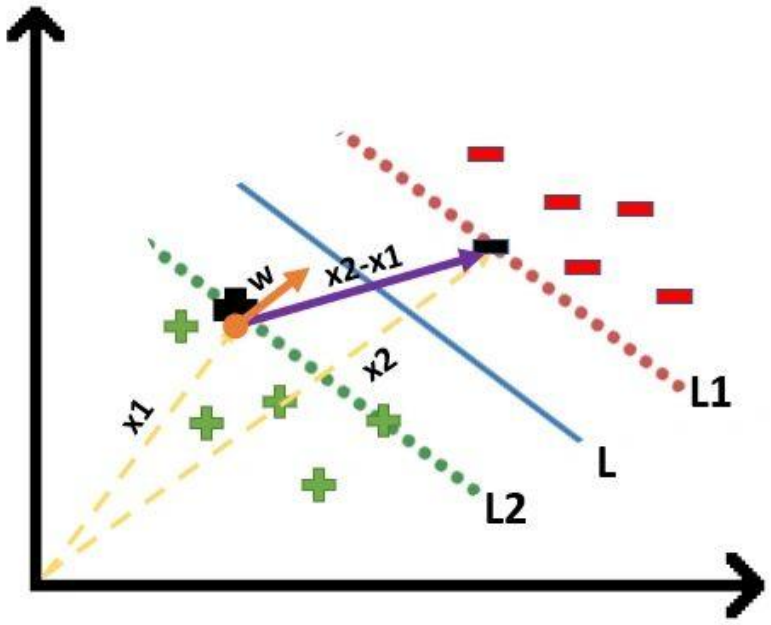


Figura 3.46: Support Vector Machine

Il problema di ottimizzazione può essere espresso come:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{soggetto a} \quad y_i(w^T x_i + b) \geq 1 \; \forall i$$

Dove:

- w è il vettore dei pesi (coefficiente della retta/iperpiano),
- b è il termine di bias (intercetta),
- x_i è il vettore delle feature dell'i-esimo campione,
- $y_i \in \{-1,1\}$ è l'etichetta associata a x_i,
- $\| w \|^2$ è la norma quadrata del vettore dei pesi, cioè $w^T w$

Minimizzare la norma di w equivale a massimizzare il margine tra le classi. Più piccolo è $\| w \|$, maggiore è la distanza tra il piano di separazione e i support vectors.

Tuttavia, nel caso in cui i dati non siano perfettamente separabili linearmente, viene introdotto un meccanismo di soft margin, che consente alcune violazioni del margine ottimale. Per gestire tali casi, si introducono le cosiddette slack variables ξ_i che quantificano il grado di errore tollerato per ciascun punto campione.

In particolare, una slack variable $\xi_i > 0$ indica che il punto i-esimo si trova all'interno del margine o addirittura sul lato sbagliato dell'iperpiano di separazione.

Il termine di penalizzazione associato a queste variabili viene controllato da un parametro di regolarizzazione C, che regola il trade-off tra l'ampiezza del margine e il numero di violazioni ammesse.

Formalmente, il problema di ottimizzazione dell'SVM con slack variables si scrive come:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

soggetto ai vincoli:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{per ogni } i$$

dove:

- w è il vettore dei pesi,
- b è il termine di bias,
- ξ_i è la slack variable associata all'esempio i-esimo
- C è un parametro positivo che controlla la penalità per gli errori di classificazione.

Per problemi non linearmente separabili nello spazio originale, l'SVM può essere esteso grazie anche al kernel trick, che permette di proiettare i dati in uno spazio a dimensione superiore dove è possibile trovare una separazione lineare. I kernel più comuni sono il polinomiale, il radiale (RBF/Gaussian kernel) e il kernel sigmoide. Il kernel RBF, ad esempio, è definito come:

$$K(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right)$$

Grazie a queste proprietà, l'SVM risulta particolarmente adatto per problemi ad alta dimensionalità, con un numero di variabili elevato rispetto al numero di osservazioni. Tuttavia, la scelta del kernel e dei parametri di regolarizzazione può influenzare sensibilmente le prestazioni.

3.7.4 Classificatori basati su MLP

Tra i modelli sperimentati per la classificazione dei difetti figura anche il Multi-Layer Perceptron (MLP), una rete neurale feedforward a più strati. Dopo averne descritto il funzionamento generale nello stato dell'arte, in questo contesto il MLP è stato impiegato come classificatore supervisionato, capace di apprendere la mappatura tra le configurazioni dei parametri di processo e la qualità del taglio (buono o cattivo). Nel caso binario, l'output del modello rappresenta la probabilità che una data configurazione produca un taglio accettabile, tramite una funzione di attivazione sigmoid. Per la classificazione dei difetti specifici, il modello può essere facilmente esteso a un'impostazione multiclasse, adottando una funzione di attivazione softmax nello strato di output.

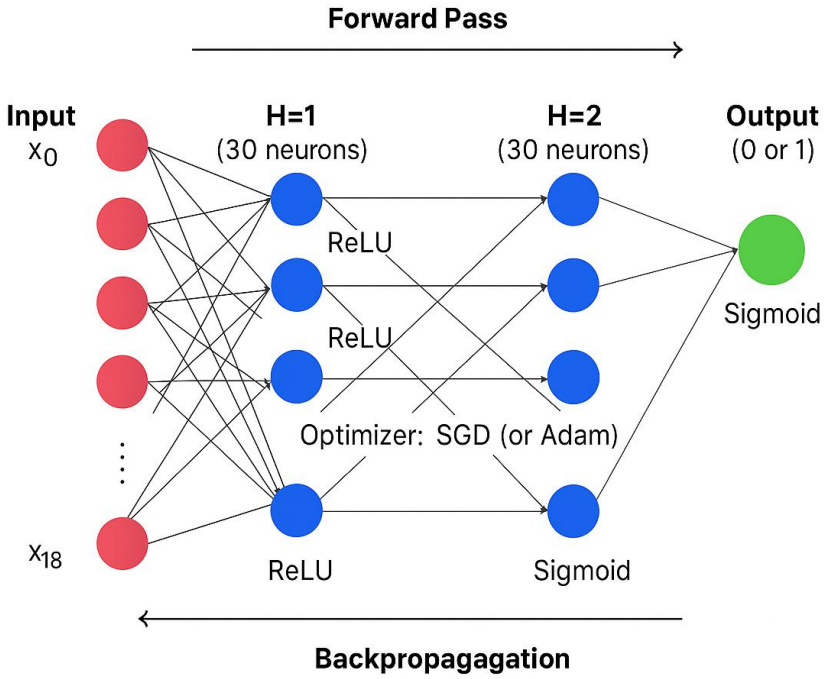


Figura 3.47: architettura del Multi-Layer Perception

L'MLP si dimostra particolarmente efficace in problemi non lineari grazie alla sua capacità di apprendere rappresentazioni complesse dai dati. Tuttavia, richiede una buona quantità di dati per essere allenato efficacemente e può risultare più suscettibile all'overfitting rispetto ad altri modelli più semplici, come gli alberi decisionali, se non opportunamente regolarizzato.

