POLITECNICO DI TORINO

Laurea Magistrale in Ingegneria Gestionale

Anno Accademico 2019/2020

Esplorazione degli algoritmi di Classificazione per la manutenzione predittiva nell'era dell'Industry 4.0

Relatore

Tania CERQUITELLI

Candidato

Rebecca CAFASSO

Correlatori

Riccardo CALLA'

Paolo BETHAZ

Luglio 2020

Indice

Elenco delle tabelle

Elenco delle figure

Sommario

L'avvento dell'Industria 4.0 ha portato con sé un radicale cambiamento del settore manifatturiero. Sistemi Cyber-Fisici, Internet of Things, Realtà Aumentata, Intelligenza Artificiale, Additive Manufacturing, Robot Autonomi, sono solo alcune delle tecnologie che hanno rinnovato il modo di produrre trasformando il vecchio concetto di fabbrica in Smart Fabric, una struttura automatizzata e largamente interconnessa dove uomo e macchinari comunicano scambiando informazioni. Protagonisti indiscussi di questa rivoluzione sono senza dubbio i Big Data: enormi flussi di dati raccolti a partire da appositi sensori posizionati su macchinari e robot per monitorarne lo stato ed il funzionamento. Questa possibilità rappresenta un prezioso asset per l'azienda che, se sfruttato adeguatamente, garantisce un vantaggio competitivo notevole. In questo contesto si afferma il Machine Learning che racchiude un insieme di metodologie che permettono a sistemi e macchine di acquisire la capacità di imparare dall'esperienza e migliorarsi in modo autonomo.

Nella presente Tesi ci si concentra su una categoria di algoritmi di apprendimento automatico supervisionato, cioè gli algoritmi di Classificazione che, nel caso specifico, saranno impiegati per realizzare un progetto di manutenzione predittiva. Il caso di studio riguarda il monitoraggio di un robot di un'importante azienda di fama internazionale e leader mondiale nel campo dell'automazione. I dati raccolti dai sensori riguardano valori di corrente consumata dal robot durante numerosi cicli di lavorazione. L'obiettivo è quello di effettuare un'analisi di tipo predittivo: a partire da un set di dati etichettati si vuole realizzare un modello di previsione che associ la corretta label di classe ai nuovi dati in arrivo dai sensori. Ciò è reso possibile dagli algoritmi di Classificazione i quali, sfruttando un approccio supervisionato, apprendono in una prima fase dai dati storici etichettati le relazioni tra variabili e labels in modo tale da poter riconoscere automaticamente la classe di appartenenza dei dati sconosciuti. Non sempre i classificatori sono accurati ed è importante valutare questo aspetto per poter selezionare modelli predittivi affidabili. E' necessario, dunque, effettuare una fase di validazione e, a questo fine, si prenderanno in considerazione le metriche di *Accuratezza*, *Precisione*, *Richiamo* ed $F - measure$ con l'obiettivo di comprendere quale classificatore

risulti effettivamente più performante. Una volta validato il modello, esso sarà in grado di riconoscere i cicli produttivi anomali che saranno etichettati con labels differenti rispetto a quelle che rispecchiano un funzionamento corretto del robot. Nel caso in esame, i malfunzionamenti che si vogliono monitorare riguardano il *belt tensioning*: un errato tensionamento della cinghia compromette il funzionamento del robot il quale consumerà una quantità di corrente che si discosta da quella standard. Per questo motivo, osservando i dati in arrivo dal robot, il modello di predizione deve essere in grado di riconoscere tempestivamente quei cicli affetti da anomalie in modo tale che si possa intervenire sulla cinghia scongiurando un vero e proprio guasto. L'importanza della manutenzione predittiva è proprio questa: si interviene solo nel caso in cui sia strettamente necessario e cioè quando, attraverso il monitoraggio costante dei cicli, arrivino segnali di malfunzionamento. Al giorno d'oggi, l'implementazione di un progetto di manutenzione predittiva è reso possibile dal fatto che, attraverso le tecniche di *Machine Learning*, i macchinari sono in grado di apprendere in modo autonomo quando ci sono anomalie, individuandole sempre più con maggiore affidabilità.

Nel *primo capitolo* si presenterà una panoramica sulle principali tecnologie dell'Industria 4.0. E' interessante sottolineare che, nonostante questo termine sia ormai parte del nostro linguaggio quotidiano, ancora non esiste una vera e propria definizione. Il concetto si spiega analizzando quelle che sono definite le "tecnologie abilitanti", cioè quelle tecnologie protagoniste della Quarta Rivoluzione Industriale.

Nel *secondo capitolo*, dopo aver introdotto il concetto di *Data Mining*, ci si sofferma sul *Knowledge Discovery Process*, cioè sul processo di estrazione della conoscenza dai dati. Riuscire a trasformare i dati in informazioni non è una banalità. Infatti, risulta molto complesso gestire i Big Data, i quali sono eterogenei e possono presentano outliers, rumore e missing values. Non considerare adeguatamente queste problematiche rende impensabile qualsiasi tipo di analisi. Ecco perché, in questo capitolo, ci si sofferma sul processo che occorre seguire per ottenere risultati affidabili che possano essere di supporto al processo decisionale.

Nel *terzo capitolo* si entra più nel dettaglio degli algoritmi utilizzati nella fase di analisi. Inizialmente, sarà fornita un'introduzione generale sul concetto di Classificazione e, successivamente, si presentano gli algoritmi del Decision Tree, Random Forest, K-Nearest Neighbors, Classificazione Bayesiana e Support Vector Machine, individuandone vantaggi e svantaggi.

Nel *quarto capitolo* si presenta il caso di studio e gli esperimenti effettuati. L'obiettivo è quello di individuare il modello predittivo più idoneo e, a questo fine, sono applicati tutti gli algoritmi analizzati nel capitolo precedente. Dopo un'attenta osservazione delle performance di ciascuno di essi, si è deciso di concentrare maggiori attenzioni al Decision Tree e al Random Forest, attraverso i quali si sono portate

avanti analisi più specifiche.

Infine, nel *quinto capitolo* si traggono le conclusioni del lavoro svolto.

Capitolo 1

Industria 4.0

1.1 Premessa

Le grandi rivoluzioni industriali hanno tracciato i confini della società a partire da tempi molto lontani e ancora oggi rimangono un fenomeno attuale, il quale alimenta l'evoluzione e la fame di progresso che da sempre contraddistingue l'essere umano. Ciò che oggi si indica con il termine *Industria 4.0* non è altro che l'ultima di quattro grandi rivoluzioni che hanno coinvolto, e coinvolgono tuttora, la società in tutti i suoi aspetti.

La rivoluzione industriale è un cambiamento radicale e repentino che si sviluppa nel momento in cui le tecnologie e le nuove modalità di concepire il mondo danno il via a trasformazioni dei sistemi economici e delle strutture sociali. Per ogni rivoluzione c'è un elemento preciso che ne rappresenta l'emblema: un'invenzione, una scoperta che impone un cambiamento inarrestabile e che diventa pietra miliare di un'evoluzione tecnologica che parte da tempi molto lontani [1].

Tutto ebbe inizio a partire dalla metà del Settecento, periodo al quale è fatto coincidere l'inizio della *Prima Rivoluzione Industriale* che interessò inizialmente l'Inghilterra, seguita da Francia e Stati Uniti. In questo periodo, la potenza dell'acqua e del vapore meccanizzarono la produzione: nel 1769, lo scozzese James Watt realizzò la macchina a vapore azionata dalla combustione del carbone modificando radicalmente il modo di produrre. In ambito tessile fu introdotta la filatrice meccanica e Henry Cort, ingegnere britannico, brevettò un sistema per ottenere ferro di buona qualità e a costi contenuti.
Il cambiamento coinvolse principalmente il settore tessile e poi quello siderurgico: ci fu un impetuoso sviluppo delle industrie e si diffuse un nuovo modo di lavorare non più a domicilio, ma in fabbrica e con turni di lavoro lunghi e prestabiliti. Nacque così la classe operaria che si trasferì in città dove vi era la più alta concentrazione di fabbriche [2][3].

La *Seconda Rivoluzione Industriale* iniziò intorno al 1870 con l'introduzione dell'elettricità e del petrolio come nuove fonti energetiche, che sostituirono progressivamente il carbone e la macchina a vapore. Coinvolse principalmente Germania, Giappone, Italia e Stati Uniti. I settori trainanti di questa trasformazione furono l'industria chimica, elettrica e metalmeccanica. Nacquero il motore a scoppio, l'automobile, il telefono, la lampadina, l'aeroplano, gli elettrodomestici, ecc. Negli Stati Uniti, nel 1911, si affermò l'organizzazione scientifica del lavoro, ideata dall'ingegnere F. W. Taylor, per rispondere all'esigenza incalzante di avere ritmi di produzione sempre più intensi. Il Taylorismo, che consiste nella scomposizione delle varie fasi del ciclo produttivo in operazioni elementari, ciascuna delle quali affidata ad un gruppo di operai che le porta a termine in modo meccanico e ripetitivo, si intersecò con le innovazioni organizzative introdotte da Henry Ford. In questo modo nacque la famosa catena di montaggio e la produzione di massa che ridussero notevolmente i tempi ed i costi di produzione[4][5].

Il personal computer, invece, è l'emblema della *Terza Rivoluzione Industriale*, nota anche come rivoluzione *"informatica"* o *"digitale"*. Essa ebbe inizio intorno agli anni Settanta del '900 con la nascita dell'informatica che contribuì a incrementare ulteriormente i livelli di automazione delle fabbriche.
Lo sviluppo di nuove tecnologie divenne il settore centrale per la crescita economica: negli anni Sessanta si diffusero i dispositivi di elaborazione ad alto livello (mainframe computer), tra gli anni Settanta ed Ottanta comparvero i primi personal computer e si arrivò, in pochi anni, alla diffusione della rete Internet. L'informatica e la telematica portarono lentamente alla sostituzione delle lavorazioni basate sull'utilizzo di manodopera con l'introduzione di macchine automatizzate. La diffusione della robotica e della motorizzazione portarono a un cambiamento radicale del modo di lavorare e di vivere la quotidianità [1].

La rivoluzione informatica, la diffusione dell'elettronica, le nuove scoperte tecnologiche hanno incrementato sempre più i livelli di automazione fino ad arrivare a quella che oggi si considera la *Quarta Rivoluzione Industriale*, chiamata *Industria 4.0*. Molti studiosi affermano che tale rivoluzione è di una portata, velocità e complessità tale che l'uomo non ha mai assistito a nulla di simile. Basti pensare alla facilità con la quale milioni di persone ogni giorno sono tra loro interconnesse grazie a PC, smartphone e tablet a velocità del tutto trascurabili. Oppure si può pensare all'intelligenza artificiale, alla robotica, ai sistemi cyber-fisici, all'Internet delle Cose, alla nanotecnologia, alla biotecnologia: tutti termini che sono entrati a far parte della vita di ogni giorno e che l'hanno trasformata profondamente. D'altronde, ogni rivoluzione ha sempre portato cambiamenti imprevedibili e inimmaginabili, ma oggi si assiste ad un fenomeno davvero eccezionale del quale è interessante capirne l'impatto e le potenzialità [1].

1.2 Industria 4.0: origini e definizioni

Alcuni accademici sono convinti che sia inadeguato utilizzare il concetto di Quarta Rivoluzione Industriale, ma sia meglio considerare questo fenomeno come una conseguenza della rivoluzione digitale. In realtà, come afferma Klaus Schwab nel suo libro *"La Quarta rivoluzione industriale"*, ci sono tre elementi tali per cui risulterebbe corretto affermare che si sta attraversando una rivoluzione a sé:

- **Velocità**: questa rivoluzione, al contrario delle tre precedenti che si sono sviluppate con velocità lineare, si sta diffondendo in modo esponenziale proprio grazie alla realtà interconnessa e alla diffusione accelerata di nuove tecnologie sempre più performanti.

- **Portata e intensità**: le trasformazioni in corso si basano sull'eredità ricevuta dalla rivoluzione digitale, ma danno luogo a cambiamenti radicali a livello economico, aziendale, sociale e individuale.

- **Impatto sui sistemi**: la Quarta Rivoluzione Industriale opera una trasformazione non solo su aziende e settori, ma anche sui Paesi e nelle società in generale[1].

Il termine *Industry 4.0* fu utilizzato per la prima volta in Germania nel 2011 alla Fiera di Hannover. Un gruppo di lavoro annunciò un progetto, chiamato *"Industrie 4.0"*, per lo sviluppo del settore manifatturiero tedesco con l'obiettivo di riportare l'industria del Paese a ricoprire nuovamente il ruolo di leader. Nel 2013 si concretizzò il progetto per l'Industria 4.0 con la diffusione di un report, ad opera di *"Industrie 4.0 Working Group"*, che conteneva la previsione di alcuni investimenti in infrastrutture, scuole, sistemi energetici, ecc. Tale modello si è poi diffuso ispirando diversi paesi europei e diventando di interesse globale [6].

Da uno studio realizzato nel 2015 ad opera di Mario Hermann, Tobias Pentek e Boris Otto , intitolato *"Design Principles for Industrie 4.0 Scenarios: A Literature Review"* [7], emergono due aspetti interessanti: tale rivoluzione, al contrario delle tre precedenti, è stata annunciata a priori anziché ex-post, inoltre, nonostante l'interesse riscosso a livello globale, non esiste ancora una definizione generale e univoca del termine. Questo aspetto si traduce anche in una difficoltà da parte delle aziende a comprendere ed implementare le linee guida della Quarta Rivoluzione Industriale. Gli stessi promotori tedeschi del progetto *Industrie 4.0* non ne forniscono una chiara definizione, ma descrivono solo le tecnologie alla base dell'idea (ad esempio i Sistemi Cyber-fisici, le macchine intelligenti, le fabbriche intelligenti, i prodotti intelligenti, i sistemi di produzione collegati con la rete). Mario Hermann, Tobias Pentek e Boris Otto, nella loro ricerca, riportano anche la definizione introdotta da

General Electric, che nel 2013 parlò di *"Industrial Internet"*: *"the integration of complex physical machinery and devices with networked sensors and software, used to predict, control and plan for better business and societal outcomes"* (Industrial Internet Consortium, 2013). L'Industry 4.0 è dunque definita come *"l'integrazione di macchinari e dispositivi fisici complessi con sensori e software collegati in rete, utilizzati per prevedere, controllare e pianificare al meglio i risultati aziendali e sociali"*.

L'obiettivo di questa interessante ricerca è quello di trovare una definizione di Quarta Rivoluzione Industriale basandosi su ciò che c'è in letteratura e, a tal fine, gli autori individuano i quattro componenti base che sono più frequentemente associati al concetto di Industria 4.0. Essi sono i seguenti:

- Sistemi cyber-fisici
- Internet of Things
- Smart Factory
- Internet of Services

1.2.1 Sistemi cyber-fisici

Il termine *Cyber Physical System*, abbreviato anche con la sigla *CPS* è stato introdotto dalla National Science Foundation nel 2006 da Helen Gill per descrivere sistemi informatici nei quali la parte cyber e la parte fisica interagiscono e si influenzano l'una con l'altra [8]. Il termine, però, deriva dal più antico concetto di Cibernetica, disciplina che negli anni Sessanta studiava la possibilità di creare macchine con le stesse capacità del cervello umano. *Cyber* deriva dal greco *kybernetes* che letteralmente si traduce con *"pilota di una nave"* e fa riferimento al concetto più esteso di *"governare"* che è quanto fatto da un software in merito all'hardware sul quale è installato. Successivamente il termine cyber è stato utilizzato per indicare il concetto di cyberspazio, cioè per fare riferimento a tutto ciò che riguarda il mondo di Internet e della rete virtuale [9].

In realtà, le definizioni in letteratura di CPS sono molteplici e fanno riferimento alle diverse applicazioni che tali sistemi possono avere. In questo caso, si considera il CPS come una delle fondamentali tecnologie abilitanti della Quarta Rivoluzione Industriale e quindi, sotto questo aspetto, può essere considerato come un sistema autonomo, intercomunicante e intelligente in grado di favorire l'integrazione tra soggetti diversi e distanti nello spazio. È un sistema informatico in grado di interagire con il sistema fisico in cui opera [10]. Questa definizione mette in luce alcuni aspetti fondamentali: primo fra tutti l'*interconnessione*, cioè il fatto che ci

siano oggetti interconnessi che generano dati riducendo le asimmetrie informative, in secondo luogo la *comunicazione*, ovvero la possibilità di scambiare i dati in tempo reale trasformandoli in informazioni preziose e di valore aggiunto. Per comprendere meglio il significato dei CPS, occorre però introdurre il concetto del *"Digital Twin"*, ovvero dell'immagine virtuale: tutti i componenti che caratterizzano un sistema di produzione non esistono solo così come li percepiamo con i cinque sensi, ma essi hanno anche un'immagine virtuale che rispecchia quella reale e fornisce diverse informazioni aggiuntive. In un CPS, quindi, gli oggetti fisici sono

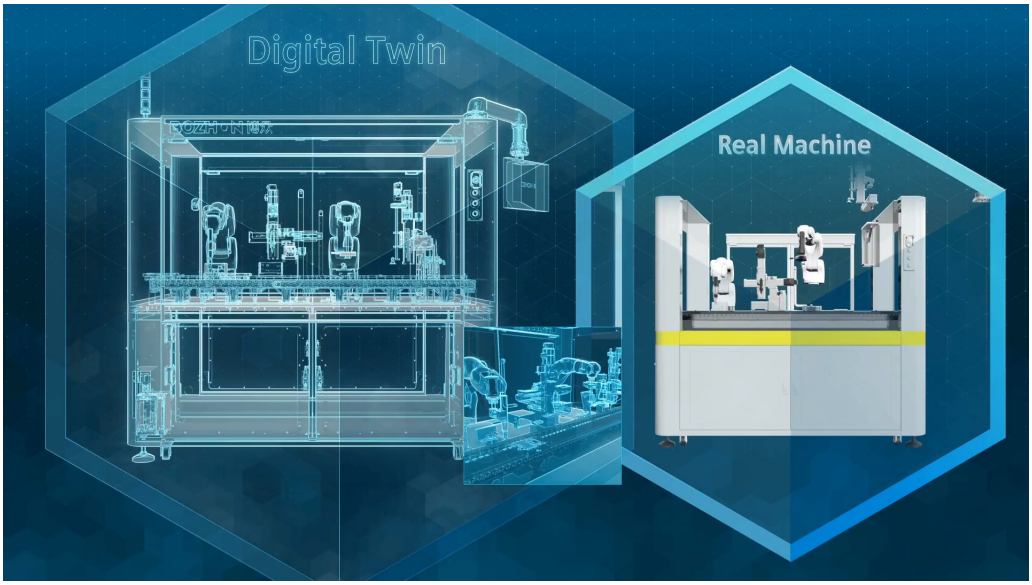


Figura 1.1: Digital Twin

affiancati dalla rispettiva rappresentazione digitale e, sulla base delle informazioni da essa fornite, il componente fisico è in grado di prendere decisioni in modo autonomo. Dunque, volendo definire con più precisione il sistema Cyber-fisico, si può affermare che esso è un sistema in cui gli oggetti fisici sono affiancati dalla propria rappresentazione nel mondo digitale, sono integrati con elementi dotati di capacità di calcolo, memorizzazione e comunicazione e sono collegati in rete tra loro[10]. In parole semplici, un sistema cyber-fisico rappresenta la fusione tra mondo fisico e mondo virtuale (Kagermann, 2014). Il termine *fisico* fa riferimento all'oggetto reale così come è percepito con i cinque sensi, mentre la componente *cyber* fa riferimento al gemello digitale. Il sistema cyber-fisico racchiude tre abilità comunemente riconosciute come le *"tre C"* che permettono di espandere le capacità del mondo fisico:

- Capacità computazionale
- Comunicazione

- Capacità di controllo

Le prime due competenze, ovvero quelle computazionali e comunicative, fanno riferimento alla componente *cyber*, mentre la componente legata alla capacità di controllo è la componente *fisica* [11]. Questo sistema è in grado di acquisire ed elaborare dati, effettuare aggregazioni e calcoli ed infine è di supporto al processo decisionale. Infatti, un sistema cyber-fisico necessita di tre elementi fondamentali:

- **Sensori**: il CPS riesce a rilevare la situazione operativa all'interno dell'ambiente in cui si trova e può fornire informazioni circa il suo stato, la sua posizione o la sua tipologia.

- **Attuatori**: sono il mezzo attraverso il quale il CPS svolge le azioni, cioè mette in pratica le decisioni correttive per ottimizzare il processo.

- **Intelligenza decentralizzata**: definisce le attività che gli attuatori devono mettere in pratica. Essa prende decisioni sulla base delle informazioni fornite dai sensori e dai CPS e le comunica agli attuatori e ad altri CPS.

Grazie a questi elementi i sistemi cyber-fisici sono in grado di valutare determinate situazioni, prendere decisioni in modo autonomo ed eventualmente dialogare con altri sistemi [10]. Questo meccanismo è rivoluzionario: l'intelligenza decentrata integra il vecchio sistema decisionale gerarchico/verticale. Grazie all'immagine virtuale e all'intelligenza decentrata i CPS sono in grado di valutare le situazioni operative in modo autonomo e prendere decisioni comunicandole ad altri sistemi. Al contrario, il vecchio processo decisionale gerarchico, prevedeva che i sensori rilevassero lo stato di un determinato processo e comunicassero tutte le informazioni all'unità di controllo centrale. L'unità di controllo, poi, analizzava lo stato effettivo del processo, prendeva decisioni che erano messe in pratica manualmente oppure mediante attuatori. Con i CPS non si vuole eliminare del tutto questo processo verticale, ma semplicemente integrarlo. A titolo d'esempio si può considerare una linea di confezionamento: un sensore ottico può autonomamente rilevare la necessità di cambiare il tipo di confezionamento a causa dell'arrivo di un nuovo prodotto con un formato differente. Il sensore comunica direttamente le nuove coordinate ai sistemi di posizionamento e ai vari utensili. L'unità di controllo, così come gli operatori, sono informati circa i cambiamenti messi in atto [12].