3.7.5 Classificatori basati su Gradient Boosting

Il Gradient Boosting [22] è una tecnica di apprendimento ensemble basata su alberi decisionali, (anche se possono essere usati anche altri tipi di predittori, come i lineari) concepita per costruire modelli predittivi robusti combinando più deboli predittori (tipicamente decision tree a bassa profondità) in modo incrementale. L'intuizione alla base di questo metodo è quella di correggere gli errori commessi dai modelli precedenti mediante un'ottimizzazione iterativa basata sul gradiente della funzione di perdita.A differenza del bagging (come nel Random Forest), che costruisce modelli in parallelo, il Gradient Boosting adotta un approccio sequenziale, dove ogni nuovo albero viene addestrato per correggere gli errori del modello composto fino a quel momento.

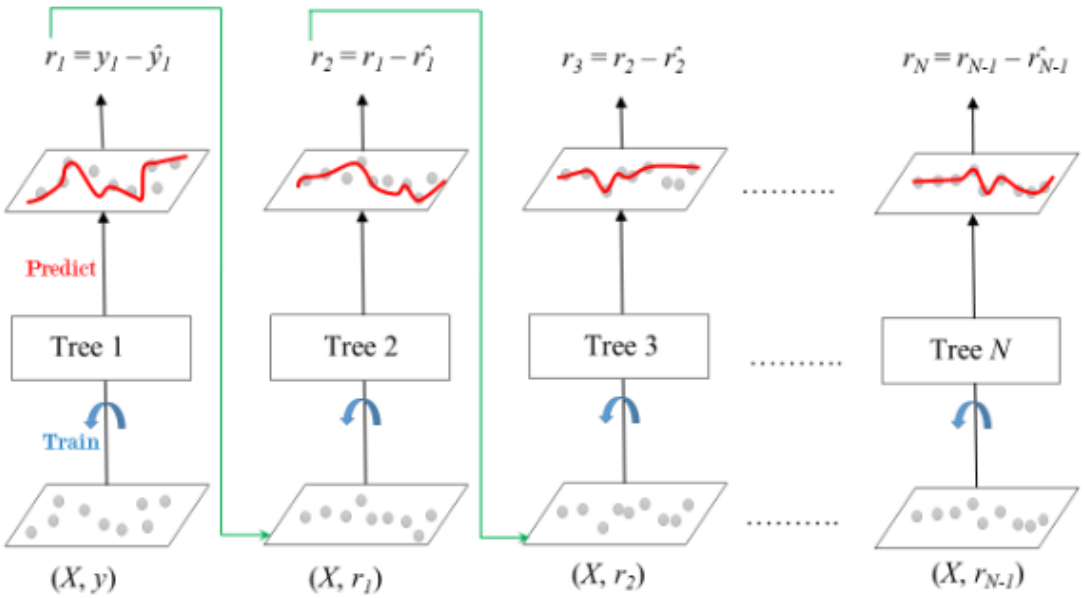


Figura 3.48: architettura del Gradient Boosting

Funzionamento

Il modello si costruisce iterativamente secondo i seguenti passaggi:

1. *Inizializzazione*:
   Si parte da una predizione iniziale costante
2. *Calcolo del gradiente*:
   Ad ogni iterazione, si calcola il gradiente della funzione di perdita rispetto alla predizione corrente: questo rappresenta l'errore residuo.
3. *Addestramento del nuovo albero*:
   Un nuovo predittore è addestrato per prevedere questo errore residuo.
4. *Aggiornamento della predizione*:
   Il nuovo albero è combinato al modello precedente con un fattore di apprendimento (learning rate), controllando l'impatto di ogni nuova aggiunta.

Ripetendo il processo per M iterazioni, il modello finale risulta dalla somma ponderata di tutti gli alberi:

$$\hat{y}(x) = \sum_{m=1}^{M} \gamma_m h_m(x)$$

Dove $h_m(x)$ è l'albero alla mmm-esima iterazione e γ_m è il learning rate applicato.

Funzione di perdita

Il Gradient Boosting può adattarsi a diverse funzioni di perdita, tra cui:

- *Log-loss:*
  Per classificazione binaria, configurazione buona o cattiva:

$$L(y, \hat{y}) = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]$$

- *Cross-Entropy loss:*
  Per classificazione multiclasse, classifica i diversi difetti

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

dove:

- C è il numero totale di classi,
- y_i è l'etichetta reale
- $\hat{y}$ è la probabilità predetta per la classe iii, tipicamente ottenuta tramite funzione softmax sull'output del modello.

La forza dell'algoritmo risiede nell'approccio iterativo, che consente di correggere progressivamente gli errori commessi nei passaggi precedenti.
Inoltre, la possibilità di utilizzare diverse funzioni di perdita lo rende estremamente flessibile sia per problemi di classificazione binaria che multiclasse. Tuttavia, è importante tenere conto del rischio di overfitting in presenza di un numero elevato di iterazioni o alberi troppo profondi.

Per questo motivo, la scelta di iperparametri come il learning rate, la profondità massima degli alberi e il numero di stime diventa cruciale per ottenere un buon compromesso tra accuratezza e capacità di generalizzazione.

3.8 Algoritmi di regressione per la correzione dei difetti

In questo capitolo vengono analizzati gli algoritmi di regressione utilizzati per la correzione automatica delle configurazioni di taglio. L'approccio adottato prevede l'addestramento di un regressore specifico per ciascuna classe di difetto. In questo modo, ogni modello riceve in input dati omogenei per struttura e caratteristiche, migliorando la coerenza del processo di apprendimento.

Un aspetto particolarmente critico riguarda la strategia di allenamento, trattata in dettaglio nel capitolo successivo. Infatti, a partire da una configurazione che produce difetti su un determinato materiale e spessore, possono esistere molteplici configurazioni alternative ugualmente valide, ovvero capaci di generare un taglio buono.

Questa molteplicità introduce una sfida specifica nel contesto della regressione, in quanto il modello tende a stimare un valore medio ottimale, mentre nel nostro caso non esiste una singola soluzione migliore. Inoltre, non è disponibile una metrica quantitativa univoca per valutare la bontà relativa di ciascuna configurazione buona, rendendo più complesso il compito di apprendimento del regressore.

3.8.1 CatBoost Regressor

Nel contesto del problema affrontato, in cui si mira a stimare una configurazione di corretta a partire da una che genera difetti di taglio, è stato adottato anche il CatBoost Regressor.

Questo modello, parte della famiglia degli algoritmi di gradient boosting su alberi decisionali, è stato sviluppato dal team di ricerca di Yandex [23] con l'obiettivo di fornire un approccio più stabile, soprattutto in scenari caratterizzati dalla presenza di feature categoriali e dataset di dimensioni limitate.

A differenza di altri algoritmi di boosting, CatBoost introduce due innovazioni principali:

1. Una gestione interna delle variabili categoriche, che evita la necessità di codifica manuale
2. Una strategia di Ordered Boosting che riduce il rischio di target leakage e overfitting, particolarmente utile in dataset piccoli o altamente rumorosi.