L'architettura di un cyber-physical system può essere rappresentata su 5 livelli, per questo motivo è conosciuta come la *"5C level architecture"*[13].

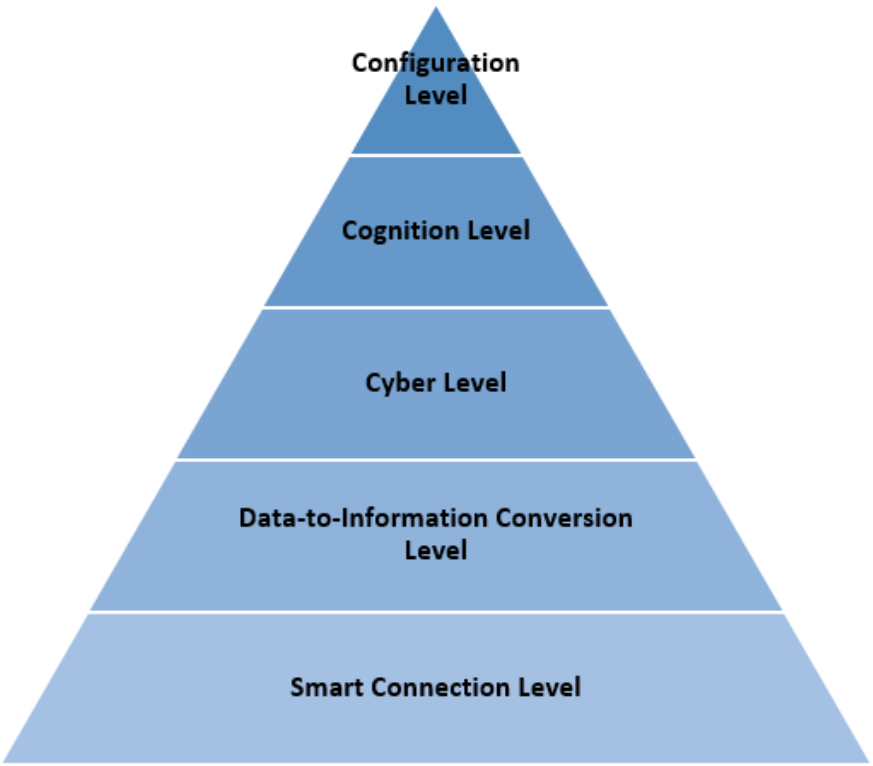


Figura 1.2: 5D architecture [13]

- **Smart connection**: i sensori riescono ad acquisire dati in tempo reale e a comunicarli grazie ad appositi protocolli. I dati possono provenire anche da sistemi informativi aziendali. In questa fase si devono considerare alcuni aspetti importanti: i dati possono essere di vario tipo e occorre scegliere il metodo più adatto per gestirne la procedura di acquisizione e trasferimento al server centrale.

- **Data-to-information conversion**: i dati possono essere aggregati e si possono acquisire informazioni ulteriori. A tal fine ci sono diversi strumenti disponibili per estrarre contenuto informativo.

- **Cyber Level**: questo livello rappresenta il centro informativo perché tutte le informazioni sono trasmesse ad ogni macchina, creando così una rete interconnessa. Esse possono in questo modo confrontare il loro stato e le loro capacità. Inoltre, si possono comparare i dati con le informazioni del passato per prevedere il comportamento futuro della macchina.

- **Cognition**: in questo livello si genera una conoscenza più approfondita del sistema e si stabiliscono le azioni correttive che sono di supporto al processo

decisionale. L'utente che prende decisioni consulta grafici attraverso i quali viene trasmesso in modo più intuitivo ed interpretabile il contenuto delle informazioni.

- **Configuration**: la realtà *cyber* fornisce a quella *fisica* un feedback e, sulla base di ciò, sono messe in atto azioni correttive prese al livello precedente.

Per quanto invece riguarda i benefici dei Cyber Physical System, essi sono identificati in sei clusters principali che ne rappresentano le potenzialità [10]:

- *"New data driven services and business models"*: sono i benefici che riguardano l'ambito manageriale dell'azienda. I sistemi cyber-fisici permettono all'azienda di realizzare nuovi modelli di business che le consentono di prestare più attenzione ai bisogni del cliente, offrendo servizi altamente personalizzati. Inoltre, la generazione attiva e in tempo reale delle informazioni, permette all'azienda di essere più flessibile e reattiva di fronte ai cambiamenti del mercato.

- *"Data-based improved products"*: i CPS permettono la realizzazione di un prodotto intelligente, fortemente digitalizzato ed in grado di comunicare dentro e fuori la fabbrica. Inoltre, lo smart product permette all'azienda che lo realizza di ottenere feedback in tempo reale circa il suo utilizzo da parte dei clienti. Tutte queste potenzialità permettono sempre più una personalizzazione massiccia, un prodotto creato su misura e per questo si parla di "personalizzazione di massa".

- *"Closed-loop manufacturing"*: sono tutti quei benefici che non riguardano solamente la fabbrica entro i suoi confini, ma tutti gli altri stakeholders come clienti e fornitori.

- *"Cyberized plant/ Plug Produce"*: sono l'insieme dei benefici a livello di unità produttiva, infatti i CPS garantiscono prestazioni migliori grazie alla forte digitalizzazione. In particolare, abilitano scenari di piena flessibilità e riconfigurabilità del sistema produttivo.

- *"Next step production efficiency"*: sono i benefici riguardanti l'ottimizzazione dell'utilizzo degli asset aziendali per avere una produzione efficiente, precisa, veloce e sostenibile.

- *"Digital ergonomics"*: sono i benefici relativi al capitale umano dell'azienda. I CPS velocizzano il processo di trasferimento delle informazioni, migliorando le relazioni uomo-macchina e la worker experience.

1.2.2 Internet of Things

Un altro importante componente tecnologico dell'Industry 4.0 è l'*Internet of Things*, anche abbreviato con la sigla *IoT*. Come si può dedurre dal nome, l'Internet delle Cose fa riferimento a tutto l'insieme di oggetti che circondano l'uomo e che sono connessi alla rete: basti pensare alle auto, ad alcuni elettrodomestici, agli impianti di climatizzazione, alle telecamere, alle lampadine, ad alcuni accessori di arredamento e molto altro ancora [14]. Gli *"smart objects"* fanno ormai parte della vita quotidiana. Ma qual è il vero vantaggio di avere oggetti connessi alla rete? L'internet of Things rappresenta la possibilità che ogni oggetto connesso ha di scambiare informazioni in modo autonomo con altri oggetti circostanti e addirittura modificare il proprio comportamento in presenza di determinati input. Da qui si può definire l'Internet delle Cose come l'insieme degli oggetti fisici che sono in grado di comunicare con l'ambiente esterno e altri oggetti, trasmettendo dati sul proprio stato o sull'ambiente in cui si trovano [15]. In letteratura si trovano diverse definizioni, tra cui quella di Luigi Atzori, Atonio Iera e Giacomo Morabito: il concetto base dell'IoT è la presenza pervasiva di oggetti, come RFID, tag, sensori, attuatori, telefoni, i quali, attraverso schemi di indirizzamento univoci, sono in grado di interagire tra loro e collaborare con oggetti a loro vicini e raggiungere obiettivi comuni [16]. Ecco perché l'oggetto viene definito "intelligente" ed avrà le seguenti funzionalità [17]:

- Identificazione: l'oggetto sarà dotato di un identificativo nel mondo digitale (indirizzo IP che ne consente l'identificazione univoca in rete)
- Localizzazione
- Diagnosi di stato
- Interazione con l'ambiente esterno
- Elaborazione di dati
- Connessione

Si stima che nel 2010 il numero di smart objects abbia superato il numero di abitanti del pianeta e nel 2020 raggiungerà la soglia dei 50 miliardi [18]. Inoltre, IDC (*"International Data Corporation"*) nel 2019 ha previsto che la spesa globale in progetti legati all'IoT raggiungerà 1,2 trilioni di dollari nel 2022, in particolare nel settore del manufaturing e dei trasporti [19].

Il concetto di IoT nasce però parecchi anni fa, intorno al 1982, quando vennero installati per la prima volta alcuni sensori su di un distributore di bevande nella Carnegie Mellon University per analizzarne il funzionamento. Successivamente, il concetto fu ripreso in modo più rigoroso da Reza Raji nel 1994, che accennò

Figura 1.3: Internet of Things

alla possibilità di connettere alla rete svariati prodotti, dagli elettrodomestici agli elementi di una fabbrica [20].

"[moving] small packets of data to a large set of nodes, so as to integrate and automate everything from home appliances to entire factories" (Reza Raji).

È proprio quando l'Internet of Things incontra la fabbrica che nasce l'*Industrial Internet of Things* o (*IIoT*) . Con questo termine si fa riferimento a tutte quelle tecnologie e sensori che permettono agli elementi della fabbrica, dai prodotti finiti ai macchinari, di comunicare tra loro, trasferendo informazioni e dati importanti attraverso la rete. Queste tecnologie danno luogo ad un nuovo modo di comunicare tra macchine, impianti e persone in tempo reale, decentralizzando il processo decisionale e dando vita ad una struttura flessibile ed efficiente.

L'Industrial Internet of Things possiede numerosi ambiti di applicazione, per esempio la produzione: attraverso il monitoraggio delle varie fasi del processo produttivo si possono prevedere le criticità e ridurre i fermi macchina, con conseguente miglioramento in termini di efficienza. In fabbrica, avere macchinari o componenti dotati di sensori che raccolgono dati e che li trasmettono è di fondamentale importanza per l'ottimizzazione degli interventi di manutenzione. Ma non solo, in ambito Supply Chain, la tecnologia IIoT è utile nel controllo dei fabbisogni, nel monitoraggio delle scorte e nella conseguente ordinazione di materiale necessario per ridurre sprechi, scarti e per alleggerire il lavoro del capitale umano [21]. L'IIoT diventa un elemento essenziale nella fabbrica 4.0 e comporta una serie di ripercussioni positive

che si possono sintetizzare nel seguente modo [22]:

- **Produzione intelligente**: i sensori e la connessione di cui possono essere provvisti i macchinari in una fabbrica, consente ai sistemi manifatturieri di comunicare e dar vita ad una fabbrica connessa. I dati raccolti possono essere immediatamente consultati dal personale che può avere un'istantanea sul numero di risorse utilizzate, sulle unità prodotte, sulle prestazioni, su eventuali guasti e anomalie e molto altro ancora. Questo comporta ovviamente un miglioramento dell'efficienza dell'azienda.

- **Gestione energetica**: attraverso le tecnologie IIoT, si possono raccogliere dati e avere informazioni puntuali su quelli che sono i consumi energetici di determinati reparti o dell'azienda stessa. Ciò può aiutare ad individuare i macchinari responsabili di eccessivi consumi e questa consapevolezza può tradursi in risparmio energetico, in ricerca di risorse rinnovabili, ecc. Una gestione efficiente dei consumi comporta una riduzione dei costi significativa per l'azienda.

- **Manutenzione predittiva**: come precedentemente accennato, il monitoraggio di un impianto può essere utile al fine di implementare un programma di manutenzione che aiuti l'operatore a comprendere in anticipo l'arrivo di un possibile guasto per intervenire tempestivamente riducendo le inefficienze e i tempi di fermo macchina.

- **Controllo remoto**: se i dispositivi, i prodotti e i macchinari sono connessi alla rete è possibile controllarli da qualsiasi posizione, anche se ci si trova fisicamente lontani.

- **Decisioni intelligenti**: avere a disposizione più dati e informazioni rende possibile l'analisi dei Big Data che è di fondamentale importanza nel supporto alle decisioni. Inoltre, i dati possono aiutare a prevedere situazioni future e cambiamenti imminenti. La capacità di prendere decisioni intelligenti e di fare previsioni sulla base dei dati aiuta l'azienda a rimanere competitiva sul mercato oltre che a ricercare nuove opportunità di business.

1.2.3 Internet of Services

Con il termine *Internet of Services* , si fa riferimento all'utilizzo di Internet come strumento di erogazione di un servizio da parte di un fornitore. Alcuni fornitori hanno rivisto il loro modello di business e hanno trasformato la vendita di prodotto in una combinazione tra prodotto e servizio da erogare con un flusso di entrate a lungo termine.

Rolls Royce non vende un motore aereo, ma ore di volo effettivamente compiute (*"Power by the hour"*) includendo anche manutenzione, riparazione e revisione del motore stesso. L'azienda può connettersi direttamente al motore del cliente per monitorarne lo stato, intervenendo eventualmente da remoto in caso di necessità. Inoltre, raccoglie dati per verificare le prestazioni dei suoi motori e migliorarle. La connessione diretta con i motori abilita una comunicazione tra prodotto, produttore e cliente.

Tesla vende veicoli provvisti di sensori e con hardware e software che possono essere aggiornati. Il cliente paga per gli aggiornamenti e questo comporta un flusso di entrate a lungo termine.

In definitiva, l'idea alla base dell'Internet of Services è quella di trovare nuove modalità di creazione di valore utilizzando Internet come strumento di erogazione di servizi. Infatti, uno degli elementi centrali dell'Industria 4.0 è l'orientamento al servizio, cioè il fatto che la produzione sarà sempre più *"service-oriented"* ed il produttore utilizza il prodotto come piattaforma per fornire servizi aggiuntivi. Si parla, infatti, di *"servitizzazione"* o *"service transformation"* per indicare la tendenza a vendere un prodotto offrendo una soluzione che coinvolge cliente e fornitore in una relazione che dura nel tempo. La svolta risiede proprio nelle tecnologie come IoT e cloud (di cui si parlerà successivamente) che permettono la connessione tra prodotti. La possibilità di ricevere informazioni di ritorno sull'utilizzo di un determinato prodotto permette al fornitore/produttore di conoscere lo stato di funzionamento e le condizioni di utilizzo che gli consentono di elaborare specifici interventi di manutenzione. In questo modo nascono nuovi tipi di contratto che si focalizzano sull'utilizzo del prodotto (pay-per-use, pay-per-availability, pay-per-performance) [23].

Questo nuovo modello di business che si sta affermando genera vantaggi sia per chi fornisce il servizio, sia per chi lo acquista. Nel primo caso, il fornitore può contare sulla fidelizzazione del cliente: ad esempio, attraverso la manutenzione predittiva, il fornitore interviene sul macchinario prima ancora che esso si rompa e ciò è reso possibile dal fatto che egli ha accesso ai dati sul funzionamento del macchinario e sull'utilizzo che il cliente ne fa. Inoltre, il produttore assisterà ad un notevole incremento della conoscenza che rappresenta un patrimonio ottenuto gratuitamente dalle analisi effettuate sui dati del cliente. Infine, questa tipologia di business permette al produttore di avere entrate protratte nel tempo per tutta la durata del servizio e non più di avere un'entrata unica legata alla vendita. Per quanto riguarda l'acquirente del servizio, esso potrà contare sullo spostamento di un Capex in un Opex e potrà usufruire di servizi che lo preservano da fermi produttivi dovuti a guasti improvvisi delle macchine [24].

1.2.4 Smart Factory

Spesso il termine Industria 4.0 è associato al concetto di *Smart Factory*, cioè la fabbrica intelligente. Essa è figlia delle grandi innovazioni introdotte dalla Quarta Rivoluzione Industriale: sistemi automatizzati e intelligenti che possono comunicare, interagire ed operare a stretto contatto con il mondo reale e che rendono la fabbrica sempre più autonoma ed efficiente. La Smart Factory, infatti, è il frutto dell'integrazione tra Internet of Things e i sistemi cyber-fisici. Nella fabbrica intelligente emergono nuovi modi di organizzare i processi produttivi collegando persone, prodotti, dispositivi, macchinari e dati. In letteratura esistono numerose definizioni di Smart Factory, per esempio Radziwon la definisce come: *"una soluzione produttiva che favorisce processi flessibili e adattivi per risolvere i problemi derivanti dalla complessità crescente attraverso un impianto di produzione dinamico e in rapida evoluzione. Da un lato, questa soluzione è correlata all'automazione, intesa come combinazione di software, hardware e/o meccanica, che dovrebbe portare all'ottimizzazione della produzione con la conseguente riduzione delle risorse impiegate. Dall'altro, rappresenta una prospettiva di collaborazione tra i diversi partner industriali e non, dove l'intelligenza deriva da un'organizzazione dinamica e partecipativa".* Da questa definizione si deduce che la Smart Factory non è semplicemente una fabbrica con processi produttivi automatizzati, ma l'elemento centrale è il suo essere smart, in riferimento alla capacità di costante interazione tra i vari elementi del sistema produttivo grazie alle varie tecnologie come sensori e smart objects. Attraverso l'integrazione con le nuove tecnologie abilitanti (IoT, CPS, stampa 3D, Big Data, ecc), la Smart Factory incrementa la propria efficienza, la competitività e soprattutto la sua capacità di soddisfare le esigenze diversificate dei clienti puntando sull'interconnessione e cooperazione tra persone e macchinari [25]. Il concetto di Smart Fabric, secondo gli esperti di Boston Consulting Group e McKinsey [6], si articola su tre livelli:

- **Smart production**: la produzione intelligente fa riferimento al fatto che le nuove tecnologie produttive sono totalmente interconnesse e vi è una stretta collaborazione tra macchine, lavoratori e strumenti.
- **Smart services**: vi è una forte integrazione non solo tra sistemi ma anche tra aziende che possono interagire in modo più semplice ed efficace.
- **Smart energy**: c'è una forte attenzione ai consumi energetici ed un'accentuata propensione ad una produzione eco sostenibile che riduca gli sprechi.

Tutte queste caratteristiche comportano un cambiamento nel paradigma produttivo che permette al settore manifatturiero delle economie più consolidate di riacquistare competitività. Infatti, nei paesi occidentali non esiste più la produzione di massa

a basso contenuto di valore aggiunto (che è stata attratta dai paesi low cost), ma è stata poco per volta sostituita da una nuova manifattura innovativa che supporta soprattutto quei settori in cui c'è necessità di forte specializzazione e differenziazione. La fabbrica intelligente, grazie alle tecnologie impiegate, permette un sistema di produzione completamente flessibile che si adatta alle esigenze di ciascun cliente, creando prodotti non solo di alta qualità ma anche su misura. Inoltre, è massimizzata l'efficienza del sistema di produzione e c'è una forte attenzione all'intero ciclo produttivo, a partire dalla progettazione, fino alla fidelizzazione del cliente. In questo senso, la Smart Factory, non è solo attività di produzione, *"ma un circuito di attività immateriali come l'ideazione, la ricerca e sviluppo, il design, l'innovazione, la modellizzazione e programmazione della produzione, la logistica, la comunicazione, la gestione degli ordini nelle filiere globali, i marchi e i significati connessi, la commercializzazione, il rapporto sempre più interattivo col mondo della distribuzione e del consumo"* (Rullani). Dunque, non siamo solo di fronte ad una semplice trasformazione tecnologica della fabbrica, ma bensì ad un sistema produttivo complesso caratterizzato da automazione e digitalizzazione che si integrano con il lavoro umano per creare prodotti su misura per il cliente finale [6].

1.2.5 Definizione

Sulla base dei quattro elementi analizzati, gli autori Mario Hermann, Tobias Pentek e Boris Otto forniscono la seguente definizione di Industria 4.0: *"Industrie 4.0 is a collective term for technologies and concepts of value chain organization. Within the modular structured Smart Factories of Industrie 4.0, CPS monitor physical processes, create a virtual copy of the physical world and make decentralized decisions. Over the IoT, CPS communicate and cooperate with each other and humans in real time. Via the IoS, both internal and crossorganizational services are offered and utilized by participants of the value chain"*[7].

Ci si trova di fronte ad un profondo processo di innovazione digitale e protagonista di questo cambiamento è certamente il settore manifatturiero. Infatti, la Quarta Rivoluzione Industriale incarna un processo in cui lo sviluppo di nuove tecnologie e l'automazione permettono la creazione di una struttura largamente interconnessa che coinvolge persone, strumenti e macchinari. La fabbrica intelligente implementa un nuovo tipo di manifattura, innovativa e vicina ai bisogni del cliente, potendo contare su un sistema di produzione ampiamente flessibile. Risponde quindi alle esigenze del mercato che richiede soluzioni sempre più personalizzate e segna profondamente il passaggio da una produzione di massa ad una produzione che mira alla personalizzazione di massa. Inoltre, le macchine 4.0 sono in grado di

auto diagnosticare problemi e correggerli grazie all'IoT ed ai sistemi cyber-fisici, garantendo efficienza [25]. Grazie a questi fattori, nascono nuovi modelli di business che prevedono rapporti contrattuali tra cliente e fornitore che hanno per oggetto la vendita di un prodotto come servizio protratto nel tempo.

1.3 Principi dell'Industria 4.0

Prendendo sempre come riferimento lo studio [7], a partire dai quattro elementi individuati come componenti principali dell' Industria 4.0 (IoT, IoS, Smart Fabric, CPS) sono dedotti i suoi principi chiave:

- **Interoperabilità**: è la capacità di scambiare informazioni tra due o più componenti. È un fattore di fondamentale importanza nel contesto dell'Industria 4.0 in cui macchine, processi e persone sono in grado di comunicare grazie ai CPS, all'IoT e all'IoS. Questi strumenti permettono lo scambio di dati e di informazioni utili in tempo reale, migliorando la collaborazione all'interno della Smart Factory. L'interoperabilità è però possibile solo quando si utilizza uno standard comune, un linguaggio standardizzato per la comunicazione tra diversi componenti.

- **Virtualizzazione**: rappresenta la forte connessione tra mondo fisico e mondo virtuale. La virtualizzazione permette di creare una copia della Smart Factory nel mondo digitale grazie ai sensori che monitorano i vari processi fisici, ai dati da essi acquisiti, ai modelli virtuali e di simulazione.

- **Decentralizzazione**: il processo decisionale è decentralizzato in quanto i CPS riescono a prendere decisioni in modo autonomo, possono comunicarle ad altri sistemi interconnessi e collaborare. In questo modo, il capitale umano dell'aziende non viene coinvolto in ulteriori attività poiché queste sono gestite direttamente dalle macchine digitalizzate, che hanno accesso a dati in tempo reale e in base ai quali riescono a prendere decisioni rapide ed efficienti. Questo aspetto comporta un'elevata reattività del processo produttivo.

- **Real time capability**: è la capacità di raccogliere ed elaborare i dati in tempo reale per poter prendere decisioni.

- **Orientamento al servizio**: il prodotto non è più solo un oggetto fisico tangibile, ma è una combinazione di prodotto e servizi. Infatti, vi è l'inclusione nella vendita non solo di un oggetto fisico, ma anche di servizi ausiliari, trasporto, manutenzione, aggiornamento software, ecc. In questo modo il fornitore vende un ibrido tra mondo fisico e virtuale (*Product as a service*).

- **Modularità**: un sistema è modulare quando è in grado di adattarsi velocemente e in modo flessibile ai cambiamenti della domanda, ad esempio in seguito al mutamento delle caratteristiche del prodotto. Nel contesto dell'Industry 4.0 non è più sufficiente la produzione automatizzata, ma è indispensabile poter garantire la cosiddetta customizzazione di massa e quindi riconfigurare le caratteristiche del prodotto sulla base delle esigenze del cliente in tempi celeri e senza sprechi.

1.4 I nove pilastri dell'Industria 4.0

Come precedentemente accennato, sono molti gli studi portati avanti per definire la Quarta Rivoluzione Industriale. Oltre ai quattro aspetti appena considerati, ci sono altre tecnologie che contribuiscono a definire ulteriormente il concetto di Industry 4.0. Nel Report *"Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries"* (2015) [26], la famosa azienda di consulenza americana Boston Consulting Group individua nove tecnologie abilitanti che rappresentano gli elementi salienti dell'Industry 4.0, cioè gli ingredienti più importanti della trasformazione. Non tutte queste tecnologie sono "nuove", ma alcune sono vecchie conoscenze che però non erano mai entrate a far parte dell'impresa e della produzione, mentre oggi, grazie all'interconnessione e alla collaborazione tra sistemi, stanno diventando vere e proprie protagoniste nel settore manifatturiero.

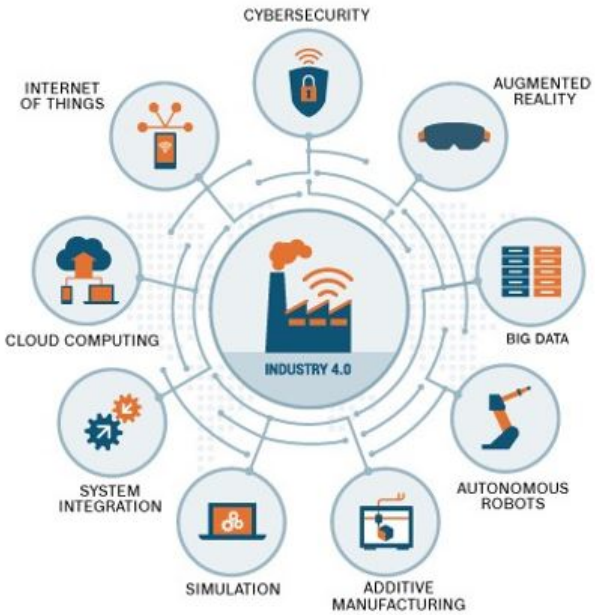


Figura 1.4: Tecnologie abilitanti Industria 4.0

I nove pilastri individuati da Boston Consulting Group sono i seguenti:

1. Internet of Things
2. Autonomous Robot
3. Cloud Computing
4. Augmented Reality
5. Big Data & Analytics
6. Cyber Security
7. Additive Manufacturing
8. Simulation
9. Horizontal and Vertical System Integration

Del primo tra questi si è già discusso precedentemente, mentre di seguito si presenta una panoramica delle restanti tecnologie.

1.4.1 Big Data and Analytics

Con il termine *Big Data and Analytics* si vuole indicare una grande mole di dati eterogenei che vengono trasmessi o ricevuti ad una velocità e ad una frequenza tali per cui non possono essere gestiti attraverso i tipici database, ma bensì sono richieste nuove tecnologie e metodologie per manipolarli in tempi ragionevoli.

Il termine Big Data è stato introdotto dalla Nasa nel 1997 al fine di indicare enormi quantità di dati difficili da archiviare su un semplice Personal Computer e difficili da analizzare [27]. Infatti, nonostante le varie definizioni presenti in letteratura, le caratteristiche peculiari e sempre associate al termine Big Data sono la difficoltà di gestione e manipolazione di un set troppo grande di dati. Ad esempio, McKinsey Global Institute definisce un sistema di Big data come *"dataset la cui taglia/volume è talmente grande che eccede la capacità dei sistemi di database relazionali di catturare, immagazzinare, gestire ed analizzare"*.

A partire dagli anni Ottanta c'è stata una vera e propria esplosione di dati. Contemporaneamente, si è assistito ad un fenomeno caratterizzato da una crescente diffusione di tecnologie quali per esempio il Personal Computer, lo smartphone ecc. Nel 2007 l'uomo è stato paragonato ad una sorgente di dati giornaliera pari a quelli contenuti in 174 giornali. Ad oggi, i dati non strutturati che vengono generati ogni giorno sul web, dai dispositivi intelligenti, dai macchinari nelle fabbriche, raggiungono cifre incredibili, tanto da dover parlare di Zettabyte (10^{21}).

I Big Data sono considerati ormai una preziosa fonte, non a caso sono un vero e proprio asset per le aziende poiché esse, attraverso le tecnologie di Data

Analytics, possono estrarre valore aggiunto indispensabile per il loro business. In particolare, l'opportunità di raccogliere enormi quantità di dati real time, correlarli ed interpretarli abilitando analisi di vario tipo, consente numerose nuove opportunità per le aziende: esse sono in grado di intervenire preventivamente in caso di eventuali guasti, possono prevedere scenari futuri, possono prendere decisioni smart, ottimizzare i processi e massimizzare l'efficienza.

I Big Data sono identificati comunemente da alcune caratteristiche conosciute come le *"3V"*. Nel Febbraio del 2001, Doug Laney, pubblicò un articolo nel quale descrisse la necessità di gestire i prodotti dell'e-commerce secondo un approccio tridimensionale caratterizzato da Volume, Velocità e Varietà, da qui il paradigma delle *"3V"*.

- **Volume**: ogni giorno sono generate enormi quantità di dati durante le attività della vita quotidiana. Secondo un'indagine de *Il Sole 24 Ore* (2019): *"Usando i rapporti sul traffico internet di Cisco e di altri operatori di rete possiamo stimare che l'intero universo digitale è grosso modo di 44 zettabytes. Se la stima è corretta vuol dire che abbiamo a disposizione in bytes 40 volte il numero di stelle osservabili nell'universo"* [28]. Si produce, quindi, un'ingente massa di informazioni, non a caso si è stimato che nel 2020 la quantità di dati è 44 volte maggiore rispetto al 2009. Quest'esplosione inimmaginabile è anche dovuta al fatto che esistono spazi di archiviazione sempre meno costosi e tecnologie all'avanguardia per sfruttare le potenzialità del dato [29].

- **Velocità**: i dati sono generati, archiviati ed analizzati a velocità esponenziali poiché sono disponibili in real time e questo rende indispensabile per le aziende avere adeguate soluzioni di Big Data e Analitycs per poter sfruttare al meglio questo vantaggio.

- **Varietà**: i dati non sono omogenei e strutturati perché provengono da fonti eterogenee (sistemi gestionali aziendali, sensori, social network, open data, smart objects...). Possono essere immagini, video, testi, dati sensibili, ecc. Circa il 90% dei dati prodotti non è strutturato, dunque servono tecniche di archiviazione avanzate e algoritmi complessi per analizzarli [27].

Accanto a queste *3V*, se ne sono aggiunte altre nel tempo [29]:

- **Variabilità**: il concetto di variabilità fa riferimento al contesto in cui vengono generati ed analizzati i dati. Essi, infatti, oltre ad avere formati diversi, provengono anche da contesti molto differenti e quindi l'interpretazione di un determinato dato varia in base al contesto in cui esso è raccolto e analizzato.

- **Veracità**: si usa dire *"Bad data is worse than no data"*, questo perché i dati devono essere affidabili e quindi è indispensabile raccoglierli ed analizzarli correttamente affinché "raccontino" il vero.

- **Valore**: i dati sono fonte di valore, soprattutto negli ultimi anni rappresentano il "nuovo oro". Se un'azienda riesce a sfruttare le potenzialità del dato attraverso le corrette analisi, avrà un significativo vantaggio competitivo rispetto a quelle aziende che non lo fanno.

- **Viralità**: la diffusione dei dati avviene in modo virale così come le informazioni che essi racchiudono.

Il ciclo di vita dei Big Data è diviso in due fasi e in ogni fase è modificato lo stato ed il contenuto dei dati con l'obiettivo di creare valore informativo:

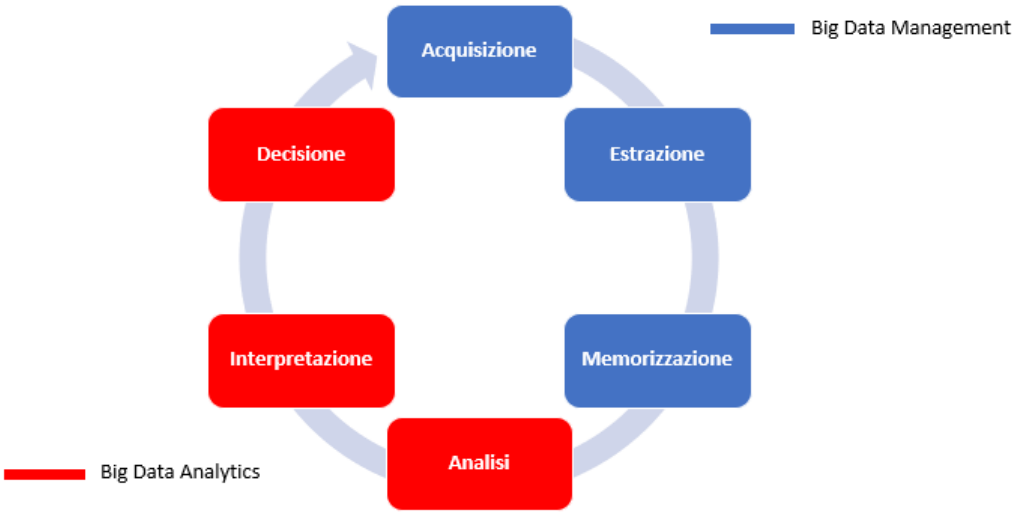


Figura 1.5: Ciclo di vita dei Big Data

1. **Big Data Management**: è l'insieme di tutti i processi e tecnologie per acquisire i dati, estrarli, rappresentarli in forma standard per poterli elaborare ed infine memorizzare. I dati possono essere generati da diverse fonti che, come già detto, sono eterogenee e si possono distinguere le seguenti categorie [30]:

   (a) Dati human generated: sono i dati che vengono generati dalle piattaforme di social network, dai blog, dai portali e-commerce e così via.

   (b) Dati machine generated: sono i dati che vengono prodotti dai sensori posti sulle macchine per il monitoraggio dei vari parametri di funzionamento.

(c) Dati business generated: sono i dati che vengono generati all'interno di un'azienda e per questo motivo possono essere sia human che machine generated.

2. **Big Data Analytics**: è l'insieme di tutti i processi e di tutte le tecnologie per effettuare analisi e quindi per estrarre contenuto informativo prezioso come supporto al processo decisionale. In particolare, ci sono quattro classi di Analytics.

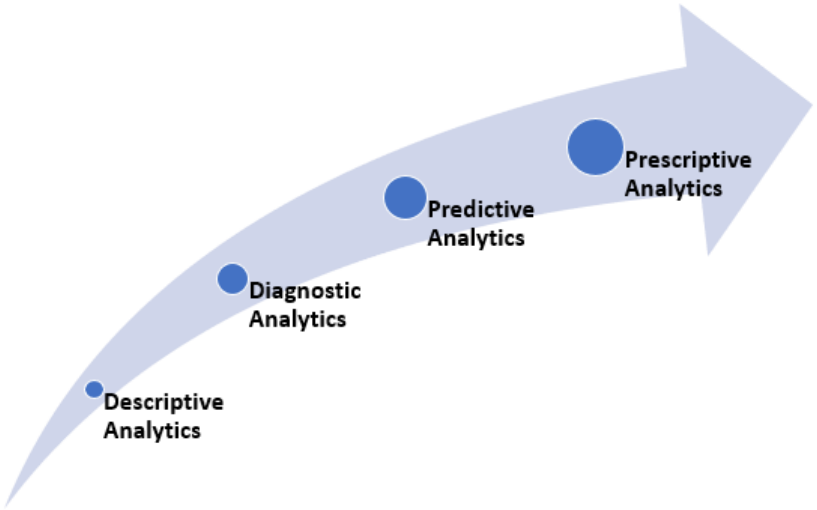


Figura 1.6: Tipologie di Analytics

(a) **Descriptive Analytics**: è il tipo di analisi più semplice, ma è di notevole importanza in quanto funge da base per le successive analisi. Essa è il primo passo di un complesso processo. Risponde alla domanda *"Che cosa è accaduto?"* cioè descrive la situazione attuale e passata dei processi aziendali. Tramite questa analisi si accede ai dati per visualizzare in modo sintetico e attraverso grafici gli indicatori di interesse per l'azienda. E' il tipo di analisi di cui la maggior parte delle aziende si serve.

(b) **Diagnostic Analytics**: questo tipo di analisi risponde alla domanda *"Perché è accaduto?"*. La sua complessità risulta maggiore rispetto a quella dell'analisi descrittiva in quanto richiede uno studio più approfondito dei dati a disposizione.

(c) **Predictive Analytics**: è l'analisi che risponde alla domanda *"Cosa è probabile che accadrà?"*. Affinché sia affidabile è indispensabile che le analisi precedenti, in particolare quella descrittiva, siano state svolte correttamente perché il suo esito sarà buono solo nel caso in cui si disponga

di dati corretti. In questo contesto si utilizzano tecniche matematiche come per esempio la regressione, la proiezione, i modelli predittivi, ecc.

(d) **Prescriptive Analytics**: l'ultimo step permette di rispondere alla domanda *"Quali azioni intraprendere?"* per ottenere un vantaggio futuro o per mitigare una minaccia. Essa si basa sui risultati dell'analisi predittiva. La prescriptive analysis suggerisce le azioni che devono essere messe in atto per un determinato obiettivo. A differenza delle analisi precedenti, utilizza un sistema di feedback per migliorare la sua efficienza.

Dunque, è chiara l'importanza e la preziosità dei Big Data e Analytics nell'era dell'Industria 4.0, tanto da rappresentare la chiave dell'innovazione in una fabbrica smart. Non a caso, secondo un'indagine effettuata dall'Osservatorio del Politecnico di Milano [31], il mercato italiano dei Big Data Analytics è in continua crescita con imprese che raggiungono competenze avanzate nell'utilizzo di queste tecnologie. In particolare, nel 2019, il mercato degli Analytics ha raggiunto 1,7 miliardi di euro, con una crescita del 23% rispetto al 2018, raddoppiando rispetto al 2015. Resta comunque non trascurabile il divario tra Pmi e grandi imprese: tra queste ultime, il 93% investe in progetti di Analytics, mentre tra le Pmi solo il 62%. A livello mondiale, secondo IDC, il mercato Big Data e Analytics è stato pari a 189 miliardi di dollari nel 2019 [32].

1.4.2 Cloud Computing

Il termine *Cloud Computing* fa riferimento a tutte quelle tecnologie che permettono la delocalizzazione dei servizi informatici. Il National Institute of Standards and Technology lo definisce come: *"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"* [33]. Il cloud computing è un modello per abilitare, tramite la rete, l'accesso diffuso, agevole e a richiesta, ad un insieme condiviso e configurabile di risorse di elaborazione (ad esempio reti, server, memoria, applicazioni e servizi) che possono essere acquisite e rilasciate rapidamente e con minimo sforzo di gestione o di interazione con il fornitore di servizi. Il Cloud incarna un vero e proprio nuovo paradigma per fornire servizi informatici attraverso la rete. Anziché dover utilizzare computer sempre più potenti e sofisticati con elevate capacità di calcolo e di memoria, l'utente può impiegare le risorse che desidera quando queste risiedono su server remoti, raggruppati nelle Server Farm, semplicemente collegandosi alla rete Internet. Il termine *Cloud*, in italiano *nuvola*, nasce intorno agli anni Novanta nel mondo delle telecomunicazioni per far riferimento a tutte

quelle unità che fornivano servizi di interconnessione digitale e che sembravano per l'appunto una nuvola. I primi servizi virtuali sono stati erogati intorno alla seconda metà degli anni Novanta, ma la vera svolta arriva con Amazon che dà il via alla vendita dei Web Services nel 2008 (AWS) [34]. I modelli di Cloud Computing si differenziano in *private cloud*, *pubblic cloud* e *community cloud* in riferimento agli utenti che usufruiscono della fornitura dei servizi. Tali modelli differiscono tra loro per via delle caratteristiche dell'infrastruttura informatica. Il *private cloud* è caratterizzato da un'infrastruttura che è ad uso esclusivo di una singola organizzazione. Può essere posseduta, gestita e diretta dall'organizzazione stessa (in house) oppure da una società terza (un hosting server oppure un outsourcer). Il *public cloud* è caratterizzato dal fatto che l'infrastruttura è di proprietà di un fornitore specializzato. Egli offre la propria infrastruttura a tutti gli utenti (che spesso non hanno alcun rapporto tra loro), garantendo loro l'erogazione dei servizi e l'utilizzo delle risorse mediante la rete. Infine, nel *community cloud* l'infrastruttura è utilizzata da utenti che condividono caratteristiche ed interessi comuni. Essa può appartenere a una o più organizzazioni della comunità oppure ad una società terza. Spesso, tale modello è ritenuto adatto per la pubblica amministrazione che infatti è caratterizzata da un insieme di organizzazioni distinte che però appartengono ad uno stesso contesto giuridico/amministrativo e che tendono a seguire uno standard comune. Esistono anche dei cloud ibridi (*indeterminate cloud*) e sono caratterizzati dal fatto che l'infrastruttura è formata da due o più tipi di cloud. Inoltre, i servizi

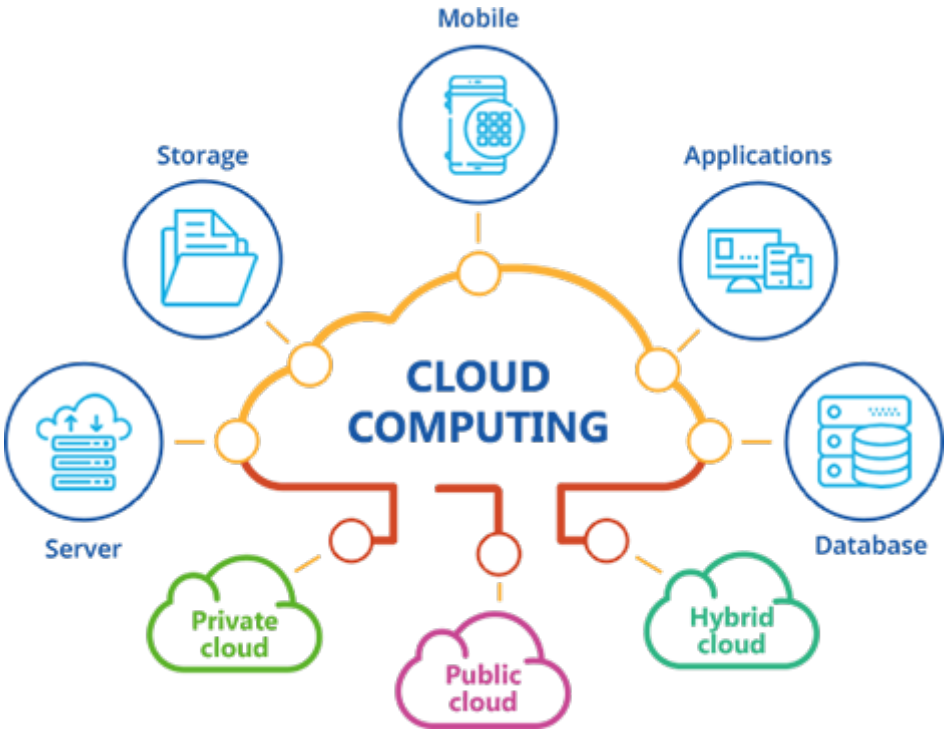


Figura 1.7: Cloud Computing

offerti possono essere suddivisi in:

- **Infrastructure as a Service (IaaS)**: l'infrastruttura è messa a disposizione dal fornitore e il consumatore acquisisce elaborazione, memoria, rete senza poterla né gestire né controllare. Il fornitore dà "in affitto" il server, la capacità di calcolo, la capacità di memorizzazione in modo tale che l'azienda non debba fare investimenti troppo onerosi.