Nel task di regressione, CatBoost inizia con una predizione iniziale costante, tipicamente la media dei valori target $\hat{y}$, e costruisce un modello additivo iterativo secondo la formula:

$$F_0(x) = \bar{y}, \quad F_m(x) = F_{m-1}(x) + \eta \cdot h_m(x)$$

dove $\boldsymbol{F_m(x)}$ è il modello alla m-esima iterazione, η è il learning rate, e $h_m(x)$ è il regressore (un decision tree) addestrato a predire i residui:

$$r_i^{(m)} = y_i - F_{m-1}(x_i)$$

La funzione obiettivo da minimizzare è la Multi-MSE per il caso di regressione, definita come:

$$\text{MultiMSE} = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} (y_{i,j} - \hat{y}_{i,j})^2$$

Nel nostro caso specifico useremo come funzione di perdita la MSE multivariata, visto che in output non vogliamo un singolo valore, ma tutti i parametri di configurazione del taglio corretti, che ricordiamo essere:
$CONTOUR\_SPEED$, $LASER\_POWER$, $CONTOUR\_GAS\_PRESSURE$, $CONTOUR\_FOCAL$,$CONTOUR\_NOZZLE\_DISTANCE$ e $NOZZLE\_TYPE$.

3.8.2 MLP Regressor

Nel contesto del problema affrontato, è stato adottato anche il Multi-Layer Perceptron Regressor (MLP Regressor), un modello di apprendimento supervisionato appartenente alla famiglia delle reti neurali artificiali feedforward, già introdotte nei capitoli precedenti, ma qui con un maggiore focus sul task di regressione. A differenza degli algoritmi basati su alberi decisionali, l'MLP si basa su una rete di neuroni organizzati in strati completamente connessi, capaci di modellare relazioni complesse e non lineari tra input e output. Ogni neurone esegue una trasformazione lineare dei propri input, seguita da una funzione di attivazione non lineare nei layer nascosti. A differenza del caso in cui viene impiegato per la classificazione, l'output finale della rete non è soggetto ad alcuna funzione di attivazione, poiché il task di regressione richiede valori reali non vincolati.

Nel task di regressione, il modello MLP cerca di approssimare una funzione $f(x;\theta)$ tale che:

$$\hat{\mathbf{y}} = f(\mathbf{x};\theta)$$

dove:

- $x \in R_n$ è il vettore delle feature di input
- $\hat{y} \in R_m$ è il vettore delle configurazioni corrette stimate,
- θ rappresenta l'insieme dei pesi e dei bias della rete.

Il modello viene ottimizzato minimizzando la funzione di perdita nota come Mean Squared Error (MSE) multivariata:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|^2 = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{m} (\hat{y}_{i,j} - y_{i,j})^2$$

dove $\hat{y}_i$ rappresenta la configurazione predetta per l'i-esimo esempio e y_i è la configurazione target corretta.

Un aspetto critico nell'impiego dell'MLP Regressor è la preparazione accurata dei dati in fase di preprocessing. A differenza di modelli come CatBoost, che gestiscono internamente le feature categoriche e risultano meno sensibili alla scala delle variabili, l'MLP non è in grado di trattare direttamente né variabili categoriche né feature con scale eterogenee.

Risulta dunque indispensabile:

- Codificare manualmente le variabili categoriche
- Normalizzare o standardizzare le variabili continue, affinché abbiano distribuzioni comparabili, evitando che alcune feature dominino il processo di ottimizzazione.

Questa esigenza rende il preprocessing dei dati per l'MLP più complesso e delicato rispetto a quello richiesto da modelli basati su gradient boosting come CatBoost, che sono progettati per essere più robusti e automatizzati in questi aspetti.

3.9 Normalizzazione e codifica delle variabili

Nel contesto del machine learning, la normalizzazione delle variabili continue **e** la codifica di quelle categoriche rappresentano passaggi fondamentali nella preparazione dei dati, ma non tutti i modelli ne hanno bisogno nella stessa misura. Alcuni algoritmi, come i Multi-Layer Perceptron (MLP), le Support Vector Machine (SVM) , i modelli di regressione lineare o in generale i modelli basati sul calcolo di distanze richiedono esplicitamente la normalizzazione degli input per garantire un apprendimento efficace. Questo perché spesso la loro ottimizzazione si basa sulla discesa del gradiente: ogni parametro viene aggiornato sulla base della derivata della funzione di errore rispetto a quel parametro. Se le variabili di input hanno scale molto diverse i gradienti associati risulteranno sbilanciati, causando un apprendimento instabile o lento, con alcuni parametri che dominano sugli altri.

Nei modelli neurali, in particolare, la retropropagazione del gradiente attraversa vari livelli della rete, rendendo cruciale che gli input abbiano una scala coerente per evitare problemi di esplosione o scomparsa del gradiente.Al contrario, nei modelli basati su alberi decisionali, come i Random Forest i modelli di Gradient Boosting o in particolare il CatBoost, la normalizzazione delle feature non è necessaria.

Questi algoritmi prendono decisioni basate su soglie, e non su distanze geometriche o proiezioni scalari. In più, anche se nel boosting si fa uso di gradienti per migliorare le predizioni, questi gradienti non sono usati per aggiornare dei pesi numerici tramite retropropagazione, bensì come target di apprendimento per i successivi alberi. Di conseguenza, la scala delle variabili non influenza direttamente il meccanismo di apprendimento, e il modello rimane robusto anche in presenza di feature su ordini di

grandezza differenti. Infine, un aspetto critico riguarda il trattamento delle variabili categoriche. Algoritmi come MLP, SVM o regressione lineare non possono gestire direttamente feature non numeriche, e richiedono che queste vengano codificate.

Al contrario, modelli come CatBoost sono in grado di gestire le feature categoriche internamente, grazie a una procedura di encoding statistico che preserva l'informazione, risultando particolarmente utile in contesti reali con variabili simboliche e dataset ridotti. Seguono le principali tecniche di normalizzazione e codifica adottate nella soluzione proposta, per i modelli che ne sfruttano le potenzialità.

3.9.1 Normalizzazione standard

La normalizzazione standard (o standardizzazione) è una tecnica di pre-processing utilizzata per trasformare le variabili continue in modo da renderle confrontabili tra loro e più adatte a determinati algoritmi di machine learning. Essa consiste nel riportare ciascuna variabile ad una distribuzione con media 0 e deviazione standard 1, secondo la formula:

$$x_i^{\text{norm}} = \frac{x_i - \mu}{\sigma}$$

dove x_i è il valore originale della variabile, μ è la media della variabile sul dataset, e σ è la deviazione standard.

3.9.2 Normalizzazione MinMax

Una tecnica comunemente utilizzata è il MinMax Scaler, che trasforma ogni valore in un intervallo prefissato, solitamente [0,1], preservando le relazioni lineari tra i dati. La formula applicata per ciascun valore xxx di una variabile continua è la seguente:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

dove:

- x_{min} e x_{max} rappresentano rispettivamente il minimo e il massimo osservati nella variabile durante il fit dello scaler;
- x_{scaled} è il valore normalizzato.

3.9.3 One-hot encoding

La one-hot encoding è una tecnica di codifica utilizzata per trasformare le variabili categoriche in un formato numerico adatto agli algoritmi di machine learning. Per ogni categoria distinta, viene creata una nuova variabile binaria (dummy variable) che assume valore 1 se l'osservazione appartiene a quella categoria, e 0 altrimenti.