- **Software as a Service (SaaS)**: il fornitore eroga direttamente il servizio e il consumatore utilizza le applicazioni fornitogli. Le applicazioni sono accessibili su diversi dispositivi attraverso browser o apposite interfacce. Il consumatore, anche in questo caso, non gestisce né controlla l'infrastruttura e neppure la rete, il server, la memoria, i sistemi operativi. Egli ha solo la possibilità di effettuare qualche personalizzazione configurando alcuni parametri dell'applicazione.

- **Platform as a Service (Paas)**: il consumatore distribuisce l'infrastruttura creata da lui, mentre il fornitore offre gli strumenti per ospitare e sviluppare l'applicazione (ad esempio librerie, programmi, ecc).

È facile dedurre quali siano i vantaggi del cloud computing: l'azienda ha la possibilità di usufruire di determinati servizi senza dover fare investimenti specifici molto onerosi e senza dedicare risorse umane alla gestione delle infrastrutture IT non legate al core business aziendale. L'utente paga solo quanto consuma (pay-as-you-use) e questa modalità è molto vantaggiosa in termini di risparmio sui costi. Inoltre, dato che le infrastrutture sono generalmente condivise tra più utenti, c'è la possibilità di ottenere economie di scala e garantire elevati livelli di sicurezza a costi non eccessivi. Il rovescio della medaglia, però, è la perdita di controllo sui propri dati: vi è un vero e proprio rischio di riversare nelle mani di pochi grandi soggetti fornitori (Amazon, Google, Microsoft) un enorme quantità di Big Data decisamente preziosi. Infine, si pone il problema legato alla riservatezza dei dati che deve essere assicurata nonostante le vulnerabilità ereditate dalle tecnologie utilizzate.

Nell'era dell'Industria 4.0 sempre più aziende si avvalgono dei servizi cloud perchè sono indispensabili per gestire i dati e le funzionalità dei macchinari, per il monitoraggio e il controllo dei processi. Il cloud permette alla smart factory di avere a disposizione potenza di calcolo, di estrapolare informazioni rilevanti a partire dai Big Data e quindi di individuare e sfruttare al meglio nuove opportunità di business [35].

1.4.3 Robot Autonomi

Con l'Industria 4.0 aumentano i robot in fabbrica che svolgono mansioni complesse diventando sempre più autonomi e flessibili. In questo contesto, migliora l'interazione uomo-macchia: la macchina è in grado di collaborare con l'operatore umano sulla linea, diventando una risorsa indispensabile all'interno della Smart Factory. Inoltre, i robot sono sempre più connessi tra di loro in modo tale da raccogliere informazioni utili al monitoraggio del sistema produttivo, diventandone parte attiva. Quando si parla di robot nel contesto dell'Industria 4.0, non si intendono semplici macchinari che eseguono istruzioni, ma bensì strumenti avanzati in grado di comunicare con il mondo reale e interagire con l'uomo nelle attività aziendali. Non a caso si parla di *"Cobot"* per indicare un robot che interagisce in sicurezza con l'uomo nel suo spazio di lavoro. Questo nome nasce proprio per distinguere i robot collaborativi dal vecchio concetto di robot, il quale era in grado di lavorare in modo autonomo ma separatamente dall'uomo. Invece, i cobot sono *collaborative robot* e cioè lavorano in sicurezza con l'uomo senza alcuna barriera fisica di protezione perché sono dotati di sistemi di sicurezza, sensori, telecamere che ne consentono la coesistenza nella fabbrica con l'uomo senza rischiare incidenti. Inoltre, caratteristica

Figura 1.8: Cobot

interessante dei cobot, è che non sono programmati prima di essere inseriti sulla linea di produzione, ma apprendono sul campo e sono riprogrammabili, motivo per cui è possibile spostarli e addestrarli allo svolgimento di altre mansioni. I cobot sono particolarmente indicati per tutte quelle lavorazioni delicate che non sarebbero fattibili con le macchine e che sarebbero però troppo alienanti per l'uomo. Sotto questo aspetto, permettono all'uomo di assumere sempre più il ruolo di parte pensante del sistema. Infine, la collaborazione uomo-macchia permette anche un miglioramento della condizione lavorativa, in termini per esempio di sicurezza [36].

1.4.4 Simulazione

La *simulazione* è uno strumento molto prezioso e potente che permette di replicare il mondo fisico in un modello virtuale, il già nominato digital twin, mediante strumenti virtuali in 3D. All'interno della smart factory ogni elemento fisico ha un suo gemello digitale: macchine, prodotti e persone. La creazione di un modello virtuale è importante per identificare eventuali problemi ed elaborare soluzioni preventive prima del passaggio alla sfera fisica. In questo contesto, risultano di fondamentale importanza i Big Data, infatti, le nuove tecnologie ed i sensori installati sulla linea di produzione permettono di acquisire dati che consentono la realizzazione della simulazione in modo preciso. Di fondamentale importanza è l'accuratezza dei dati perché da essa dipenderà l'affidabilità e la precisione della simulazione. Grazie all'utilizzo della simulazione le aziende sono in grado di abbattere i tempi e costi di progettazione perché possono prevedere eventuali problematiche in anticipo e trovare di conseguenza una soluzione nella "sfera virtuale" senza dover prendere provvedimenti a posteriori sulla sfera fisica, momento nel quale sarebbe più complicato e costoso intervenire, migliorando lo sviluppo del prodotto e la sua qualità [37].

1.4.5 Realtà Aumentata

Con il termine *realtà aumentata* si intende la tecnica attraverso cui si aggiungono informazioni alla scena reale [38], cioè si aumenta il mondo reale senza però sostituirlo. Le tecnologie che permettono di fare ciò, consentono all'uomo di essere interattivo con l'ambiente che lo circonda e di manipolarlo digitalmente. Il termine nasce nel 1990 ad opera di Thomas Caudell che ha utilizzato questa espressione per riferirsi ai display che gli elettricisti portavano sulla testa per assemblare complessi cablaggi [39]. Oggi, il progresso tecnologico ha permesso il realizzarsi di un nuovo gruppo di tecnologie per i settori professionali. Queste tecnologie innovative comportano molti benefici, tra cui, per esempio, l'ottimizzazione della realizzazione di un progetto o prodotto, in quanto sarà possibile valutare anticipatamente tutti i dettagli e tutti i parametri in questione prima di costruire fisicamente un prototipo. Alcune applicazioni di tecnologie di realtà aumentata si possono trovare in ambito logistico, dove è possibile effettuare la localizzazione dei prodotti in magazzino o verificare la conformità degli ordini, oppure, nell'ambito della manutenzione, i visori ottici possono supportare l'operatore nel trovare componenti difettosi o guasti. Inoltre, si possono controllare i parametri di un impianto per verificarne il suo funzionamento e addirittura vedere un prodotto prima che esso venga realizzato per valutarne gli aspetti estetico-funzionali ed, eventualmente, migliorarli [40].

1.4.6 Cyber Security

"Prima dell'avvento di Internet la fabbrica era un'entità separata dall'esterno, ma successivamente, e soprattutto con l'implementazione di tecnologie tipiche dell'Industria 4.0, la fabbrica è sempre connessa con l'esterno e, per questo, dal punto di vista della sicurezza dei dati è più vulnerabile di prima "(G. Ferrari) e quindi *"spesso non è più sufficiente criptare i dati, ma occorre adottare politiche di sicurezza e di privacy affidabili e effettuare scelte a partire dalla fase di progettazione del sistema sulla base di un'attenta analisi dei rischi"* (P. Degano). La digitalizzazione dei sistemi industriali e dei macchinari permettono di acquisire dati in tempo reale e comunicarli ad altri sistemi attraverso la rete. Questi dati possono essere utilizzati dall'azienda stessa per il monitoraggio del proprio sistema produttivo, ma anche dai clienti finali e dai fornitori. Il fatto che ci siano dati scambiati in rete e numerosi device collegati tra loro è sicuramente un elemento di vulnerabilità nei confronti di eventuali attacchi, non a caso diventa importante il tema legato alla Cyber-Security. Le tecnologie per la sicurezza dei dati servono per proteggere i sistemi informatici da eventuali danni. Il Laboratorio Nazionale di Cyber-Security e il CIS-Sapienza, nel 2015, hanno stilato un Framework Nazionale per la Cyber-Security, con tutte le attività necessarie per gestire in modo corretto il problema della vulnerabilità dei dati. Spesso le aziende non hanno la reale consapevolezza del rischio correlato al fatto di essere connessi alla rete, ma in realtà si tratta di un rischio reale da non sottovalutare. Il framework identifica nello specifico cinque categorie di azioni [37]:

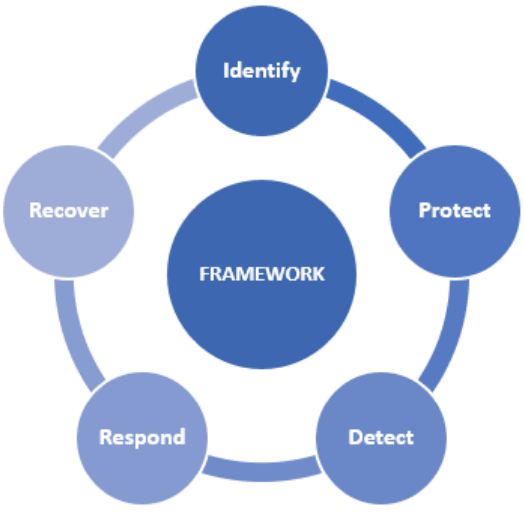


Figura 1.9: Framework Cyber-Security

- **Identify**: azioni necessarie per individuare i rischi.
- **Protect**: azioni necessarie per proteggersi dagli attacchi informatici.

- **Detect**: misure per individuare gli attacchi in corso.
- **Respond**: misure da mettere in pratica per fronteggiare un attacco.
- **Recover**: azioni da mettere in atto per ripristinare il sistema dopo un attacco.

L'industria della Cyber-Security ha sfiorato, a livello mondiale, nel 2015, i 75 miliardi di dollari e nel 2020 è previsto il raddoppio di questa cifra. La compagnia di assicurazioni inglese Lloyds ha stimato che i danni causati dagli attacchi a livello globale ammontano a circa 450 miliardi l'anno. Il 2019 è stato un anno molto duro, non a caso il 66% delle piccole medie imprese è disposto ad aumentare i propri investimenti in Cyber-Security dopo il picco degli attacchi dell'anno precedente. Per questo motivo il 2020 è l'anno della sicurezza, le aziende aumentano la spesa in software per la protezione dei dati e ciò si traduce in una crescita del mercato della Cyber-Security [41]. Secondo l'Osservatorio Information Security e Privacy della School of Management del Politecnico di Milano [42] gli investimenti in sicurezza continuano ad aumentare in Italia raggiungendo nel 2019 il valore di 1,3 miliardi di euro, quasi l'11% in più rispetto al 2018. *"Il mercato italiano dell'Information Security si conferma dinamico e in crescita anche nel 2019. La sicurezza informatica non è più percepita come un ostacolo all'adozione di nuove tecnologie e servizi, ma come un fattore fondamentale per il successo del business, ma c'è ancora molta strada da fare nella maturità organizzativa. Ben il 40% delle imprese non ha una funzione specifica che si occupi di sicurezza informatica: questo genera incertezza e oltre un'impresa su due è insoddisfatta di come viene gestita. Emerge la necessità di adottare un modello integrato di governance della security che permetta di definire modalità di intervento uniformi e monitorare in maniera completa e affidabile le potenziali minacce"* (Alessandro Piva, Direttore dell'Osservatorio Information Security Privacy).

1.4.7 Additive manufacturing

L' *additive manufacturing*, anche conosciuta come 3D printing o stampa 3D, consiste nella realizzazione di oggetti tridimensionali tramite produzione additiva: si parte da un modello digitale 3D realizzato con l'ausilio di software appositi e, successivamente, si crea l'oggetto fisico, strato dopo strato (da qui il termine additive manufacturing), attraverso una stampante 3D o una tecnologia analoga [37]. Dunque, viene meno la necessità di fondere materiali in apposite stampe o rimuoverli dalle forme piene (subtractive manufacturing).

Il modello 3D del pezzo che si vuole realizzare è ottenuto mediante un software CAD o CAM ed è inoltrato sotto forma di file ad un'apposita stampante che lo suddivide in strati della stessa altezza. L'oggetto viene realizzato sovrapponendo

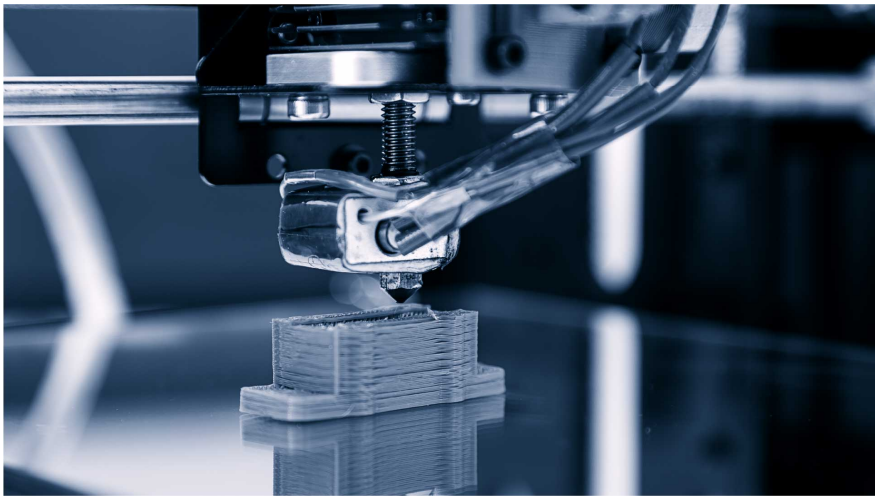


Figura 1.10: Additive Manufacturing

strato per strato. La stampante 3D non è una tecnologia nuova, essa infatti deriva dall'ambito del "rapid prototyping" (RP), termine che fa riferimento alla realizzazione di un prototipo in tempi molto brevi e che risale agli anni Ottanta. La stampa 3D, nella seconda metà degli anni Novanta, raggiunge in modo significativo diversi settori manifatturieri, in particolar modo quello dell'automotive, fino a diffondersi in ambiti del tutto nuovi come quello biomedicale, aerospaziale, ecc. L'aspetto innovativo introdotto dalla manifattura additiva è che gli oggetti non si realizzano più attraverso stampi o per asportazione di materiale (fresatrici, torni, ecc), ma per stratificazione. Ciò, senza dubbio, introduce la possibilità di realizzare nuove forme geometriche, di utilizzare in modo più efficiente il materiale (diminuendo gli scarti), di ridurre le scorte di magazzino e, in linea con le tendenze introdotte dall'Industria 4.0, è possibile realizzare una "personalizzazione di massa" dei vari prodotti, rendendo la produzione altamente flessibile. Nello stesso lotto di produzione è possibile creare oggetti diversi, su misura, senza dover intervenire sull'attrezzaggio dei macchinari. IDTechEx, in un report sull'argomento in questione, afferma che il mercato globale di apparecchiature, software, servizi di stampa 3D raggiungerà nel 2029 il valore di 31 miliardi di dollari [43][44].

1.4.8 Integrazione dei sistemi verticale e orizzontale

Un'altra tecnologia abilitante nel contesto dell'Industry 4.0 è l'*integrazione orizzontale*, cioè dei processi produttivi e quella *verticale*, cioè della produzione con le altre aree aziendali (ad esempio progettazione, acquisti, controllo qualità,...). In altre parole, l'integrazione orizzontale consiste nella connessione tra macchine, parti di impianti o unità produttive: essa consente a strumenti, dispositivi, processi di lavorare insieme. L'integrazione verticale, invece, fa in modo che i dati siano

utilizzati come supporto al processo decisionale, trasferendoli dalla produzione ai livelli organizzativi più alti (ad esempio il marketing). L'integrazione è la parola d'ordine dell'Industria 4.0 e l'integrazione orizzontale e verticale sono la spina dorsale della smart factory. Per quanto riguarda l'integrazione orizzontale, si è già precedentemente parlato di quanto l'Industria 4.0 punti all'interconnessione tra sistemi fisici e informatici conquistando livelli di automazione e flessibilità senza precedenti. L'integrazione orizzontale può avvenire su tre livelli. Nel momento in cui le macchine sono connesse con le unità produttive, possono scambiare continuamente informazioni in tempo reale riguardo le performance, eventuali malfunzionamenti, errori e lo stato di manutenzione. Si parla quindi di integrazione orizzontale a livello di produzione, che ha come obiettivo quello di massimizzare l'efficacia, cioè ridurre i costi legati alla manutenzione. Se invece un'azienda è caratterizzata da più impianti dislocati sul territorio, l'integrazione orizzontale permette ad essi di comunicare e condividere tutti i dati. Infine, a livello di supply chain, la possibilità di condividere dati e informazioni migliora enormemente tutta la filiera di approvvigionamento e produzione, per questo motivo è bene che anche i fornitori siano coinvolti in questo sistema di integrazione orizzontale. L'integrazione verticale, invece, ha come finalità quella di unire tutte le parti all'interno dell'organizzazione, a partire dalla produzione fino al management, includendo marketing, controllo qualità, ufficio acquisti, vendite, reparto ricerca e sviluppo e così via. In un'azienda integrata verticalmente i dati sono trasmessi in modo veloce e trasparente dai livelli più bassi a quelli più alti e viceversa. In questo modo, sono facilmente utilizzabili come supporto al processo decisionale[45][46][47].

1.5 Manutenzione

In questa tesi ci si concentra su un particolare aspetto dell'Industria 4.0 e cioè la possibilità di effettuare la manutenzione predittiva grazie all'accesso real time ai dati. L'utilizzo di sensori e di algoritmi per il monitoraggio e la gestione degli impianti aiuta le aziende a trarre valore concreto dal dato per il proprio business e trascina il settore manifatturiero verso una nuova economia *data-driven* [48]. Il termine *manutenzione* è stato definito dalla norma SS-EN 13306 (2001, p.7) come la combinazione di tutte le azioni tecniche, amministrative e gestionali messe in atto durante il ciclo di vita di un'entità, destinate a mantenerla o a riportarla ad uno stato in cui essa possa eseguire la funzionalità richiesta [49]. La manutenzione si ripartisce in [50]:

- **Manutenzione ordinaria**: si intendono tutte quelle attività volte a mantenere l'integrità originaria del bene, a ripristinarne l'efficienza, a contenerne il

normale degrado d'uso, a garantirne la vita utile e a far fronte ad eventuali eventi accidentali.

- **Manutenzione straordinaria**: si intendono quegli interventi non ricorrenti e non ripetibili volti a prolungare la vita utile del macchinario o a migliorarne l'efficienza, la produttività, l'affidabilità e la manutenibilità [51].

In questa sede, ci si concentrerà sul primo tipo di manutenzione. La manutenzione ordinaria si divide in diverse tipologie [52]:

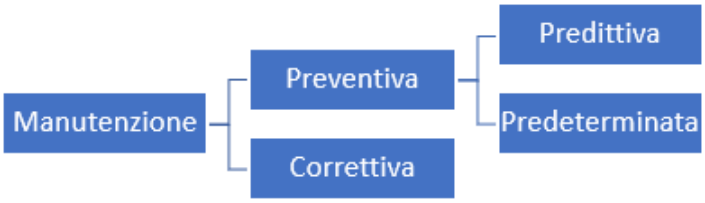


Figura 1.11: Tipologie di manutenzione ordinaria

- **Manutenzione preventiva**: questa manutenzione ha come obiettivo quello di prevenire il verificarsi di un problema. Gli interventi di questo tipo, quindi, sono programmati ed eseguiti ad intervalli periodici indipendentemente dallo stato del macchinario. Poiché i macchinari sono caratterizzati da un'infinità di componenti soggetti ad usura, essi vanno monitorati costantemente nel tempo per evitare avarie: una corretta gestione della manutenzione preventiva permette all'azienda di non trascurare questi aspetti e di intervenire prima che i componenti si usurino e si guastino. Pianificare in modo efficiente le operazioni di manutenzione preventiva consente di ridurre gli interventi di manutenzione correttiva (dispendiosi in termini di costi e tempi) e migliorare le condizioni di lavoro e l'efficienza del sistema produttivo [53].
  - **Manutenzione predittiva**: rispetto alla manutenzione preventiva i cui interventi sono programmati sulla base del tempo o dell'intensità di utilizzo di un determinato macchinario, la manutenzione predittiva si focalizza sullo stato di salute di un macchinario e, attraverso tecniche di condition monitoring e modelli matematici, predice quando si verificherà un guasto e quindi il tempo residuo prima che esso avvenga [54].
  - **Manutenzione predeterminata**: è un tipo di manutenzione preventiva che è eseguita ad intervalli predeterminati per ridurre la probabilità di

guasto e la degradazione del macchinario. Essa prevede la messa in atto di interventi manutentivi indipendentemente dallo stato di salute del macchinario.

- **Manutenzione correttiva**: detta anche manutenzione reattiva, è il tipo di manutenzione che si basa sulla riparazione del guasto una volta che esso è avvenuto. Non si tratta quindi di un intervento da pianificare, ma semplicemente di azioni che vengono messe in atto in caso di necessità. Questo tipo di manutenzione comporta senza dubbio perdite dovute ai fermi produttivi, al tempo investito e ai costi che si generano [55].

1.5.1 Manutenzione predittiva

La manutenzione predittiva, come accennato, è un tipo di manutenzione preventiva che si effettua a seguito dell'individuazione e della misurazione di uno o più parametri e dell'estrapolazione, secondo i modelli appropriati, del tempo residuo prima del guasto (UNI EN 13306). A differenza della manutenzione preventiva, la manutenzione predittiva entra in gioco solo quando determinati parametri indicano che occorre intervenire. Dunque, è possibile individuare anticipatamente il deterioramento di un asset sulla base dei dati che esso stesso trasmette durante il suo funzionamento, ottimizzando gli interventi, minimizzando i fermi macchina ed evitando le manutenzioni non necessarie. Secondo alcuni studi condotti già intorno agli anni Sessanta e Settanta dalla Marina Americana, circa il 18% dei guasti che si verificano lungo linea di produzione sono dovuti all'età della macchina, mentre l'82% si origina in modo del tutto casuale. Alla luce di ciò, la manutenzione preventiva risulta inefficace e anche costosa: basti pensare che se si programmano interventi di manutenzione su asset che non ne hanno alcuna necessità, si sprecano risorse, tempo e si incrementano inutilmente i costi. Infatti, da un'analisi condotta da Oniqua Enterprise Analytics emerge che il 30% degli interventi di manutenzione sono messi in atto con una frequenza superiore a quella necessaria e ciò sembrerebbe essere uno dei principali fattori che causano sprechi eccessivi [56]. Sotto questi aspetti, dunque, la manutenzione predittiva rappresenta uno strumento di analisi avanzato di sistemi industriali e manifatturieri complessi che ha come obiettivo l'aumento di efficienza e produttività. Le aziende oggi, grazie alle nuove tecnologie, riescono a raccogliere una quantità smisurata di dati e senza dubbio, il fatto di avere a disposizione una mole di informazioni simile è un grande vantaggio, ma allo stesso tempo emerge l'inadeguatezza dei classici modelli statistici di gestione dei dati. Per questo motivo si sono sviluppate e si stanno affermando sempre più tecniche di analisi statistiche moderne, note come *Machine Learning*. Queste tecniche stanno mostrando alle realtà manifatturiere e industriali le loro enormi potenzialità nell'implementare un

sistema di manutenzione predittiva automatico [57]. Il termine Machine Learning, in italiano Apprendimento Automatico, è un concetto che non ha una definizione univoca, ma anzi è difficile definirlo in modo specifico e universale poiché comprende una vasta rete di tecniche, algoritmi e strumenti che possono essere utilizzati ed implementati in contesti anche molto differenti. In generale si può affermare che il Machine Learning è un ramo dell'Intelligenza Artificiale che fa in modo che sistemi e macchine acquisiscano la capacità di imparare e migliorare le proprie capacità in modo autonomo dall'esperienza, senza l'intervento dell'uomo. Le macchine e i robot sono così in grado di eseguire i loro compiti e funzioni in modo sempre migliore, perfezionando le loro capacità e le loro risposte, grazie all'esperienza acquisita nel tempo. Il funzionamento di questo meccanismo si basa sull'utilizzo di particolari algoritmi che riescono a prendere determinate decisioni e a mettere in atto azioni, partendo da una conoscenza primitiva. Le modalità di apprendimento automatico sono tre e sono differenti tra loro per via degli algoritmi che le caratterizzano e del modo in cui il sistema accumula i dati e impara:

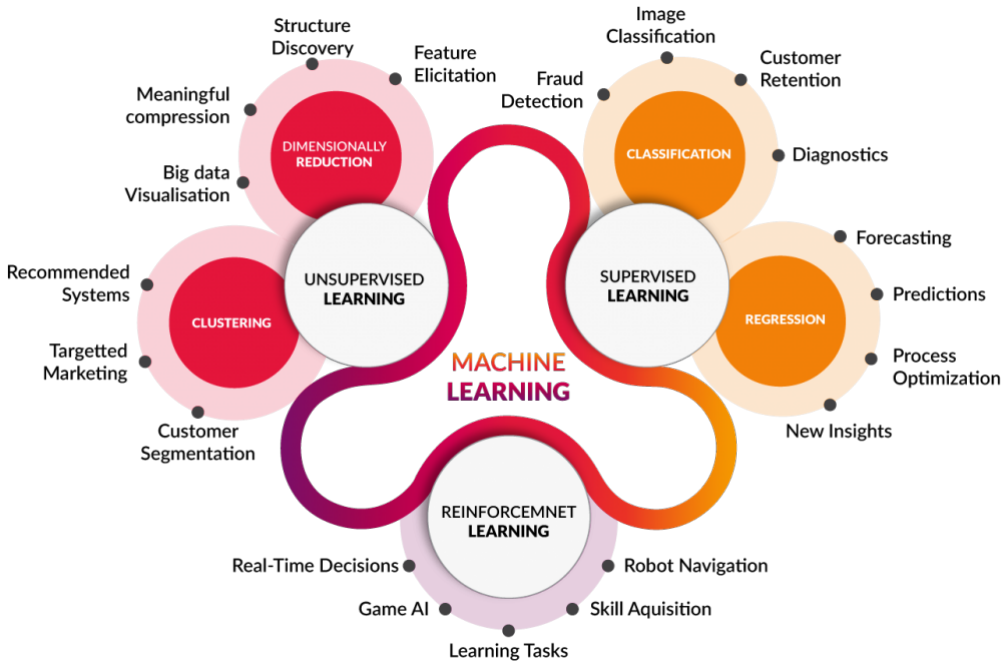


Figura 1.12: Tipi di apprendimento [58]

- **Apprendimento supervisionato**: gli algoritmi che appartengono a questa categoria necessitano di alcune informazioni specifiche e di modelli per costruire un database di esperienze. La macchina, quando dovrà risolvere un determinato problema, attingerà informazioni dal proprio bagaglio di esperienze. In questo caso si parla di algoritmi di Classificazione e modelli di regressione, i quali

necessitano della conoscenza delle etichette di classe in modo da predire quelle dei dati futuri.

- **Apprendimento non supervisionato**: sono algoritmi che non hanno necessità di informazioni e conoscenze circa lo stato reale del sistema, ma permettono alla macchina di analizzare essa stessa le informazioni a propria disposizione e a catalogarle e organizzarle. In questo caso ci si riferisce ad algoritmi di clustering che non richiedono la conoscenza delle etichette, ma che dividono i dati identificando similitudini e raggruppandoli in funzione di determinate caratteristiche. Fanno parte di questa categoria di Machine Learning anche l'association rule mining per l'estrazione delle associazioni frequenti, molto utile soprattutto nel settore commerciale e del marketing e la riduzione della dimensionalità, che prevede la selezione senza supervisione delle features più rilevanti (con maggiore valore predittivo) al fine di associare le etichette di classe. L'aspetto positivo di questa categoria di algoritmi è, senza dubbio, la scarsa necessità di informazione che nella realtà è effettivamente difficile da avere a disposizione, ma allo stesso tempo è più complesso valutare questi modelli rispetto a quelli supervisionati [59].

- **Apprendimento per rinforzo**: è un processo in cui le labels sono generate da un modello che si aggiorna in modo automatico con l'arrivo di nuovi dati. Appartiene a questa categoria il semi-supervised learning in cui si utilizzano le previsioni fatte da un modello non supervisionato come target class in un processo supervisionato. Sono modelli adatti a quelle situazioni in cui si hanno a disposizione pochi dati etichettati e molti non etichettati. In pratica, si utilizzano i dati provvisti di labels per valutare se i clusters ottenuti in modo unsupervised sono significativi e, in caso affermativo, si valida il modello ottenuto e lo si utilizza per le successive predizioni.

La manutenzione predittiva si basa proprio su questi metodi di apprendimento: gli algoritmi ricevono i dati raccolti dai sensori posti sulle macchine e, grazie a modelli sempre più performanti, esse riescono a rispondere adeguatamente di fronte a malfunzionamenti o anomalie sulla base di ciò che hanno appreso con l'esperienza e non semplicemente in seguito al superamento di soglie definite a tavolino [60] [58].

Capitolo 2

Stato dell'arte

Nel capitolo 1.4.1, si è ampiamente discusso dell'importanza dei Big Data e delle tecniche di Data Analytics attraverso le quali le aziende possono estrarre valore dai dati che esse stesse generano durante le loro attività, migliorando così le proprie performance e prendendo decisioni "smart", strategiche, efficienti ed efficaci. Tra i vari utilizzi che l'azienda può fare dei suoi dati c'è sicuramente la predictive maintenance, che si è visto essere di fondamentale utilità per contribuire a ridurre sprechi di tempo e risorse. Tuttavia, per sfruttare questi vantaggi e ottenere valore aggiunto, è indispensabile effettuare le analisi corrette, altrimenti le informazioni ricavate potrebbero essere fuorvianti o addirittura errate. Per questo motivo, nel presente capitolo, si illustra la corretta metodologia da seguire per estrapolare conoscenza dai dati, cioè per trasformare il dato grezzo ed apparentemente sterile in contenuto informativo prezioso. Il percorso di estrazione della conoscenza è chiamato **Knowledge Discovery Process** o **KDD** ed è un processo iterativo ed interattivo che ha come obiettivo l'identificazione delle relazioni tra i dati. È caratterizzato da diverse fasi, tra le quali, la più importante, è quella del *Data Mining*, non a caso questo termine è spesso considerato sinonimo di Knowledge Discovery Process, anziché come un suo sotto-processo [61].

2.1 Data Mining

Con il termine *Data Mining* ci si riferisce a quell'insieme di tecniche che hanno come fine l'estrapolazione di un massiccio numero di informazioni, non risapute a priori, a partire da un'enorme quantità di dati [62]. Le tecniche di Data Mining permettono di scoprire le relazioni tra i dati, le associazioni, gli schemi ricorrenti e le anomalie. Formalmente, per indicare il risultato dell'estrazione dei dati, si utilizza il termine *Pattern* che sta ad indicare una rappresentazione *sintetica* che sia *comprensibile*, potenzialmente *utile*, *valida* e precedentemente *sconosciuta*. Quindi,

partendo da contenuti "criptati", eterogenei, ridondanti e non strutturati si giunge ad una conoscenza sfruttabile in molti contesti, soprattutto in quello aziendale [63]. Le tecniche di Data Mining nascono intorno agli anni Ottanta per superare i limiti delle tradizionali tecniche di analisi dei dati, le quali hanno come obiettivo quello di verificare o rifiutare un'ipotesi presa ad oggetto di studio. Le nuove tecniche, al contrario, prevedono la generazione di ipotesi proprio a partire dalle attività di analisi, cioè sono gli algoritmi che, in modo automatico, cercano schemi, relazioni e quindi l'utente diventa una figura generica, mentre i dati sono i veri protagonisti. In altre parole, con il Data Mining non si vuole rispondere ad una domanda specifica o confermare un'ipotesi, bensì cercare di ricavare tutte le informazioni potenzialmente utili ed interessanti a partire da un dataset sconosciuto [64]. Proprio per questa differenza con le tecniche tradizionali, si può affermare che il Data Mining si configura come l'integrazione di più discipline, quali la *statistica*, il *machine learning* e l'*intelligenza artificiale*. Infatti, si sviluppa sui concetti tipici della statistica tradizionale come l'utilizzo del campionamento, la stima e i test d'ipotesi, ma con l'integrazione di algoritmi, modelli e tecniche di apprendimento automatico tipiche del machine learning e dell'intelligenza artificiale. Esistono due principali modelli di Data Mining:

- **Modelli descrittivi**: identificano nel dataset gruppi di dati tra loro correlati o individuano correlazioni frequenti. L'obiettivo è quindi quello di estrapolare i pattern che riassumono le relazioni tra i dati che non sono note a priori.

- **Modelli predittivi**: richiedono una conoscenza a priori sui dati e hanno come obiettivo quello di classificare gli eventi futuri e cioè di prevedere il valore di un determinato attributo sulla base dei valori assunti da altri attributi noti.

2.2 Knowledge Discovery Process

Il processo di estrazione della conoscenza dai dati può essere suddiviso in cinque step, ognuno dei quali è caratterizzato da una molteplicità di algoritmi che possono essere utilizzati e scelti in base all'obiettivo dell'analisi [65].

Prima di iniziare a percorrere il vero e proprio Knowledge Discovery Process è importante soffermarsi su una fase preliminare molto significativa. Come prima cosa è fondamentale comprendere e conoscere il dominio applicativo, l'ambito di raccolta dei dati e gli obiettivi dell'analisi. Gli esperti che si occupano di portare avanti le analisi sui dai, devono conoscere le esigenze dell'azienda. A tal fine, risulta particolarmente importante il supporto da parte degli esperti di dominio, i quali conoscono perfettamente i processi da cui si generano i dati e sanno cosa deve essere

fatto. Essi hanno un quadro chiaro della situazione e sono in grado di indirizzare l'analista verso una corretta impostazione del lavoro.

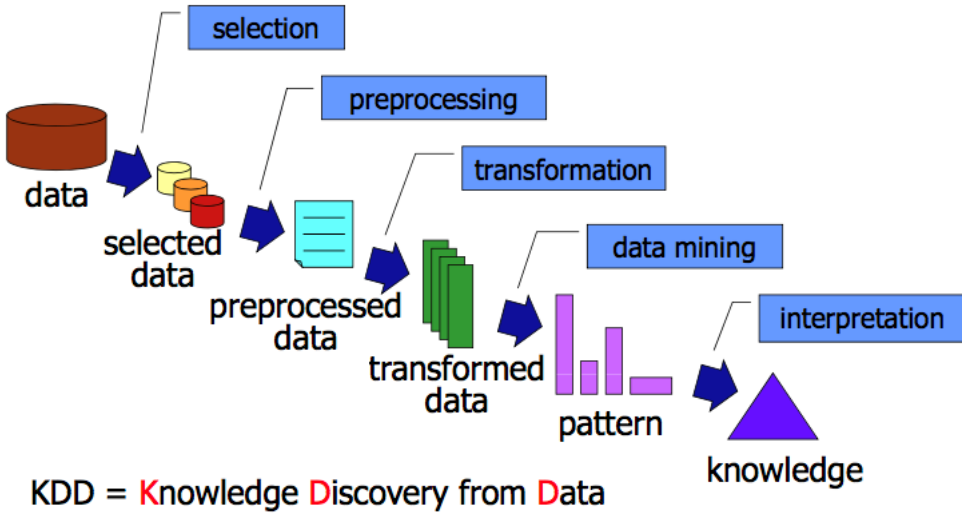


Figura 2.1: Knowledge Discovery from Data [65]

2.2.1 Data Selection

Il primo passo del Knowledge Discovery Process è la selezione dei dati. Quando si inizia una nuova analisi ci si trova di fronte ad un dataset ricco di dati eterogenei. Tra questi, è fondamentale selezionarne una porzione rilevante, cioè è importante considerare nell'analisi solo un campione rappresentativo del dataset a partire dal quale si generalizzeranno i risultati ottenuti. Il campione deve essere "rappresentativo" nel senso che deve mantenere e rappresentare tutte le caratteristiche del dataset. In questo modo, si opererà solo sui *target data* (o dati obiettivo) e non si avranno difficoltà dal punto di vista computazionale. Questa operazione può essere effettuata mediante un *sampling* (o campionamento). Esistono differenti tecniche di sampling:

- **Simple Random Sampling**: si crea un sottoinsieme di item a partire da un grande insieme di dati (popolazione). Ogni item ha la stessa probabilità degli altri di essere selezionato poiché è estratto casualmente.
- **Sampling without replacement**: ogni item, dopo essere stato selezionato, non è reinserito nella popolazione dei dati di partenza, quindi si evita che l'estrazione di un item possa avvenire più di una singola volta.
- **Sampling with replacement**: ogni item, dopo essere stato selezionato, è reinserito nella popolazione dei dati di partenza e quindi può essere nuovamente estratto.

- **Stratified sampling**: questo tipo di sampling crea un sottoinsieme di oggetti caratterizzato dal fatto di avere un numero di dati per ogni etichetta che rispecchia le proporzioni del dataset completo. In questo modo, all'interno del campione sono rappresentate tutte le classi, anche quelle meno rilevanti (con pochi record), al contrario nei sampling randomici le classi meno frequenti potrebbero non essere rappresentate [65].

2.2.2 Preprocessing

La seconda fase del Knowledge Discovery Process è molto importante e può anche risultare molto complessa. Il preprocessing consiste nella preparazione dei dati per le successive analisi. Infatti, i dati reali solitamente non sono adatti per essere utilizzati così come sono, in quanto possono esserci valori mancanti, dati che non appartengono ai range ammessi ed errori di varia natura che, se non gestiti, influirebbero negativamente sui risultati degli esperimenti. L'esito di un'analisi è affidabile quando è frutto di dati attendibili e di qualità e proprio per questo motivo è importante la fase in esame. Thomas C. Redman afferma che la scarsa qualità dei dati è causa di una perdita del 10-20% delle entrate per l'azienda (*DM Review, 2014*). I problemi che maggiormente affliggono la qualità del dato sono:

- **Rumore e outliers**: il rumore consiste nella variazione dei valori originali ed è un errore casuale, mentre un outlier è un record per il quale uno o più attributi assumono un valore significativamente diverso da quelli assunti dagli altri record. La presenza di questi valori anomali all'interno di un dataset può essere dovuta a molteplici fattori, tra i quali ci sono gli errori nella raccolta dei dati, gli errori casuali ed eventuali difetti nel funzionamento degli strumenti utilizzati per la loro raccolta.

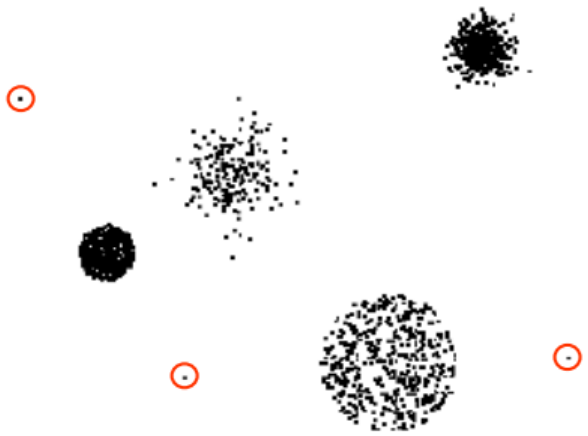


Figura 2.2: Outliers [61]

- **Valori mancanti**: può accadere che alcuni record non abbiano un'informazione circa un determinato attributo poiché essa non è stata raccolta per un qualche motivo (basti pensare ad una persona che decide di non dichiarare la propria età) oppure perché quella data informazione non può essere raccolta per ogni tipo di record (ad esempio non si può chiedere ad un bambino quali siano i suoi guadagni annuali). È importante, però, capire come gestire i missing values: si può decidere di eliminare il record che presenti valori mancanti, si può sostituire il valore mancante sulla base di un criterio di stima (ad esempio sostituendolo con un valore fisso o la media dei valori assunti per quell'attributo all'interno del dataset) oppure si può ignorare l'attributo nell'analisi.

Il preprocessing è caratterizzato da due fasi principali:

1. **Data Cleaning**: questo passo del preprocessing consiste nella gestione delle anomalie sopracitate e quindi nella rimozione effettiva di tutto il rumore e gli outliers presenti nei selected data e nella risoluzione di eventuali problemi come valori mancanti o la presenza di dati inconsistenti. Per portare a termine correttamente questa fase delicata bisogna prima effettuare un' *analisi univariata* che consiste nello studio dei dati e dei valori assunti dai vari attributi. Per fare questo tipo di lavoro occorre avere una conoscenza approfondita del contesto in cui si opera e del significato di ciascun attributo per vedere se effettivamente i valori assunti dalle features siano coerenti e non ci siano anomalie. Successivamente si passa all'*analisi multivariata* che consiste nell'applicazione di particolari algoritmi che effettuano l'outlier detection. Ad esempio, si può utilizzare l'algoritmo di clustering DBSCAN per individuare i valori anomali (esso, infatti, isola le aree dense da quelle sparse e i punti appartenenti alle aeree sparse rappresentano gli outliers che sono inseriti nel cluster 0). Una volta individuati gli outliers si può procedere con la loro rimozione.

2. **Data Integration**: questa fase non è sempre realizzabile, ma può essere messa in atto qualora si abbiano a disposizione più dataset differenti, ma che tra loro potrebbero integrarsi per arricchire i contenuti delle analisi. Ad esempio, se si sta effettuando uno studio su un dataset che riguarda l'inquinamento atmosferico, potrebbe essere interessante associare un dataset con i dati sul traffico urbano per trovare eventuali correlazioni e informazioni aggiuntive [65].