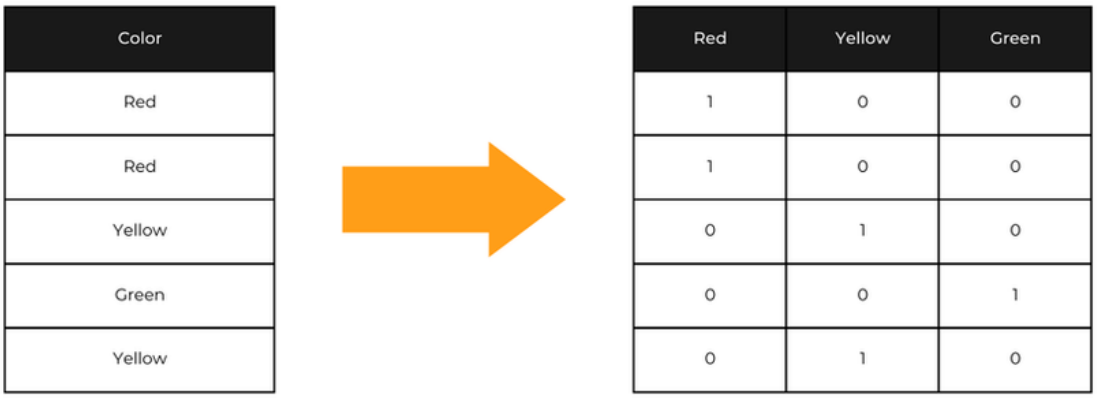


Figura 3.49: One-hot encoding per una variabile categorica

Questa rappresentazione è necessaria per tutti i modelli che non sono in grado di gestire direttamente feature non numeriche.

Tuttavia, il numero di nuove variabili introdotte cresce con il numero di categorie e può aumentare significativamente la dimensionalità del dataset, con conseguenze in termini di memoria e overfitting.

3.10 Strategia di allenamento dei modelli di regressione

Per far fronte alla molteplicità delle soluzioni valide a parità di input, è stata adottata una strategia che associa a ogni configurazione non buona la configurazione corretta più vicina in termini di distanza nello spazio dei parametri. Questo approccio si è reso necessario per affrontare una criticità tipica dei problemi di regressione in contesti con molteplici soluzioni ottime: a differenza della classificazione, dove si prevede una sola etichetta corretta per ogni input, nel caso specifico del nostro task, una stessa configurazione di taglio difettosa può essere corretta in diversi modi ugualmente

validi, che portano a risultati soddisfacenti. Tuttavia, i modelli di regressione tradizionali, come MLP o CatBoost Regressor, sono progettati per minimizzare una funzione di perdita rispetto a un solo valore target, assumendo implicitamente che esista una sola soluzione ottimale per ogni input. Questo può indurre il modello a "mediare" tra soluzioni diverse, generando configurazioni non realistiche o subottimali.

3.10.1 Nearest Neighbor Matching tra input e target

Come già introdotto nei paragrafi precedenti, la strategia di Nearest Neighbor Matching adottata per accoppiare ciascuna configurazione difettosa a una configurazione corretta mira a definire una mappatura più coerente tra input e target.
L'idea è quella di fornire al regressore un obiettivo realistico e raggiungibile: applicare la minima correzione necessaria per trasformare una configurazione che genera difetti in una che produce un taglio di qualità.

Questa impostazione si basa sul principio che configurazioni difettose simili tra loro dovrebbero corrispondere a target (configurazioni corrette) anch'essi simili, riducendo la variabilità inutile nel dataset. Tale approccio aiuta a regolarizzare l'apprendimento e a limitare il rischio che il modello si disperda nel tentativo di approssimare target molto diversi per input quasi identici, fenomeno comune nei problemi di regressione in cui esistono molteplici soluzioni valide.
In questo modo, si favorisce una maggiore stabilità delle predizioni e si facilita la convergenza del modello durante l'allenamento. Se da un lato questa strategia consente di effettuare correzioni più conservative, dall'altro può limitarne la capacità di adattarsi a configurazioni difettose fuori distribuzione o lontane da quelle osservate in fase di addestramento, per via della scarsa capacità di generalizzare nelle possibili soluzioni.

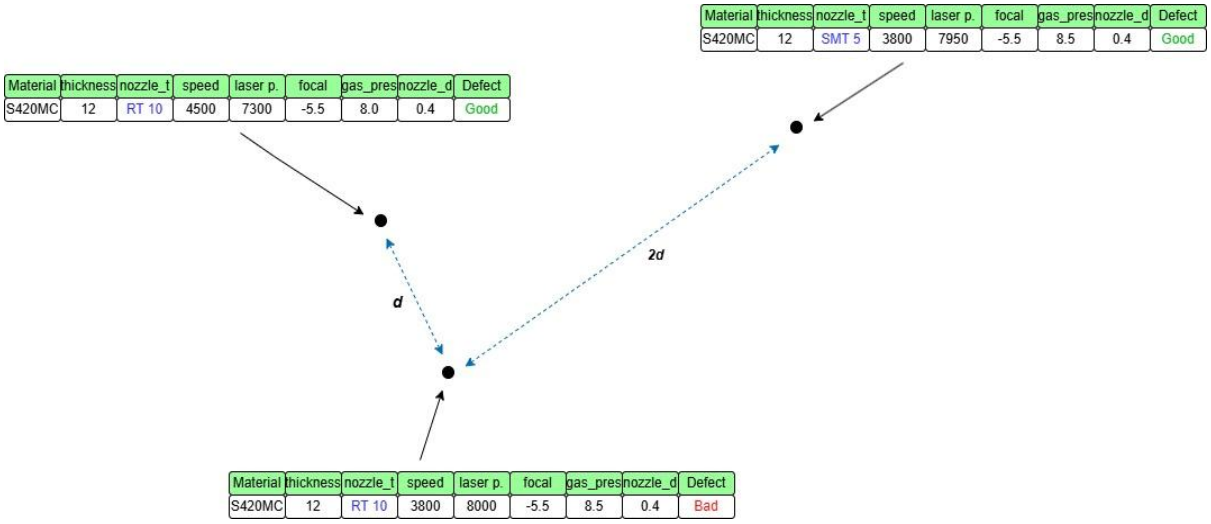


Figura: 3.50: Nearest Neighbor Matching

Per realizzare la strategia di accoppiamento tra configurazioni difettose e buone è stato adottato un approccio basato su una metrica ispirata alla distanza di Gower, in grado di combinare informazioni sia da feature continue che categoriche definita come:

$$D_{\text{Gower}}(x, y) = \frac{1}{p} \sum_{j=1}^{p} d_j(x_j, y_j)$$

dove:

- p è il numero totale di feature (sia continue che categoriche);
- $d_j\ (x_j, y_j)$ è il contributo alla distanza della j-esima feature, calcolato come:

dove per le feature cotinue:

$$d_j(x_j, y_j) = \frac{|x_j - y_j|}{R_j}$$

con $R_j = max(x_j) - min(x_j)$ range della feature, in pratica è la differenza assoluta normalizzata.

Mentre invece per le features categoriche:

$$d_j(x_j, y_j) = \begin{cases} 0 & \text{se } x_j = y_j \\ 1 & \text{se } x_j \neq y_j \end{cases}$$

Le feature continue (ad esempio: potenza del laser, velocità di taglio, pressione del gas, distanza focale, ecc.) vengono preventivamente normalizzate tramite tecniche come il MinMaxScaler o lo StandardScaler, con l'obiettivo di uniformare le scale di misura e rendere le grandezze confrontabili. La distanza tra configurazioni viene quindi calcolata come somma delle differenze assolute normalizzate sulle feature continue, riducendo l'impatto di unità e range eterogenei.

Per quanto riguarda le feature categoriche, nel caso specifico rappresentate dal tipo di ugello ($NOZZLE\_TYPE$), si applica una penalità discreta ogniqualvolta il valore differisce tra configurazione difettosa e buona. Questo approccio si basa su due osservazioni fondamentali:

1. una considerazione tecnica, ovvero che il cambio di ugello implica una sostituzione fisica effettiva, con impatti tangibili sul processo di taglio;
2. una considerazione analitica, secondo cui le diverse tipologie di ugelli generano gruppi separati nello spazio delle configurazioni, rendendo ragionevole attribuire maggiore distanza a combinazioni eterogenee.

Il valore finale della distanza complessiva viene infine normalizzato rispetto al numero totale di feature (continue + categoriche), ottenendo così una misura bilanciata e omogenea della dissimilarità tra le due configurazioni.

Capitolo 4

Test e risultati

In questo capitolo mostreremo i risultati ottenuti con i modelli di classificazione e di regressione per la correzione dei parametri di taglio, confrontandoli anche con quelli che si sarebbero ottenuti senza la fase di ottimizzazione del dataset. Inoltre verranno forniti anche i parametri di configurazione del TVAE per la generazione dei dati sintetici e i parametri di contaminazione dell'isolation forest associati ad ogni difetto per la fase di outlier detection. In seguito vengono introdotte brevemente le metriche utilizzate per misurare le performance dei gli algoritmi e gli errori commessi sulle predizioni.

4.1 Metriche di valutazione delle performance del classificatore

Per valutare l'accuratezza del classificatore nel caso di classificazione multiclasse, è stata utilizzata la metrica f1-score, poiché tiene conto sia delle prestazioni per ogni classe, sia della loro rappresentatività.

4.1.1 F1-Score

L'F1-score è una metrica ampiamente utilizzata per valutare le performance di un modello di classificazione, soprattutto nei casi in cui vi sia squilibrio tra le predizioni delle classi. L'F1-score rappresenta la media armonica tra *precision* e *recall*, e cerca di trovare un equilibrio tra questi due aspetti.

Dove il termine *precision* è calcolato come:

$$\text{Precision} = \frac{TP}{TP + FP}$$

dove:

- TP: veri positivi (True Positives)
- FP: falsi positivi (False Positives)

E indica quante volte il classificatore ha avuto ragione su tutte le volte che ha predetto una determinata classe.

Mentre il termine *recall* è calcolato come:

$$\text{Recall} = \frac{TP}{TP + FN}$$

dove:

- FN: falsi negativi (False Negatives)

Indica: su tutte le volte in cui qualcosa era davvero di una determinata classe, quante volte il modello ha indovinato

Pertanto l'F1-score si calcola come la media armonica tra precision e recall:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

E serve a garantire che l'F1-score sia alto solo se entrambi sono alti, e che sia basso se almeno uno dei due è basso.

4.2 Metriche di valutazione delle performance sulla correzione in fase di test

Nel task di regressione volto alla correzione dei parametri di taglio, la valutazione dell'errore sulle predizioni in fase di test è stata effettuata confrontando ogni configurazione predetta con la configurazione buona più vicina nel dataset, a parità di materiale e spessore. Tale accoppiamento consente di quantificare l'errore in modo più realistico rispetto all'ambito applicativo, poichè come già detto, in questo caso, non esiste un unico target buono. Per quanto riguarda le variabili continue, l'errore è stato calcolato utilizzando la Mean Absolute Error (MAE), secondo la formula:

dove:

- y_i è il valore reale della variabile continua nella configurazione target (nel nostro caso la più vicina)
- $\hat{y}$ è il valore predetto dal modello,
- n è il numero totale di variabili continue.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Per le variabili categoriche, come ad esempio il tipo di ugello (*NOZZLE_TYPE*), non si può applicare una metrica di tipo numerico. In tal caso, si utilizza una misura basata sulla frequenza di mismatch, cioè la proporzione di predizioni errate rispetto alla configurazione target. La frequenza di errore categoriale è calcolata come:

$$\text{Frequenza\_mismatch} = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{\{c_j \neq \hat{c}_j\}}$$

dove:

- c_j è il valore della feature categorica nella configurazione target,
- $\hat{c}$ è il valore predetto dal modello,
- $1_{\{c_j \neq c^j\}}$ è una funzione indicatrice che vale 1 se c'è un errore (mismatch),0 altrimenti,
- m è il numero totale di feature categoriche considerate.

Queste due metriche permettono di valutare separatamente l'accuratezza sulle dimensioni numeriche e su quelle simboliche, fornendo un quadro completo della capacità del modello di suggerire una correzione coerente, vicina e compatibile con le buone configurazioni osservate nel dataset. A questo valore si affianca un valore di "incertezza" (da 0 a 100%) che rappresenta sostanzialmente la probabilità che quella configurazione non sia in realtà buona, ed è ottenuta tramite il classificatore, sfruttando le probabilità grezze associate alla classe, ma è meno rilevante rispetto al valore calcolato con il MAE.

4.3 Risultati ottenuti nel task di classificazione

In questa parte mostriamo la capacità discriminativa del classificatore nel distinguere i vari difetti, prima e dopo la fase di ottimizzazione del dataset. Inoltre gli hyperparameters dei classificatori sono stati scelti mediante grid search utilizzando il 20% del training set per la validazione.

Pre-ottimizzazione del dataset:

Full dataset				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
115	89	45	25	25

Train				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
92	71	20	20	36

Test				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
23	18	5	5	9

Figura 4.0: dataset originale

Train F1-Score Accuracy					
n. esecuzioni: 100	*Burr*	*Cutting loss*	*Cutting torn*	*No defects*	*Plasma*
Decision Tree	0.87	0.76	0.87	0.85	0.88
Random Forest	0.86	0.59	0.91	0.84	0.89
MLP Classifier	0.72	0.34	0.65	0.71	0.69
CatBoost Classifier	0.86	0.71	0.91	0.85	0.91
Linear SVM	0.71	0.56	0.62	0.67	0.81
Polinomial SVM	0.97	1.0	0.96	0.98	1.0

Test F1-Score Accuracy					
n. esecuzioni: 100	*Burr*	*Cutting loss*	*Cutting torn*	*No defects*	*Plasma*
Decision Tree	0.62	0.41	0.62	0.60	0.82
Random Forest	0.66	0.17	0.73	0.65	0.83
MLP Classifier	0.62	0.14	0.53	0.60	0.57
CatBoost Classifier	0.67	0.44	0.67	0.61	0.87
Linear SVM	0.61	0.00	0.67	0.57	0.50
Polinomial SVM	0.62	0.33	0.78	0.61	0.80

Figura 4.1: Risultati F1-score per test e train

Tutti i modelli ottengono punteggi piuttosto alti in fase di addestramento, in particolare:

- Random Forest e CatBoost raggiungono valori elevati in quasi tutte le classi (fino a 0.91 per '*Cutting torn'* con CatBoost).
- Il Decision Tree si comporta bene su tutte le classi (0.85–0.88), segno che è in grado di adattarsi bene ai dati.
- MLP Classifier mostra valori molto bassi, soprattutto su '*No defects'* (0.27), il che suggerisce difficoltà nell'apprendere questa classe anche in training.