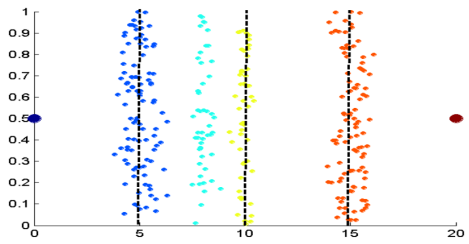
2.2.3 Data Transformation

In fase di data transformation si effettuano ulteriori modifiche sui dati selezionati in base alle analisi che si vogliono effettuare nella fase successiva di Data Mining e gli obiettivi da raggiungere. Ad esempio, si possono effettuare operazioni di sampling e feature selection per selezionare solo gli attributi rilevanti al fine delle analisi. Inoltre, i dati possono essere trasformati e convertiti in altri formati attraverso cambiamenti di scala e operazioni matematiche. L'obiettivo di questa fase è rappresentare i dati in modo tale che siano congeniali alle analisi. A tal fine le operazioni possibili sono le seguenti:

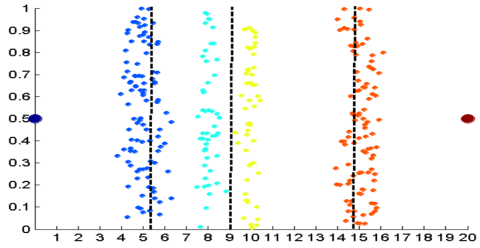
- **Aggregazione dei dati**: consiste nel combinare due o più record in uno unico cambiando quindi la granularità del dato. Questa operazione può servire per ridurre la cardinalità del dataset e per rendere i dati più stabili perché una volta aggregati, essi tendono ad avere una minore variabilità. Ad esempio, si possono aggregare le città in regioni o stati effettuando quindi un cambio di scala oppure le vendite giornaliere in vendite mensili o annuali. Questa operazione comporta però anche la perdita di informazioni preziose, quindi è importante trovare il giusto trade-off tra la perdita di informazioni utili e la riduzione della cardinalità del dataset che comporta vantaggi in termini di tempi di esecuzione degli algoritmi e memoria utilizzata.
- **Riduzione dei dati**: consiste nell'eliminazione dei dati che può avvenire sia lungo le righe (quindi a livello di record), sia lungo le colonne (a livello di attributi).
  - **Sampling**: come già accennato per la prima fase del KDD, si parla di sampling quando la riduzione avviene in termini di record e cioè quando si seleziona un campione rappresentativo di dati all'interno di un più ampio insieme.
  - **Feature Selection**: si parla di feature selection quando si selezionano solo determinati attributi scartandone altri. Uno dei criteri per effettuare la feature selection consiste nel calcolare la matrice di correlazione tra gli attributi, cioè si considera per ogni coppia di caratteristiche il coefficiente di correlazione di Pearson. Successivamente, si eliminano dall'analisi gli attributi fortemente correlati perché introdurrebbero rumore e risulterebbero ridondanti. Inoltre, si possono eliminare gli attributi che non risultano rilevanti ai fini dell'analisi, in questo caso può essere utile il supporto dell'esperto di dominio.
  - **Discretizzazione**: si applica quando si ha a che fare con valori continui che si trasformano in discreti. Essa è indispensabile per l'applicazione

di alcuni algoritmi in fase di Data Mining come ad esempio le Regole di Associazione. La discretizzazione consiste nel raggruppare in N split i valori assunti da una variabile continua per ridurre la cardinalità del dominio di tale variabile. La discretizzazione può avvenire in tre modi differenti ed è difficile stabilire a priori quale comporti prestazioni migliori e quindi solitamente si provano tutte e tre le metodologie:

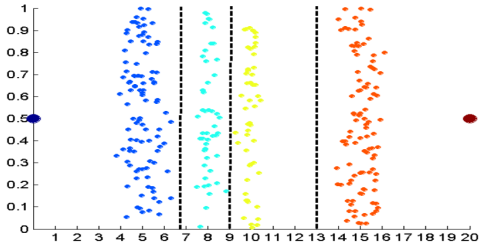
* Si dividono i valori ammissibili in N intervalli della stessa ampiezza: l'attributo numerico viene trasformato in attributo categorico e appartiene ad uno split che ha la stessa ampiezza degli altri. Il vantaggio di questa discretizzazione è che è incrementale.
* Si dividono i valori ammissibili in N intervalli della stessa frequenza, ovvero ogni split contiene lo stesso numero di punti. Lo svantaggio di questa soluzione è che non è incrementale.
* Si utilizza una tecnica di Clustering, ad esempio il K-Means.



(a) *Discretizzazione ad intervalli della stessa ampiezza*



(b) *Discretizzazione ad intervalli della stessa frequenza*



(c) *Discretizzazione mediante K-Means*

Figura 2.3: Tipologie di discretizzazione

- **Trasformazione degli attributi**: consiste nell'applicazione di una funzione che mappa i valori assunti da un determinato attributo in un nuovo set di valori in modo tale che ogni vecchio valore sia identificato con uno nuovo. Ad esempio, si può applicare la normalizzazione, che risulta indispensabile per l'applicazione di alcune tecniche di Data Mining, tra cui gli algoritmi di

Clustering. Esistono differenti tipi di normalizzazione e spesso occorre provarli tutti prima di poter capire quale garantisce prestazioni migliori:

- **Z-Score**: in questo tipo di normalizzazione i valori di un attributo sono normalizzati in base alla media e alla deviazione standard

$$z_i = \frac{x_i - \mu}{\sigma}$$

  x_i è il valore dell'attributo da standardizzare, μ è la media e σ la deviazione standard. Questo tipo di normalizzazione è utile quando non si conoscono il minimo e il massimo dell'attributo o quando ci sono molti outliers.

- **Min-max**: questa normalizzazione si basa sull'utilizzo del minimo e del massimo di un determinato attributo e si calcola con la seguente formula

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

  x_i è il valore dell'attributo da standardizzare, $\min(x)$ e $\max(x)$ sono il valore minimo e massimo assunti dall'attributo all'interno del dataset [61][65].

2.2.4 Data Mining

A questo punto si passa alla vera e propria fase di *Knowledge Extraction* attraverso le tecniche di Data Mining. È importante identificare l'obiettivo del Knowledge Discovery Process per capire quale algoritmo si deve applicare per estrarre la conoscenza desiderata. Una volta ben chiari gli obiettivi e scelti gli algoritmi, si potranno applicare valutando i parametri che garantiscono migliori prestazioni. Le tecniche di analisi applicabili sono raggruppabili in quattro categorie: *Regole di Associazione*, *algoritmi di Clustering*, *algoritmi di Classificazione* e *Concept Drift*.

Regole di Associazione

Le *Regole di Associazione* appartengono alla sfera delle analisi esplorative e il loro obiettivo è quello di far emergere le relazioni nascoste tra i dati e cioè le correlazioni frequenti o i pattern in un database transazionale. Un esempio pratico di applicazione delle regole di associazione lo si può trovare nelle *market basket analysis* e cioè nell'analisi sulle abitudini del consumatore: è emerso, per esempio, che il cliente che acquista i pannolini, acquista spesso anche la birra. In questo caso, la correlazione frequente è data dalla coppia di oggetti "pannolini" e "birra" e questa informazione può essere molto importante nell'organizzazione degli scaffali del supermercato (si potrebbe pensare di mettere i due oggetti in scaffali limitrofi)

oltre che a pensare a strategie di prezzo mirate (si potrebbero applicare promozioni che si attivino nel momento in cui si acquistino i due prodotti congiuntamente). Una regola di associazione non è altro che una regola di implicazione:

$$r : A \Longrightarrow B$$

A e B sono insiemi di oggetti e A rappresenta il corpo della regola, mentre B rappresenta la testa della regola. Per poter chiarire il concetto delle regole di associazione è importante introdurre il significato di:

- **Itemset**: è un insieme di oggetti (può contenere da 1 a n oggetti).
- **K-Itemset**: è un insieme di K oggetti.
- **Supporto**: è la frequenza statistica della regola, cioè rappresenta la frequenza dell'itemset all'interno del dataset. Un itemset è definito frequente se il supporto è pari o maggiore di una certa soglia. In termini formali, data la regola r: $A \Longrightarrow B$, il supporto è la frazione di transazioni (T) che contengono sia A che B:
$$Supporto = \frac{\#(A, B)}{|T|}$$
- **Confidenza**: misura la forza dell'implicazione e cioè, formalmente, data la regola r: $A \Longrightarrow B$, la confidenza è la frequenza di B nelle transazioni che contengono A.
$$Confidenza = \frac{Supporto(A, B)}{Supporto(A)}$$

$Supporto$ e $Confidenza$ sono due metriche attraverso le quali si misura la robustezza della regola. Sono due parametri di input del modello perché le regole di associazione sono estratte secondo il seguente criterio:

$$\begin{cases} Supporto \geqslant minsupthreshold \\ Confidenza \geqslant minsupthreshold \end{cases}$$

Questo significa che una correlazione verrà considerata *frequente* e quindi degna di nota se e solo se il supporto e la confidenza superano o eguagliano le soglie prestabilite. Le soglie sono da scegliere con attenzione poiché un supporto troppo alto potrebbe portare alla scoperta di regole scontate, mentre un supporto troppo basso farebbe emergere un numero troppo elevato di regole. Si tratta quindi di un problema di giusto trade-off.

E' importante introdurre anche un'altra metrica, cioè il $Lift$, poiché la confidenza potrebbe risultare fuorviante quando la testa della regola ha una frequenza molto alta. In questi casi occorre osservare il $Lift$ che misura le correlazioni tra gli oggetti

nel corpo e nella testa della regola (indica, cioè, come l'occorrenza di un evento fa aumentare le occorrenze dell'altro):

$$Lift = \frac{Confidenza(r)}{Supporto(A)}$$

Se:

- $Lift = 1$: le regole sono *irrilevanti* e quindi non devono essere considerate poiché gli eventi sono statisticamente indipendenti.
- $Lift > 1$: gli eventi hanno correlazione positiva.
- $Lift < 1$: gli eventi hanno correzione negativa.

Clustering

La *Cluster Analysis* si basa su una categoria di algoritmi di Data Mining che ha come obiettivo quello di trovare e raggruppare item che siano tra loro simili o correlati e, allo stesso tempo, di separarli dagli oggetti che siano differenti o non correlati. Gli oggetti che condividono caratteristiche comuni sono raggruppati in sottoinsiemi detti clusters. La somiglianza tra item di uno stesso gruppo può essere valutata secondo due criteri che sono scelti in base al tipo di dato che si sta analizzando:

- **Distanza**: si utilizza questo concetto quando i dati sono di tipo numerico. La distanza, a sua volta, può essere calcolata in modalità differenti, ad esempio può essere la distanza Euclidea, quella di Minkowski o di Chebychev.
- **Similarità**: si utilizza il concetto di similarità in caso di dati testuali.

L'obiettivo degli algoritmi di Clustering è quello di trovare gruppi che siano:

- **Coesi**: in caso di dati numerici la distanza tra i punti appartenenti allo stesso cluster deve essere minimizzata (distanza intra-cluster), al contrario, se si ha a che fare con dati testuali, la similarità deve essere massimizzata all'interno del cluster.
- **Ben separati**: in caso di dati numerici la distanza tra punti appartenenti a clusters diversi deve essere massimizzata (distanza inter-cluster), al contrario, la similarità, in caso di dati testuali, deve essere minimizzata tra clusters differenti.

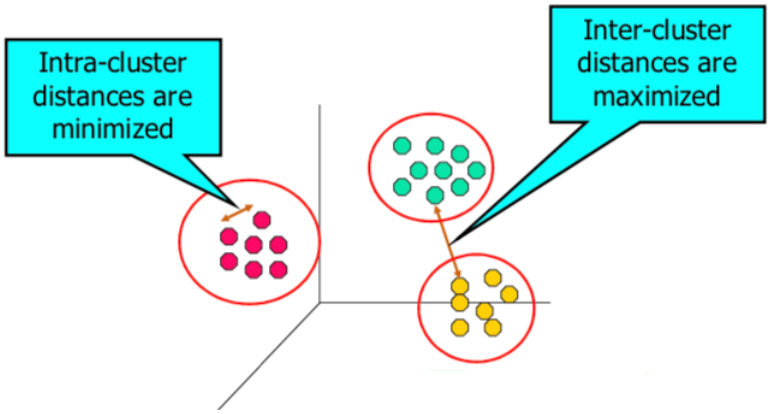


Figura 2.4: Separazione e coesione tra clusters [65]

Gli algoritmi di Clustering si suddividono in due macro-categorie:

- **Algoritmi gerarchici**: la soluzione è un insieme di clusters nidificati, identificabili con una rappresentazione gerarchica ad albero o dendrogramma.
- **Algoritmi partizionali**: il dataset è suddiviso in n clusters e ogni punto è attribuito ad uno ed un solo cluster.

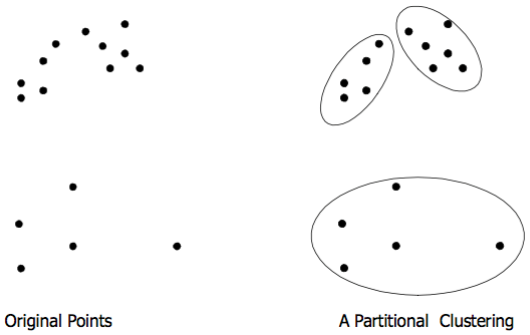


Figura 2.5: Clustering partizionale [65]

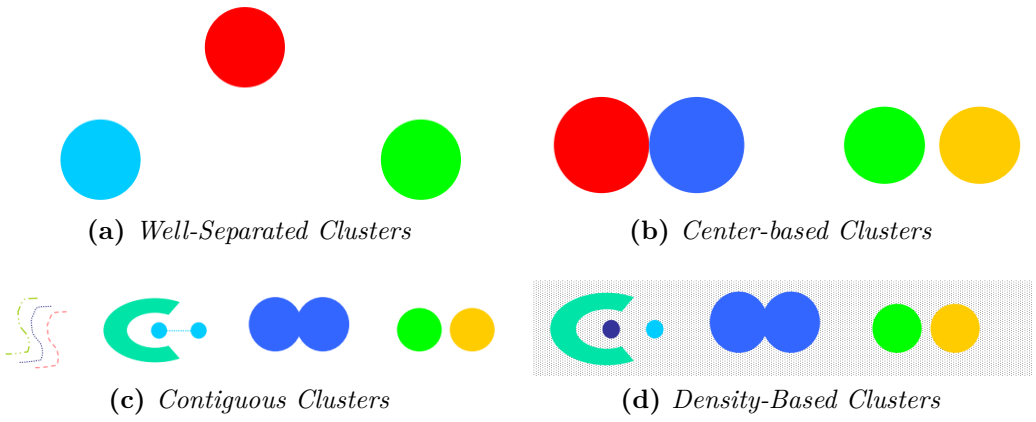
In base alla tipologia di algoritmo scelto, si possono ottenere partizionamenti:

- **Esclusivi o non esclusivi**: un partizionamento è esclusivo quando ogni punto appartiene ad uno ed un solo cluster. Questa soluzione si ottiene con gli algoritmi di Clustering di tipo partizionale. Al contrario, la soluzione è non esclusiva se un punto può appartenere a più clusters a seconda del partizionamento che si seleziona. Questo tipo di soluzione si ottiene con gli algoritmi gerarchici.
- **Completi o parziali**: il partizionamento è completo se ad ogni punto del dataset è attribuito un cluster di appartenenza, è parziale se invece l'appartenenza ad un cluster è definita solo per un sottogruppo di punti.

- **Omogenei o eterogenei**: i clusters ottenuti possono essere omogenei o eterogenei in termini di cardinalità, forma e densità.

Inoltre, esistono diversi tipi di cluster:

- **Well-Separated Clusters**: i clusters sono well-separated se c'è la massima distanza tra punti appartenenti a gruppi diversi.
- **Center-based Clusters**: i punti all'interno del gruppo sono più vicini al centro del proprio cluster piuttosto che al centro degli altri clusters. Il centro è solitamente un centroide (cioè la media di ogni punto del gruppo) o un medoide (se si elegge un punto rappresentativo in ogni cluster). La differenza tra centroide e medoide consiste nel fatto che il centroide, essendo frutto di una media, potrebbe non essere un elemento del gruppo, mentre il medoide, poiché è l'oggetto che meglio rappresenta il gruppo, è sicuramente un punto della base dati.
- **Contiguous Clusters**: un punto nel cluster è più vicino (o simile) agli altri punti del cluster piuttosto che agli altri punti che non appartengono a quel cluster.
- **Density-based Clusters**: i gruppi hanno la stessa densità, ovvero si separano le regioni ad alta intensità di punti da quelle a bassa intensità. Questa tipologia di cluster è molto utile nel caso in cui ci siano outliers e rumore all'interno del dataset da identificare e rimuovere.



(a) *Well-Separated Clusters* **(b)** *Center-based Clusters*

(c) *Contiguous Clusters* **(d)** *Density-Based Clusters*

Figura 2.6: Tipologie di cluster [65]

In generale, il concetto di cluster è ambiguo, non esiste una definizione formale per valutare se la soluzione ottenuta applicando un determinato algoritmo sia buona oppure no. Ad esempio, se si considera un dataset caratterizzato da 20 punti, essi potranno essere rappresentati secondo diversi partizionamenti (sei, due, quattro

clusters) e non esiste una soluzione migliore delle altre a priori, ma dipende dal tipo di analisi e dall'obiettivo che si ha.

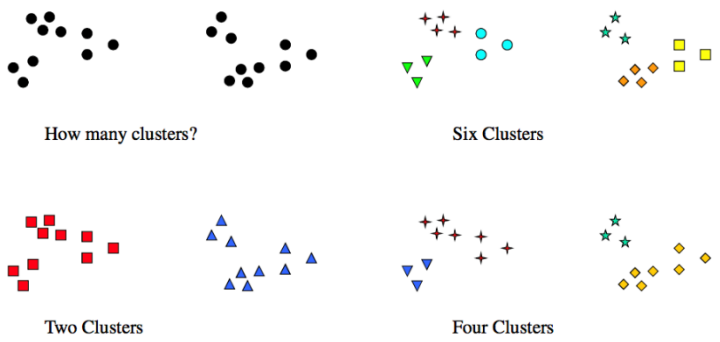


Figura 2.7: Differenti clusters [65]

Gli algoritmi di Clustering sono utilizzati nelle analisi di tipo esplorativo, cioè quando non si ha un obiettivo specifico, ma si cercano le relazioni nascoste tra i dati. Inoltre, fanno parte della categoria di algoritmi di apprendimento unsupervised poiché non è richiesta la conoscenza dell'etichetta di classe dei dati: le labels sono assegnate sulla base di caratteristiche comuni nel momento stesso in cui i dati sono associati ad un determinato cluster, il quale rappresenta la classe di appartenenza. La Cluster Analysis è spesso impiegata in ambito economico-aziendale per identificare clienti che hanno caratteristiche comuni, al fine di promuovere campagne promozionali diversificate, oppure in ambito biologico può essere utilizzata per derivare le tassonomie di piante e animali o per suddividere in categorie i geni che hanno funzionalità simili. In poche parole, l'analisi dei clusters è ampiamente utilizzata in contesti molto differenti tra loro.

Si fornisce ora una breve descrizione degli algoritmi di clustering principali: *K-Means*, *DBSCAN* e *Hierarchical Clustering*. Si ricorda che, poiché il clustering è basato sul concetto di distanza e similarità, i dati devono essere sempre normalizzati prima dell'applicazione degli algoritmi [65].

K-Means Il K-Means è un algoritmo di tipo partizionale e fornisce una soluzione completa, dunque ogni punto appartiene ad un cluster ed avrà la sua etichetta. L'algoritmo riceve come input un parametro K che rappresenta il numero di partizioni che si vuole ottenere e, successivamente, assegna i punti ai clusters in modo tale da rendere la similarità intra-cluster elevata e la similarità inter-cluster bassa. La similarità è misurata rispetto al valore medio degli oggetti di un cluster e dunque ogni gruppo è rappresentato dal proprio centroide. Al primo step, l'algoritmo seleziona in modo randomico K oggetti nel dataset che rappresentano

inizialmente i K centroidi. Gli oggetti rimanenti vengono associati al centroide più vicino (l'algoritmo calcola per ogni punto la distanza dai K centroidi nominati, ad esempio con la distanza euclidea). Al termine del primo step, si ricalcolano i centroidi poiché quelli nominati precedentemente non sono più rappresentativi. Una volta calcolati, si ripetono gli step precedenti e quindi si riassegnano i punti del dataset al centroide più vicino. Il processo è iterato fino a quando i centroidi non si modificano più. Occorre ricordare, però, che la soluzione ottenuta dipende sempre dai K centroidi che sono selezionati inizialmente in modo randomico e quindi, riapplicando l'algoritmo per più di una volta, si otterranno soluzioni differenti. Dunque, è importante avere un indicatore di riferimento che identifichi la soluzione migliore. Una metrica molto utilizzata è il *Sum of Squared Error*, o SSE. Questa metrica è calcolata come la somma degli errori al quadrato di ogni punto, dove l'errore è la distanza tra il punto del cluster e l'oggetto rappresentativo del cluster stesso:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

x è un punto nel cluster C_i e m_i è il punto rappresentativo del cluster C_i. Questo indicatore misura quanto un gruppo è coeso: più il gruppo è coeso, meglio è rappresentato dal centroide, se il gruppo invece è poco coeso è mal rappresentato dal centroide. L'SSE deve essere calcolato per ogni cluster e, più il suo valore è basso, più il gruppo è coeso e dunque tra due clusters, si sceglie quello con SSE inferiore. Inoltre, i gruppi tendono ad essere più coesi quando sono più piccoli e cioè quando K è più elevato. Quindi l'SSE diminuisce al crescere di K. Una tecnica utilizzata per trovare il valore di K ottimale da passare come input al modello consiste proprio nel plottare l'SSE rispetto al valore di K. La curva che si ottiene ha un andamento decrescente poichè all'aumentare del numero di clusters, essi diventano più coesi e il valore dell'SSE diminuisce. Il K ottimale può essere identificato nel punto di massima decrescita (ginocchio). Il tipo di partizionamento usato dal K-Means è omogeneo rispetto alla cardinalità, densità e forma. Se i partizionamenti sono omogenei rispetto alla cardinalità significa che il K-Means tende a creare dei clusters bilanciati. L'omogeneità rispetto alla forma si traduce in gruppi che sono rappresentati tramite centroide o medoide e quindi sono center-based. Infine, l'omogeneità rispetto alla densità si traduce nel fatto che se il database non è omogeneo, l'algoritmo non funziona [65].