Questi risultati sembrano soffrire di overfitting visto che ottengono punteggi molto più alti sul train rispetto al test, ad eccezione del MLP Classifier che non riesce a trovare una buona funzione di mapping tra configurazione in input e difetto (target) andando male praticamente in tutti i casi.

Post-ottimizzazione dataset:

Full dataset				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
299	240	196	147	110

Train				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
239	192	157	117	88

Test				
Burr	*No defects*	*Cutting torn*	*Cutting loss*	*Plasma*
60	48	39	30	22

Figura 4.2: Dataset post ottimizzazione

Il dataset ottimizzato è stato ottenuto mediante l'utilizzo dei seguenti hyperparametri che riguardano principalmente il TVAE, mentre il parametro *target_len* rappresenta la dimensione desiderata in output dall'algoritmo greedy:

Generazione sintetica con TVAE**:**

TVAE + Greedy Selection					
	Burr	No defects	Cutting torn	Cutting loss	Plasma
Epochs	1000	1000	2000	2000	2000
Gen_lr	1e-5	1e-5	1e-5	1e-5	1e-5
Disc_lr	1e-5	1e-5	1e-5	1e-5	1e-5
Batch_size	20	18	10	5	5
Raw_target_len	1000	1000	650	500	500
Target_len	345	267	180	150	100

Figura 4.3: hyperparametri del TVAE

Rimozione anomalie con Isolation Forest:

Isolation Forest					
	Burr	No defects	Cutting torn	Cutting loss	Plasma
contamination	0.50	0.40	0.20	0.20	0.20
n_estimators	100	100	100	100	100
output_len	299	240	196	147	110

Figura 4.4: hyperparametri Isolation Forest

In questo caso il parametro *contamination* è stato utile anche per ridimensionare le classi, sfoltendo quelle più numerose.

In figura 4.5 vengono mostrati i valori di F1-Score ottenuti sul dataset ottimizzato:

Train F1-Score Accuracy					
n. esecuzioni: 100	Burr	Cutting loss	Cutting torn	No defects	Plasma
Decision Tree	0.92	0.92	0.96	0.90	0.98
Random Forest	0.93	0.93	0.97	0.91	0.97
MLP Classifier	0.85	0.85	0.94	0.80	0.96
CatBoost Classifier	0.93	0.95	0.97	0.92	0.98
Linear SVM	0.75	0.73	0.83	0.68	0.93
Polinomial SVM	0.93	0.94	0.97	0.93	1.0

Test F1-Score Accuracy					
n. esecuzioni: 100	Burr	Cutting loss	Cutting torn	No defects	Plasma
Decision Tree	0.75	0.76	0.88	0.68	0.94
Random Forest	0.81	0.79	0.92	0.75	0.95
MLP Classifier	0.78	0.80	0.91	0.73	0.94
CatBoost Classifier	0.82	0.84	0.93	0.76	0.94
Linear SVM	0.70	0.52	0.88	0.74	0.95
Polinomial SVM	0.79	0.68	0.90	0.77	0.93

Figura 4.5: F1-Score dopo l'ottimizzazione dei dati

I risultati sul set di test, che rappresentano il riferimento principale per valutare la capacità di generalizzazione dei modelli, mostrano che CatBoost, Random Forest e MLP Classifier offrono le prestazioni migliori sulla versione ottimizzata del dataset. Nonostante ciò, si osservano miglioramenti significativi per tutti i modelli rispetto alla versione precedente del dataset. Tuttavia, la classe "No defects" continua a rappresentare una sfida: la sua accuratezza in test rimane sensibilmente inferiore rispetto a quella ottenuta in fase di training. Questo divario potrebbe indicare la presenza di un potenziale overfitting su questa classe.

In seguito, vengono mostrati gli hyperparametri utilizzati per i modelli riportati in figura 4.5:

Hyperparametri Random Forest									
bootstrap	ccp_alpha	criterion	max_depth	max_features	max_leaf nodes	min_impurity decrease	min_sample leaf	min_samples split	n_estimators
True	0.0	entropy	40	sqrt	None	0.0	2	5	50

Hyperparametri MLP Classifier							
activation	alpha	max_iter	batch_size	hidden_size	learning_rate	learning_rate init	solver
ReLu	0.001	300	32	(50,30)	constant	0.005	Adam

Hyperparametri CatBoost Classifier						
min_data_in leaf	learning_rate	l2_leaf_reg	iterations	grow_policy	depth	Bootstrap type
10	0.05	5	500	Depthwise	8	Bernoulli

Hyperparametri Decision Tree						
ccp_alpha	criterion	max_depth	min_impurity decrease	min_samples leaf	min_samples split	max_leaf nodes
0.0005	0.76	20	0.0005	2	2	None

Hyperparametri SVM Poly			
C	coef0	degree	gamma
1	1	6	scale

Hyperparametri Linear SVM	
C	max_iter
100	-1

Figura 4.6: Hyperparameters dei modelli di classificazione

4.4 Risultati ottenuti nel task di regressione

Analogamente a quanto visto nel Capitolo 4.3, vengono riportate le performance ottenute dai modelli di regressione per la correzione delle configurazioni di taglio, considerando direttamente il dataset ottimizzato, in quanto, nella fase di classificazione, si è osservato un incremento considerevole delle prestazioni grazie all'ottimizzazione. Nel caso delle performance sul training set, utilizzeremo il Mean Absolute Error (MAE) come metrica per valutare l'errore commesso sulle predizioni delle configurazioni di taglio a parità di materiale e spessore. Inoltre, come già specificato in precedenza nel capitolo relativo al task di classificazione, anche qui verrà utilizzato il 20% del training set come validation, ma solo il 10% come test.

Hyperparametri CatBoost Regressor							
bagging_temperature	depth	early_stopping_rounds	eval_metric	l2_leaf_regularizzation	iterations	loss_function	random_stength
0	8	20	MultiRMSE	3	1000	MultiRMSE	2

MAE sul training set per le variabili continue con CatBoost Regressor				
CONTOUR_SPEED	LASER_POWER	CONTOUR_GAS_PRESSURE	CONTOUR_FOCAL	CONTOUR_NOZZLE_DIST
241.64	12.64	0.29	0.23	0.00

Mismatch (%) delle variabili categoriche
NOZZLE_TYPE
0.00

MAE sul test set per le variabili continue con CatBoost Regressor				
CONTOUR_SPEED	LASER_POWER	CONTOUR_GAS_PRESSURE	CONTOUR_FOCAL	CONTOUR_NOZZLE_DIST
246.71	39.47	0.18	0.06	0.00

Mismatch (%) delle variabili categoriche
NOZZLE_TYPE
0.00

Correction Score (%) delle correzioni effettuate sul test set per il CatBoost Regressor		
MLP Classifier	CatBoost Regressor	Random Forest
28	24	40

Hyperparametri CatBoost Regressor								
activation	alpha	batch_size	early_stopping	hidden_layer_sizes	learning_rate	max_iter	n_iter_no_change	solver
relu	0.001	32	True	(32,32)	constant	500	20	adam

MAE sul training set per le variabili continue con MLP Regressor				
CONTOUR_SPEED	LASER_POWER	CONTOUR_GAS_PRESSURE	CONTOUR_FOCAL	CONTOUR_NOZZLE_DIST
363.47	32.17	0.26	0.22	0.00

Mismatch (%) delle variabili categoriche
NOZZLE_TYPE
0.54

MAE sul test set per le variabili continue con MLP Regressor				
CONTOUR_SPEED	LASER_POWER	CONTOUR_GAS_PRESSURE	CONTOUR_FOCAL	CONTOUR_NOZZLE_DIST
461.25	162.50	0.33	0.32	0.00

Mismatch (%) delle variabili categoriche
NOZZLE_TYPE
2.5%

Correction Score (%) delle correzioni effettuate sul test set per il MLP Regressor		
MLP Classifier	CatBoost Regressor	Random Forest
15	25	26

Figura 4.7: MAE e categorical mismatch sul training e test set dei regressori

Dalla Figura 4.7 si osserva che CatBoost Regressor ottiene MAE generalmente più bassi rispetto all'MLP Regressor, indicando maggiore precisione e robustezza nella regressione delle variabili continue.
Per quanto riguarda la classificazione della variabile categoriale *NOZZLE_TYPE*, l'MLP ha un mismatch leggermente più alto rispetto a CatBoost.
Questo comportamento suggerisce che l'MLP potrebbe soffrire di overfitting, adattandosi troppo ai dati di training, mentre CatBoost, pur con un piccolo errore, mostra una migliore capacità di generalizzazione.
Per completezza vengono mostrati i grafici relativi al miglior modello della loss del training e validation lungo le iterazioni:

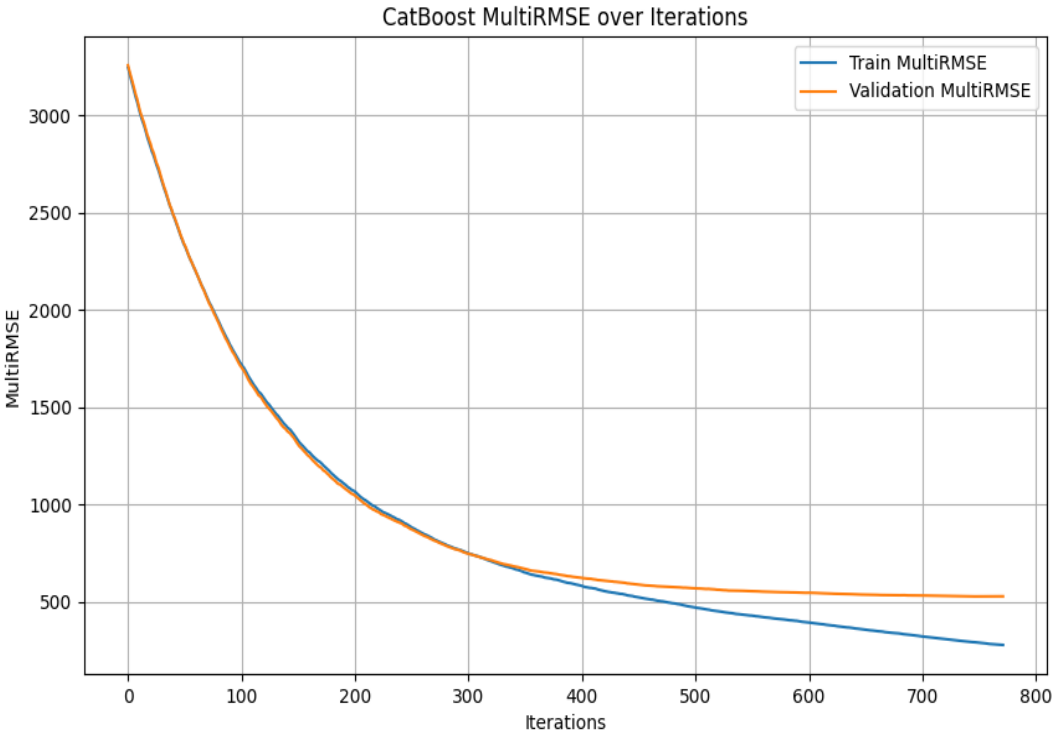


Figura 4.8: training e validation loss per CatBoost Regressor

4.5 Linguaggio e librerie utilizzate

L'architettura descritta e l'analisi dei dati sono state sviluppate utilizzando il linguaggio di programmazione Python, nella versione 3.10.11. Python è ampiamente utilizzato nell'ambito del data science e del machine learning grazie alla sua sintassi semplice e all'ampia disponibilità di librerie specializzate, che facilitano l'implementazione di metodi complessi.
Per la realizzazione e l'addestramento dei modelli predittivi, sono state utilizzate principalmente le seguenti librerie:

- *scikit-learn*:

  per l'implementazione di algoritmi di machine learning classici, la gestione del preprocessing e la valutazione delle performance dei modelli;
- *CatBoost*:

  una libreria di gradient boosting particolarmente efficace nella gestione automatica delle variabili categoriche e nella riduzione dell'overfitting;
- *pandas*:

  per la manipolazione e l'organizzazione dei dataset in strutture tabellari (DataFrame), utili per l'esplorazione e il preprocessing;
- *numpy*:

  per il supporto al calcolo numerico e la gestione efficiente di array e operazioni vettoriali.

Per quanto riguarda la visualizzazione dei dati, sono state impiegate:

- *matplotlib*:

  per la creazione di grafici statici e personalizzabili, utili durante l'analisi esplorativa dei dati;
- *seaborn*:

  una libreria costruita su matplotlib, che consente di creare visualizzazioni statistiche più sofisticate e di facile interpretazione con meno codice.

Queste librerie hanno fornito un ecosistema completo e flessibile, capace di supportare tutte le fasi del progetto: dalla preparazione dei dati, all'analisi, alla modellazione fino alla valutazione finale dei risultati.