DBSCAN Il DBSCAN è un algoritmo che fornisce soluzioni qualitativamente migliori rispetto ad altri modelli, ma ha lo svantaggio di essere molto oneroso dal punto di vista computazionale, tanto da rendere spesso necessaria un'operazione di sampling per poterlo applicare. Il DBSCAN è un algoritmo che fornisce una

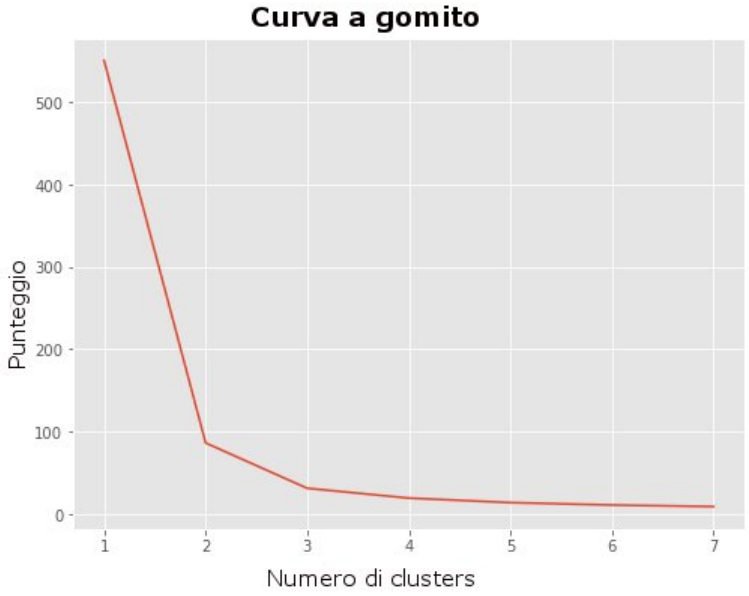


Figura 2.8: SSE [65]

soluzione partizionale e parziale, inoltre i clusters ottenuti sono density-based e cioè l'algoritmo identifica e divide le aree dense da quelle sparse. Le aree dense rappresentano il contenuto informativo principale, mentre quelle sparse gli outliers (proprio per questo motivo la soluzione è parziale). Ecco perché, il DBSCAN, è spesso utilizzato già in fase di preprocessing [65]. I parametri che l'algoritmo richiede sono due :

- **Epsilon**: è il raggio dell'area che si vuole considerare.
- **Minpoints**: è il numero minimo di punti che deve essere presente in un'area di raggio Epsilon affinché essa venga definita densa e quindi è la cardinalità minima di un gruppo.

L'idea alla base del DBSCAN è che un punto A è raggiungibile da un altro punto B se la loro distanza è minore di *Epsilon* e se il punto A è circondato da un numero sufficiente di punti pari al *minpoints*. Se valgono queste due condizioni, allora A e B appartengono ad un cluster. L'algoritmo inizialmente considera un punto casuale e calcola la regione intorno ad esso di raggio *Epsilon*. Se quest'area contiene un numero di punti almeno pari al *minpoints*, allora si crea un cluster, altrimenti è etichettato come rumore. Se un punto è associato ad un cluster, tutti i suoi vicini (ovvero i punti che appartengono all'area di raggio *epsilon* intorno ad esso), sono etichettati con la medesima classe. Quindi, un punto che inizialmente è considerato rumore, durante le iterazioni successive potrebbe far parte del vicinato di un punto che viene associato ad un cluster e di conseguenza, anch'esso, sarà associato al

medesimo gruppo e non sarà più rumore. Il processo continua sino a quando non sono stati considerati tutti i punti [66].

Hierarchical Clustering Un algoritmo di clustering gerarchico produce un set di clusters nidificati rappresentabili attraverso un dendrogramma. Sul dendrogramma, l'asse delle ordinate rappresenta la distanza tra i clusters, mentre l'asse delle ascisse riporta i singoli punti del dataset. A seconda di dove si "taglia" il dendrogramma si ottengono partizionamenti diversi. Ogni punto può appartenere a più clusters in base al partizionamento scelto e non è necessario definire a priori il numero di gruppi che si vuole ottenere.

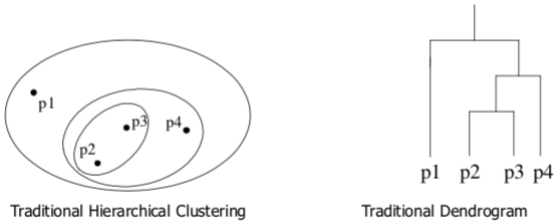


Figura 2.9: Clustering gerarchico e dendrogramma [65]

Esistono due approcci principali per poter applicare gli algoritmi gerarchici, anche se entrambi portano allo stesso risultato:

- **Agglomerativo**: il dendrogramma è generato dal basso verso l'alto, cioè si parte dalla fase in cui ogni punto costituisce un cluster a sé e ad ogni step successivo si aggregano tra loro i punti/clusters per ottenere nuovi partizionamenti.
- **Divisivo**: il dendrogramma è generato a partire dalla radice e quindi dalla situazione in cui tutti i punti appartengono ad un unico cluster. Si procede con successive divisioni dei punti/clusters sino ad arrivare alla situazione in cui ogni punto costituisce un gruppo a sé.

Per valutare come aggregare oppure dividere i dati, l'algoritmo si basa sull'utilizzo di una *matrice di similarità/distanza* detta anche *matrice di prossimità*. Questa matrice considera inizialmente la distanza tra ogni coppia di punti per decidere quali di essi aggregare in clusters. Negli step successivi, la matrice avrà sulle righe e sulle colonne non più punti ma clusters e quindi valuterà quali tra essi unire finché non si giunge ad ottenerne solo uno. Quindi, risulta particolarmente importante il calcolo della distanza inter-cluster, ovvero tra gruppi differenti. Essa può essere calcolata in modi differenti:

- **Minimo**: l'algoritmo calcola la distanza tra ogni coppia di punti dei gruppi e sceglie la distanza minima.

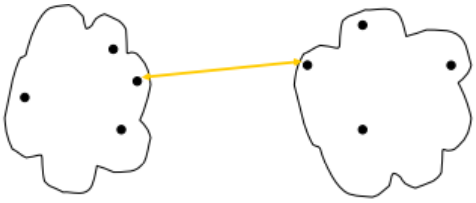


Figura 2.10: Minima distanza [65]

- **Massimo**: La distanza tra due clusters è rappresentata dalla massima distanza tra quelle calcolate tra ogni coppia di punti appartenenti ai due gruppi.

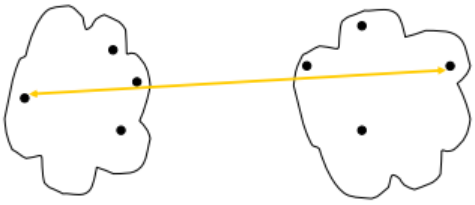


Figura 2.11: Massima distanza [65]

- **Media**: si calcola la media delle distanze tra ogni coppia di punti dei due gruppi.

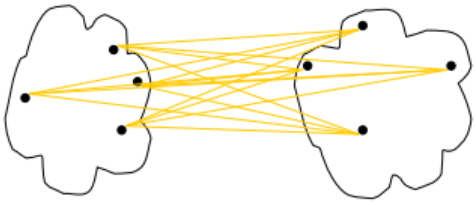


Figura 2.12: Distanza media [65]

- **Distanza tra centroidi**: si calcola la distanza tra i rispettivi centroidi dei due gruppi.

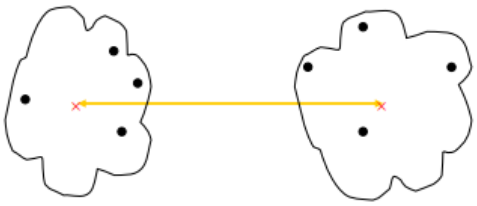


Figura 2.13: Distanza tra centroidi [65]

Queste metriche producono risultati diversi e non è possibile decidere a priori quale tra esse fornisca la soluzione migliore. Bisogna quindi provarle tutte e scegliere alla luce dei risultati ottenuti [65].

Classificazione

Gli algoritmi di *Classificazione* sono tecniche di Data Mining appartenenti alla sfera dell'apprendimento automatico supervisionato poiché generano un modello sui dati andando ad analizzare un dataset arricchito con informazioni note priori con l'obiettivo di fare delle predizioni. Dunque, gli algoritmi di Classificazione si possono utilizzare solo quando sono note le etichette di classe sui dati storici e cioè quando si ha un'informazione sui dati che è quella che si vuole predire. Essi hanno lo scopo di associare una classe a ciascun record sulla base di dati storici dai quali il classificatore apprende alcune relazioni [65]. Essendo questa categoria di algoritmi l'oggetto del presente lavoro, si parlerà più in dettaglio dell'argomento nel capitolo successivo.

Concept Drift

Quando si creano modelli predittivi, occorre considerare il problema del *Concept Drift*. Esso si riferisce al fatto che i dati, nel corso del tempo, mutano e modificano la loro distribuzione, basti pensare ai dati che sono raccolti in un contesto di produzione industriale: essi cambiano nel tempo a causa del consumo degli strumenti e delle attrezzature oppure per via dell'introduzione di nuovi macchinari o ancora per fattori ambientali. Di conseguenza, i modelli predittivi che erano stati programmati per riconoscere determinati dati, potrebbero non essere in grado di captare l'arrivo di quelli nuovi e, di conseguenza, le predizioni potrebbero diventare meno accurate e cioè essere afflitte da degrado o addirittura errate. Questo fenomeno si verifica perché i modelli predittivi basano il loro funzionamento su relazioni statiche tra variabili che sono ricavate dai dati storici. Dopodiché, si presuppone che tale modello possa essere valido e performante anche sui dati futuri, ma questa assunzione non è sempre vera poiché i dati non hanno una natura stazionaria. L'obiettivo è quello riuscire a rilevare quando le predizioni sono afflitte da degrado e quindi quando le prestazioni del modello si sono ridotte e, allo stesso tempo, l'attuale sfida consiste nel riuscire a mantenere aggiornati i modelli sulla base dei dati che cambiano, senza doverli ricreare da zero [67].

Esistono differenti tipologie di Concept Drift che possono essere classificate, ad esempio, in base all'istante temporale ed alla velocità cui si manifesta il fenomeno:

- **Sudden concept drift**: quando il cambiamento nella distribuzione dei dati è istantaneo o improvviso ed è noto l'istante di tempo in cui si verifica tale mutamento repentino.
- **Gradual concept drift**: quando il cambiamento nella distribuzione dei dati è graduale e avviene col passare del tempo. Ci saranno quindi fasi in cui

saranno presenti congiuntamente una quota di dati vecchi e una quota di dati nuovi finchè i dati nuovi non sostituiranno del tutto quelli vecchi.

- **Incremental drift**: il cambiamento avviene in modo incrementale, le variazioni sono molto piccole e si accumulano nel corso del tempo.
- **Re-occurring drift**: si verifica un mutamento nella distribuzione dei dati per poi ritornare allo stato precedente [68][69].

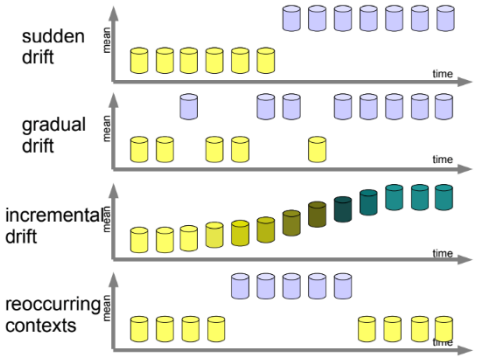


Figure 4: Illustration of the four structural types of the drift.

Figura 2.14: Tipologie di Concept Drift [69]

2.2.5 Data Interpretation

La fase conclusiva del Knowledge Discovery Process consiste nell'interpretazione dei risultati ottenuti con le analisi effettuate. Quest'ultimo step permette di consolidare la conoscenza estratta e di ricavare quelle informazioni che potrebbero essere di supporto al processo decisionale.

Capitolo 3

Metodologia

L'uomo ha un'innata abilità nel classificare gli oggetti in categorie, ma è in grado di farlo solo su dati semplici, poco numerosi e non particolarmente ricchi di attributi. Per questo motivo nascono gli algoritmi di *Classificazione*, che sono impiegati in quelle situazioni in cui si ha a che fare con complessi dataset di grandi dimensioni nei quali sarebbe impossibile, per la mente umana, estrapolare relazioni tra variabili. Gli algoritmi di Classificazione si applicano sui dataset che hanno una particolare struttura: essi sono caratterizzati da item ai quali è associata una tupla (X,y), dove X è un insieme di attributi che descrivono il record, mentre y rappresenta l'etichetta, o label, che è una variabile categorica e identifica la classe di appartenenza del record. Un modello di classificazione è una rappresentazione astratta della relazione che c'è tra gli attributi e le etichette e il suo obiettivo, una volta ricevuto come input il set di attributi di un determinato item da classificare, è quello di associare ad esso un'etichetta di classe.

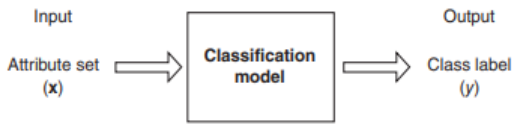


Figura 3.1: Modello di Classificazione [70]

I modelli di classificazione possono essere utilizzati in fase di Data Mining per due principali funzioni:

- **Modelli predittivi**: il classificatore è impiegato come modello predittivo quando attribuisce la label di classe ai dati che non erano etichettati.
- **Modelli descrittivi**: il classificatore può essere utilizzato per trovare le caratteristiche che meglio identificano le varie classi. Questo aspetto può essere importante in molti ambiti, ad esempio quello della medicina, nel quale

non è sufficiente scoprire che una determinata paziente rischia di sviluppare un cancro al seno, ma è necessario individuare quali siano i fattori determinanti che la espongono al rischio.

In questo contesto si utilizzano gli algoritmi di Classificazione come modello predittivo poiché l'obiettivo è quello di associare le labels agli item sulla base della relazione che c'è tra variabili ed etichette. Il modello, per venire a conoscenza di questa relazione astratta, deve essere creato utilizzando una porzione di dataset che si chiama **training set**. Il dataset di training è una porzione dei dati originali che contiene una serie di record con i rispettivi attributi e le rispettive etichette. Il modello di classificazione è creato attraverso l'utilizzo di un *learning algorithm* e il processo di apprendimento sui dati di training si chiama **induzione**. L'obiettivo del learning algorithm è quello di trovare il modello che meglio rappresenti la relazione intrinseca tra attributi e labels. Una volta applicato il learning algorithm sul dataset di training e una volta "allenato" il modello di classificazione, si passa alla sua applicazione sul **test set**, ovvero sulla porzione rimanente di dataset alla quale, però, è rimossa l'etichetta di classe. Il modello associa ad ogni record una label e questo processo prende il nome di **deduzione**. In un secondo momento si valutano le prestazioni del classificatore per verificare che il modello abbia predetto correttamente le etichette di classe. Per fare ciò, si mettono a confronto le labels predette sul test con quelle originali (che nella fase di test erano state eliminate). L'informazione circa la precisione del classificatore può essere riassunta nella cosiddetta **matrice di confusione**. Essa racchiude sulle righe le etichette originali, sulle colonne quelle predette: l'elemento (i, j) rappresenta il numero di punti della classe i a cui è stata assegnata la classe j. I valori sulla diagonale sono i record predetti correttamente. In Tabella 3.1 si può vedere un esempio di matrice di confusione: gli elementi a e d rappresentano gli item ai quali il classificatore attribuisce una corretta etichetta, mentre b rappresenta il numero di punti appartenenti alla classe 1 a cui però è stata associata l'etichetta 0 e c sono i punti appartenenti alla classe 0 a cui però è stata associata l'etichetta 1. Sebbene la matrice di confusione dia informazioni interessanti e di immediata

	Classe 1 pred	Classe 0 pred
Classe 1 orig	a	b
Classe 0 orig	c	d

Tabella 3.1: Matrice di confusione

comprensione visiva circa la bontà del modello di classificazione, è utile calcolare altre metriche che riassumano il medesimo concetto in un unico numero in modo da poter effettuare confronti tra modelli diversi. Una tra le metriche più importanti

è senza dubbio l'*Accuratezza* che calcola il rapporto tra numero di item le cui etichette sono state predette correttamente e il numero di item totale [70].

$$Accuratezza = \frac{Numero\ di\ predizioni\ corrette}{Numero\ di\ predizioni\ totali}$$

Considerando l'esempio in Tabella 3.1, l'*Accuratezza* sarà calcolata come:

$$Accuratezza = \frac{a+d}{a+b+c+d}$$

Tuttavia questa metrica non è sempre affidabile, in particolar modo quando il problema non è ben bilanciato (cioè quando c'è una classe maggioritaria ed una minoritaria). Ad esempio, si supponga di avere un problema binario in cui gli item appartengono alla classe 0 e 1 secondo le seguenti proporzioni:

- Classe 0 = 9900
- Classe 1 = 100

Se il modello associa tutti i record alla classe 0, l'accuratezza sarà:

$$Accuratezza = \frac{0+9900}{9900+100} = 99\%$$

Osservando il valore che si è ottenuto si può constatare che il modello è molto performante, ma in realtà ha predetto erroneamente tutti i punti della classe 1. Questo fenomeno accade perché l'*Accuratezza* misura la bontà del classificatore in generale e non relativamente alla singola classe. Per valutare, invece, la performance del modello separatamente per classe si introducono il *Richiamo* e la *Precisione* [71]. Il *Richiamo* della classe C si misura come:

$$Richiamo = \frac{Numero\ di\ oggetti\ correttamente\ assegnati\ a\ C}{Numero\ di\ oggetti\ che\ appartengono\ a\ C}$$

Ad esempio, il *Richiamo* della classe 1 sarà:

$$Richiamo\,(C=1) = \frac{a}{a+b}$$

Il *Richiamo* sarà pari ad 1 quando il numero di oggetti assegnati correttamente alla classe 1 coincide con il numero di oggetti che appartengono ad 1 e quindi non sono stati commessi errori nel classificare i record appartenenti a questa classe. La *Precisione* della classe C si calcola come:

$$Precisione = \frac{Numero\ di\ oggetti\ correttamente\ assegnati\ a\ C}{Numero\ di\ oggetti\ assegnati\ a\ C}$$

Quindi, la *Precisione* della classe 1 in esempio sarà pari a:

$$Precisione\,(C=1) = \frac{a}{a+c}$$

Come si può notare la *Precisione* assumerà un valore pari ad 1 quando il numero di oggetti correttamente assegnati alla classe 1 coincide con il numero di oggetti assegnati ad 1: ciò significa che nessun altro elemento appartenente ad altre classi è stato associato erroneamente ad 1, ma l'indicatore non dice nulla circa gli elementi della classe 1 che non sono etichettati correttamente. Infine, un'ultima metrica interessante è l'$F - measure$ che non è altro che la media armonica tra *Precisione* e *Richiamo*:

$$F - measure = \frac{2 * Richiamo * Precisione}{Richiamo + Precisione}$$

Gli indicatori appena introdotti sono tutti da massimizzare e quindi più alto è il loro valore, più il modello di classificazione è affidabile. I diversi tipi di algoritmi di Classificazione sono valutati sulla base di [71]:

- **Accuratezza**: come appena spiegato, questa metrica misura capacità del modello di predire correttamente le etichette. È un indice di bontà complessiva, ma è importante considerare anche le misure di *Precisione*, *Richiamo* ed $F - measure$ che, invece, misurano l'accuratezza del modello rispetto ad una determinata classe e non rispetto a tutte le classi in generale.

- **Efficienza**: si intende l'efficienza in termini di tempi di esecuzione. Negli algoritmi di Classificazione esistono due tempi da considerare: il tempo del training e il tempo per fare la predizione sul test. Generalmente tutti i modelli sono efficienti in fase di predizione, ma possono essere caratterizzati da tempi molto lunghi nel training.

- **Scalabilità**: si valutano i tempi di esecuzione dell'algoritmo all'aumentare della dimensione del training o del numero degli attributi. Se la dimensione del dataset in futuro crescerà occorre valutare la scalabilità, cioè come variano le performance dell'algoritmo con il variare della dimensione del dataset.

- **Robustezza**: è importante vedere se il classificatore è robusto o meno, ovvero se predice correttamente le etichette anche in presenza di rumore.

- **Interpretabilità**: l'utente deve essere in grado di interpretare i risultati dell'algoritmo. Ad esempio, se l'output di un'analisi classifica un determinato soggetto come "cliente inaffidabile", il modello, per essere interpretabile, deve fornire una spiegazione dell'etichetta, cioè deve poter spiegare per quale motivo il soggetto in questione è stato definito tale.

A livello applicativo, bisogna chiedersi di volta in volta quali siano i requirements principali e di conseguenza valutare quale misura sia più importante da considerare e alla quale attribuire una maggiore priorità. Sulla base di questa valutazione, si seleziona l'algoritmo che meglio si addice al problema di cui ci si deve occupare.

Di seguito, si presenteranno gli algoritmi di Classificazione utilizzati nelle analisi oggetto di questo lavoro, indicandone i meccanismi di funzionamento, i vantaggi e gli svantaggi così da poter capire quali si adattino di più a determinate situazioni piuttosto che ad altre.

3.1 Decision Tree

Il *Decision Tree* è un grafo caratterizzato da *nodi* e *archi*. Il nodo più alto è definito nodo radice, ciascun nodo intermedio rappresenta un test effettuato su un determinato attributo e gli archi rappresentano il risultato del test. Gli attributi sulla base dei quali in ciascun nodo intermedio si effettua il test sono chiamati *attributi di splitting* poiché suddividono i dati in sottogruppi. Infine, i nodi foglia (o terminali) contengono l'etichetta di classe o la distribuzione delle classi [61].

Infatti, le foglie possono essere:

- **Pure**: quando tutti i record che ricadono sulla foglia appartengono ad una sola classe.
- **Impure**: quando i record che ricadono sulla foglia appartengono a classi differenti.