Capitolo 5

Conclusioni

5.1 Considerazioni finali

Alla luce dei risultati ottenuti, si può concludere che il CatBoost Regressor ha mostrato, in generale, le performance migliori. Tuttavia, l'accuratezza raggiunta dai classificatori sui dati rigenerati non risulta particolarmente elevata. In particolare, solo un numero limitato di campioni è stato classificato come "buono", il che potrebbe far pensare che i regressori non siano riusciti a correggere efficacemente le configurazioni difettose.
Tuttavia, è importante sottolineare che i classificatori stessi non sono ancora sufficientemente accurati per affermare con certezza che tali configurazioni siano effettivamente scorrette. Inoltre, la strategia di mapping adottata durante l'addestramento prevedeva una correzione minima: l'obiettivo era quello di trasformare una configurazione difettosa in una priva di difetti, con il minimo intervento possibile. Questo ha spesso portato i regressori a generare configurazioni al limite della classificazione positiva, dove il classificatore potrebbe non essere ancora in grado di discriminare con precisione.
Per quanto riguarda l'errore assoluto medio (MAE), si può ritenere che i valori ottenuti siano accettabili. L'azienda fornitrice dei dati ha indicato delle tolleranze sui parametri di processo, entro cui le variazioni non compromettono la qualità del taglio. I risultati ottenuti dai regressori rientrano generalmente in tali margini, suggerendo che le modifiche proposte non peggiorano la qualità del risultato finale.

CONTOUR_SPEED	LASER_POWER	CONTOUR_FOCAL	CONTOUR_GAS PRESSURE	CONTOUR_NOZZLE DISTANCE
300	500	0.5	1	0.1

.

Figura 5.0: Tolleranza dei parametri di taglio

Nonostante i risultati promettenti, sarebbe opportuno raccogliere un maggior numero di dati reali, poiché quelli generati artificialmente tendono a essere troppo simili a quelli già osservati. Da un lato, questi dati sintetici contribuiscono a irrobustire le predizioni e a densificare le correlazioni esistenti; dall'altro, non introducono nuove variabilità nel dataset. Questo limita la capacità del modello di

apprendere relazioni più profonde e complesse. Inoltre, il forte squilibrio tra le variabili categoriche continua a rappresentare un ostacolo, riducendo l'efficacia dell'apprendimento di pattern più dettagliati e generalizzabili.

5.2 Sviluppi futuri

In presenza di una futura raccolta dati più ampia e mirata, sarà possibile rivedere l'approccio con cui viene modellato il problema. In particolare, si potrebbe passare da una strategia discriminativa, che mira a correggere una configurazione non valida in una valida, a una strategia generativa, che apprende direttamente la distribuzione delle configurazioni corrette. In questo contesto, l'impiego di modelli generativi permetterebbe di apprendere la struttura statistica dei dati target e di generare nuove configurazioni valide, analogamente a quanto già fatto nella fase di generazione sintetica, ma in modo più controllato e variabile. Ciò consentirebbe di generare più configurazioni corrette per uno stesso materiale e spessore, ed eventualmente anche per spessori mai visti, attraverso tecniche di interpolazione.

Un possibile sviluppo interessante sarà anche l'integrazione con algoritmi metaeuristici, che potranno essere utilizzati per affinare ulteriormente le soluzioni generate. In questo scenario, un classificatore accurato resterà centrale nell'architettura, svolgendo il ruolo di giudice delle configurazioni proposte, guidando l'ottimizzazione verso regioni del dominio di input che massimizzano la probabilità di ottenere tagli di buona qualità

Bibliografia

[1] Mushtaq, R.T.; Wang, Y.; Rehman, M.; Khan, A.M.; Mia, M.: State-of-the-art and trends in CO2 laser cutting of polymeric materials-a review. Materials. 13, 3839 (2020).

[2] K. Abdel Ghany, M. Newishy: Cutting of 1.2mm thick austenitic stainless steel sheet using pulsed and CW Nd: YAG laser

[3] Laser Cutting Parameters: The Definitive Guide, [Online]. Available at: https://baisonlaser.com/blog/laser-cutting-parameters/#elementor-toc__heading-anchor-1, accessed May, 23, 2025

[4] S. Ürgün, H. Yiğit, S. Fidan, S. Sinmazçelik: *Optimization of Laser Cutting Parameters for PMMA Using Metaheuristic Algorithms*

[5] H. Chmelickova, M. Polak, Nd:YAG pulsed laser cutting of metals, Experimental stress analysis, 39th International Conference, Tabor, Czech, 2001.

[6] Alsaadawy, M., Dewidar, M., Said, A., Maher, I., & Shehabeldeen, T. A. (2023). *A comprehensive review on laser cutting of metals: Process parameters, quality measures, and optimization techniques*. The International Journal of Advanced Manufacturing Technology. Available at:https://link.springer.com/article/10.1007/s00170-023-12768-1,accessed May, 25, 2025

[7] X. S. Yang, *Nature-Inspired Optimization Algorithms*, Elsevier, 2014.

[8] Sean Luke, 2013, Essentials of Metaheuristics, Lulu, second edition, Available at : https://cs.gmu.edu/~sean/book/metaheuristics/Essentials.pdf, accessed June 4, 2025.

[9] Z. Wan, Y. Zhang and H. He, "Variational autoencoder based synthetic data generation for imbalanced learning," *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, USA, 2017, pp. 1-7, doi: 10.1109/SSCI.2017.8285168

[10] Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. Modeling Tabular Data Using Conditional GAN (2019). *Advances in Neural Information Processing Systems, 32*.

[11] *Aditya Sharma, 2021, Variational Autoencoder in Tensorflow, Available at:https://learnopencv.com/variational-autoencoder-in-tensorflow/*, accessed June 4, 2025.

[12] Liu,Fei Tony (7 July 2014).Disponible a:https://sourceforge.net/projects/iforest/ , accessed June 1, 2025.

[13] Danny Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu**.** *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 1996, pp. 226–231.

[14] Campello, R. J. G. B., Moulavi, D., & Sander, J**.** *Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection.* In: *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2015, **10**(1), ArticleNo.5. DOI: 10.1145/2733381

[15] TRUMPF, *INTECH: TRUMPF uses AI to improve laser cutting edges*, 2025. [Online]. Available at: https://www.trumpf.com/en_US/newsroom/global-press-releases/press-release-detail-page/release/intech-trumpf-uses-ai-to-improve-laser-cutting-edges-9214.

[16] ADH Machine Tool, *the role of AI and automation in enhancing fiber laser cutting machines*, 2025. [Online]. Available at: https://shop.adhmt.com/the-role-of-ai-and-automation-in-enhancing-fiber-laser-cutting-machines/, accessed June 4, 2025.

[17] ADH Machine Tool, *Fiber Laser Cutting and Its Role in Reducing Manufacturing Waste*, https://shop.adhmt.com/fiber-laser-cutting-and-its-role-in-reducing-manufacturing-waste/, accessed June 4, 2025.

[18] *Rumelhart, D. E., Hinton, G. E., & Williams, R. J. "Learning Internal Representations by Error Propagation."* In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, MIT Press (1986)

[19] GeeksforGeeks. (n.d.). *Multi-Layer Perceptron learning in TensorFlow*. GeeksforGeeks. Retrieved June 6 2025, from https://www.geeksforgeeks.org/deep-learning/multi-layer-perceptron-learning-in-tensorflow/

[20] Breiman, L. (2001). *Random Forests*. *Machine Learning*, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

[21] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. https://doi.org/10.1007/BF00994018

[22] Friedman, J. H. (2001). *Greedy function approximation: A gradient boosting machine*. Annals of statistics, 1189–1232. DOI: 10.1214/aos/1013203451

[23] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, *CatBoost: Unbiased Boosting with Categorical Features*, in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.