L'obiettivo dell'algoritmo è quello di effettuare vari test sui valori degli attributi a partire dal nodo radice, in modo tale che ogni record, sulla base del risultato conseguito in ogni test, percorra un cammino più o meno lungo tra i vari nodi intermedi fino ad arrivare ad un nodo foglia che definisce la sua classe di appartenenza.

Come per tutti gli algoritmi di Classificazione ci sarà una prima fase di training dove avviene la creazione dell'albero sui dati etichettati, la fase di test in cui si valida il modello e poi, successivamente, si potrà utilizzare l'albero per predire le labels dei dati non etichettati se il modello è risultato affidabile.

Ad esempio, si consideri il dataset in Figura 3.2 che riporta per 10 clienti di una banca alcune caratteristiche, quali ad esempio *Refund* che indica se il cliente ha o meno restituito il prestito, lo stato civile e il reddito. L'etichetta *Cheat* indica se il cliente è stato o meno un truffatore. Sulla base di questo dataset di training si vuole creare l'albero delle decisioni per poter successivamente predire l'etichetta di eventuali nuovi clienti, al fine di prevedere se saranno buoni pagatori oppure no. L'albero delle decisioni si costruisce su un dataset etichettato dal quale l'algoritmo apprende le relazioni tra variabili ed etichette. Inizialmente l'attributo di splitting selezionato sul nodo radice è *Refund* ed esso può assumere due valori: "Yes" o "No". Se il record assume il valore "Yes" per l'attributo in questione, allora l'etichetta sarà "NO", cioè il cliente ha restituito il prestito e di conseguenza non

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Figura 3.2: Dataset di esempio [71]

è un truffatore. Invece, se i dati assumono il valore "No" per l'attributo *Refund*, non si può direttamente predire l'etichetta, ma servono ulteriori indagini su altri attributi e quindi si passa ad un nodo intermedio che considera come variabile di splitting lo stato civile. Se il cliente è sposato, allora si può associare direttamente l'etichetta di classe che sarà "NO", mentre se il cliente è single o divorziato si procede con un'ulteriore analisi. Si passa al nodo successivo che considera il reddito e si ripete il meccanismo finché i test effettuati permetteranno di associare ad ogni record un'etichetta.

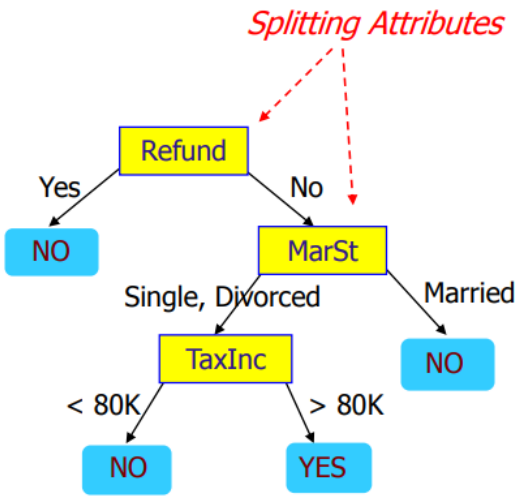


Figura 3.3: Albero creato come modello [71]

Per uno stesso dataset di training si possono creare diversi alberi, ad esempio in Figura 3.4 è rappresentato un albero con una struttura diversa rispetto a quello

in Figura 3.3, ma realizzato a partire dallo stesso dataset. La differenza consiste nell'aver selezionato in ordine diverso gli attributi di splitting.

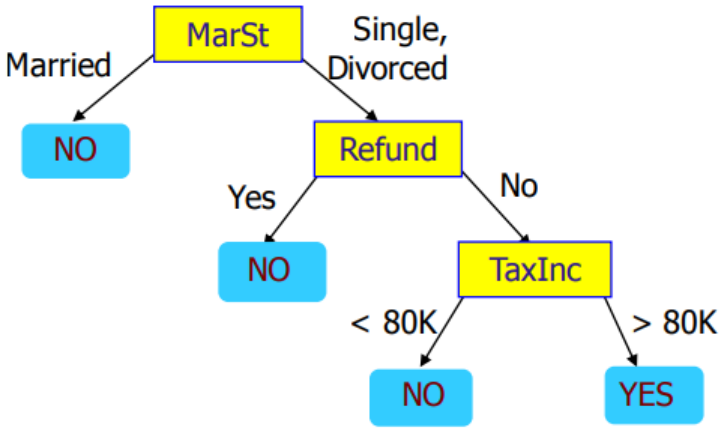


Figura 3.4: Albero alternativo [71]

Terminata la fase di training e quindi di creazione del modello, si passa alla fase predittiva, cioè si vuole associare l'etichetta di classe ad un nuovo cliente del quale non si conosce l'etichetta *Cheat*, ma si hanno le seguenti informazioni:

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Figura 3.5: Nuovo cliente da classificare [71]

Applicando il modello creato in fase di training, sulla base del valore assunto dall'attributo *Refund* e *Marital Status*, si attribuisce l'etichetta *Cheat* ="NO", come rappresentato in Figura 3.6. Per costruire l'albero in fase di *Decision Tree Induction* si possono utilizzare diversi algoritmi, tra i quali, ad esempio, l'**Hunt's Algorithm** [70]. Inizialmente tutto il dataset di training è racchiuso nel nodo radice. Se, all'interno del dataset vi sono record appartenenti ad almeno due classi diverse, l'algoritmo seleziona l'attributo di splitting migliore che genera dei sottogruppi e si crea un nodo intermedio per ognuno di essi. Questo meccanismo di espansione dell'albero è applicato in modo ricorsivo su tutti i nodi che contengono più di un'etichetta. Se un nodo, al contrario, raggruppa item che hanno una sola label, allora non verrà più espanso. Può anche accadere che un nodo con più etichette non sia espanso quando il numero di record su quel nodo è inferiore ad una determinata soglia, in questo caso, generalmente, si associa l'etichetta che si

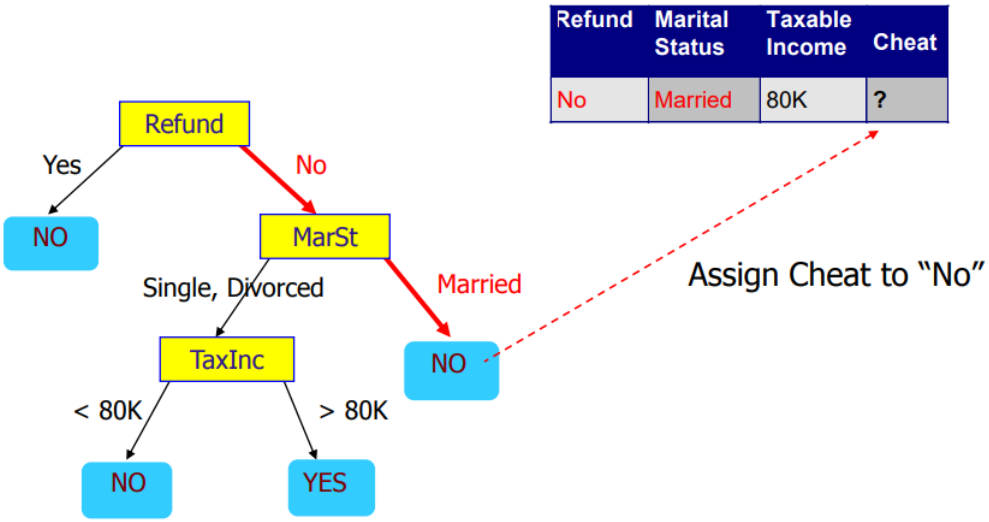


Figura 3.6: Predizione [71]

presenta con maggior frequenza. Un esempio del meccanismo di funzionamento dell'algoritmo è rappresentato in Figura 3.7.

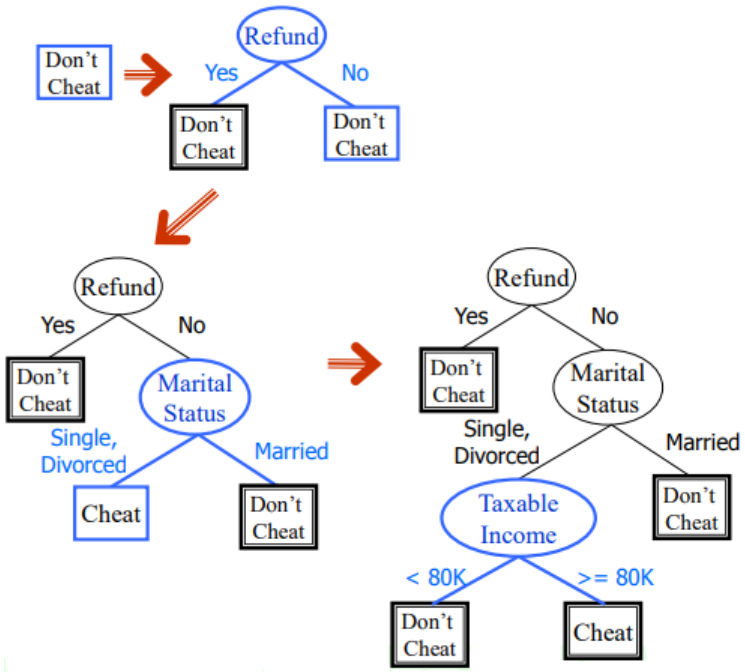


Figura 3.7: Algoritmo di Hunt [71]

L'algoritmo di Hunt, come molti induction algorithms, si basa su un **approccio greedy** per selezionare il *best splitting attribute* e ciò permette di ottenere soluzioni accurate in tempi contenuti. L'approccio greedy consiste nell'individuare localmente,

ad ogni iterazione, l'attributo di splitting che separa meglio i dati di input, cioè quello che crea sotto partizioni il più possibili pure. Gli split possono essere binari (quando a partire da un nodo si generano due sottogruppi), oppure ennari (quando da un nodo si generano n sottogruppi). La tipologia di split dipende dalla natura dell'attributo [70]:

- **Attributi binari**: si generano due sottogruppi e lo split sarà quindi binario.

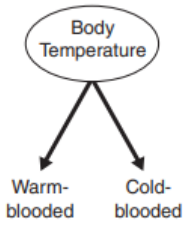


Figura 3.8: Attributi binari [70]

- **Attributi nominali**: questo tipo di attributo può assumere diversi valori e quindi potrà generare sia uno split ennario (dove n corrisponde al numero di valori distinti assunti dall'attributo), sia uno split binario nel caso in cui si partizionino in due gruppi gli n valori.

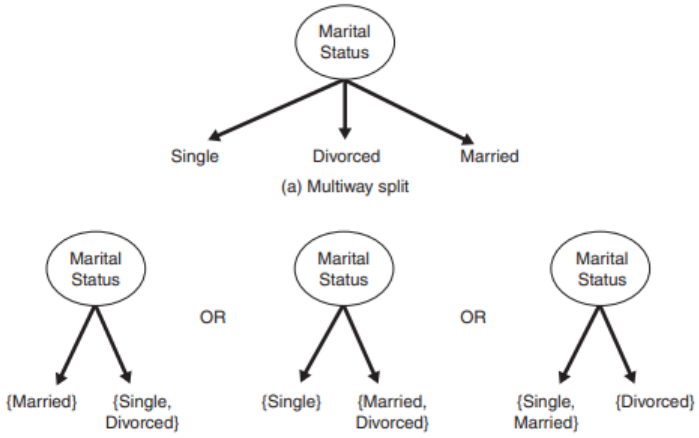


Figura 3.9: Attributi nominali [70]

- **Attributi ordinali**: anche in questo caso si possono generare split binari o ennari poiché i valori possono essere raggruppati purché si rispetti sempre la proprietà dell'ordinamento (in Figura 3.10, ad esempio, lo split c viola tale proprietà).

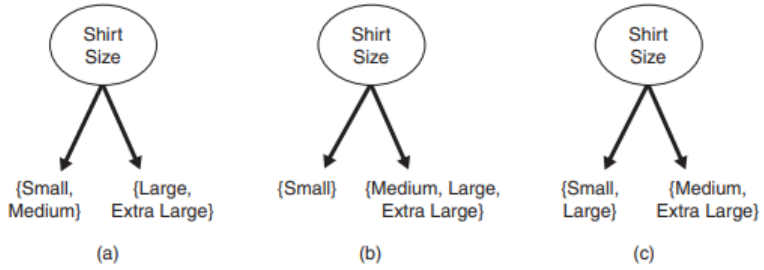


Figura 3.10: Attributi ordinali [70]

- **Attributi continui**: per gli attributi di questo tipo si possono ottenere entrambe le modalità di splitting in base a come si esprimono i valori. Se, ad esempio, si traducono come un confronto ($A < v$) si ottiene uno split binario, altrimenti si può suddividere il dominio della variabile in intervalli mutuamente esclusivi, ad esempio con la discretizzazione, in modo tale da poter associare ad ogni ramo uno specifico range.

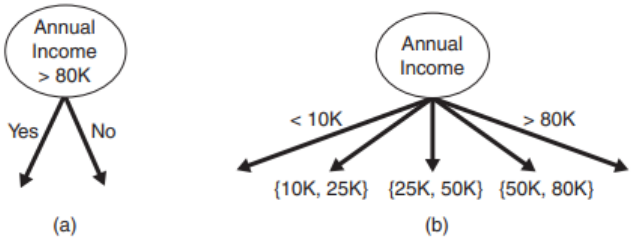


Figura 3.11: Attributi continui [70]

Alla luce di ciò, occorre introdurre le metriche che devono essere calcolate per capire quale sia l'attributo migliore e quale tipologia di splitting si adatti meglio alla situazione. L'obiettivo di queste misure è quello di calcolare l'**impurità** dei nodi affinché si possa prediligere la combinazione che massimizzi la purezza delle foglie. Infatti, se un nodo è puro, esso diventerà foglia. Al contrario, un nodo impuro sarà ancora ulteriormente espanso e questo comporta la creazione di alberi molto profondi. Gli alberi con elevata profondità non sono desiderabili poiché, oltre ad essere di più complessa interpretabilità, potrebbero generare il fenomeno dell'Overfitting, di cui si parlerà a breve [70]. L'impurità rispecchia la distribuzione delle classi all'interno di un nodo. Un'impurità molto bassa fa riferimento ad una distribuzione di classi sbilanciata a favore di una determinata label (che sarà la classe etichettante), al contrario l'impurità massima si ottiene quando i record del nodo appartengono in modo omogeneo ed equilibrato a classi differenti. Per valutare il test su ogni nodo si confronta il punteggio ottenuto dall'indicatore di impurità sul nodo padre con quello del nodo figlio e cioè si verifica come varia

l'impurità prima e dopo lo split. Se la differenza è elevata allora il test è buono, altrimenti no. L'impurità può essere calcolata con i seguenti indicatori in cui j è la classe, t il nodo e $p\,(j|t)$ è la frequenza della classe j nel nodo t:

$$GINI\ \ (t) = 1\ \ - \sum_{j} p[(j|t)]^2$$

$$Entropy\,(t) = -\sum_{j} p(j|t) log_2 p(j|t)$$

$$Error\,(t) = 1 - max_j[p(j|t)]$$

In Figura 3.12, si riporta un esempio numerico da cui si deduce che tutte e tre le misure riportano un'impurità pari a 0 quando un nodo contiene solo un'etichetta di classe, mentre raggiungono il loro valore massimo quando il nodo ha lo stesso numero di istanze per ogni etichetta. Il nodo N_1 risulta quello con impurità più bassa.

Node N_1	Count
Class=0	0
Class=1	6

Gini $= 1 - (0/6)^2 - (6/6)^2 = 0$
Entropy $= -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$
Error $= 1 - \max[0/6, 6/6] = 0$

Node N_2	Count
Class=0	1
Class=1	5

Gini $= 1 - (1/6)^2 - (5/6)^2 = 0.278$
Entropy $= -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$
Error $= 1 - \max[1/6, 5/6] = 0.167$

Node N_3	Count
Class=0	3
Class=1	3

Gini $= 1 - (3/6)^2 - (3/6)^2 = 0.5$
Entropy $= -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$
Error $= 1 - \max[3/6, 3/6] = 0.5$

Figura 3.12: Esempio di calcolo dell'impurità [70]

A seconda della metrica che si utilizza si ottiene una soluzione diversa, in particolar modo quando si ha a che fare con problemi complessi.

A questo punto, occorre domandarsi quando interrompere la crescita dell'albero. Infatti, solitamente l'algoritmo si arresta quando tutti i nodi sono puri oppure quando i record hanno i valori degli attributi molto simili. Potrebbe però essere necessario interrompere lo sviluppo dell'albero per evitare il fenomeno che si crea in presenza di alberi troppo profondi: l'**Overfitting** [71]. Tale fenomeno si genera quando l'albero è troppo specifico e cioè quando possiede un numero di nodi elevato. Il numero dei nodi può crescere in profondità oppure in ampiezza se si effettuano troppi test. All'aumentare del numero dei nodi l'errore sul dataset di training diminuisce, mentre l'errore sul dataset di test inizialmente diminuisce, poi si stabilizza ed infine aumenta perché il modello creato è troppo complesso e specifico per apprendere la vera natura delle relazioni tra attributi ed etichette sul dataset completo [70]. Avere un albero molto profondo e specifico fa sì che esso sia poco distorto (la distorsione è la differenza tra la previsione media del modello e il valore

corretto da prevedere), ma allo stesso tempo avrà una varianza elevata perché sarà molto sensibile alla casualità dei dati all'interno del set di addestramento. In altre parole, un modello che presenta una varianza elevata si concentra troppo sui dati di addestramento e si adatta molto bene ad essi, senza però poter essere generalizzato per i dati che non si conoscono. Per ovviare a questo problema occorre innanzitutto trovare la giusta cardinalità del dataset di training in modo tale che il modello riesca ad apprendere il legame tra attributi ed etichette: per questo motivo aumentare la cardinalità del set di allenamento riduce il rischio di Overfitting in quanto più dati si introducono, più è probabile che il modello sia generale e meno specifico. Inoltre, si possono applicare le tecniche di *pruning*, cioè di potatura dell'albero. In particolare, si parla di *prepruning* quando si interrompe la crescita dell'albero prima che esso si adatti perfettamente ai dati di training, mentre si parla di *postpruning* quando si costruisce l'albero completo e successivamente si rimuovono i nodi in modo bottom-up [70]. Il problema opposto all'Overfitting è l'**Underfitting** e si origina

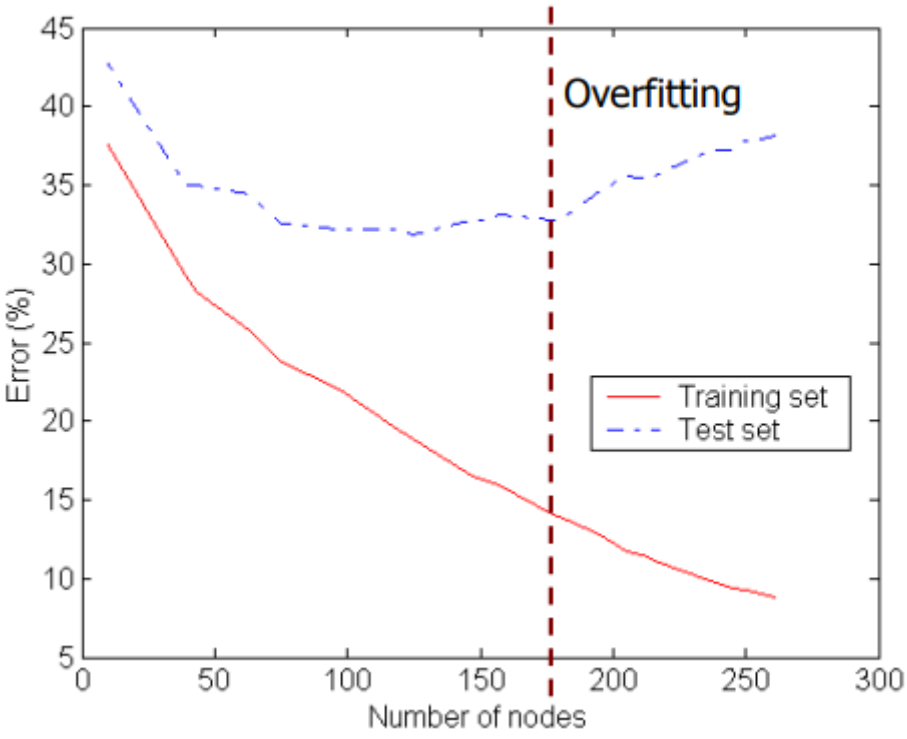


Figura 3.13: Fenomeno dell'Overfitting [70]

quando l'albero creato come modello è troppo semplice e quindi non è in grado di rappresentare la relazione tra classi e attributi. Il fenomeno dell'Underfitting è tipico di quelle situazioni in cui si utilizza una porzione di dati come training set non sufficiente per costruire un modello accurato. Oltre ai problemi dell'Overfitting e dell'Underfitting, un altro svantaggio che caratterizza gli alberi decisionali è che, in presenza di missing values, l'accuratezza diminuisce. Allo stesso tempo, però,

garantiscono buone performance nella maggior parte dei problemi di classificazione senza essere troppo onerosi dal punto di vista computazionale grazie all'approccio greedy. Gli alberi decisionali sono abbastanza robusti rispetto alla presenza di attributi ridondanti e rumore, sono veloci nel classificare nuovi record e soprattutto garantiscono una facile interpretabilità poiché basta percorrere l'albero per dare una spiegazione alle etichette di classe [71][72].

3.2 Random Forest

L'algoritmo *Random Forest* o delle *Foreste Casuali* è un potente strumento di Data Mining in quanto può essere definito un *ensamble method*, cioè sfrutta la tecnica dell'*Ensable Learning*. Essa consiste nell'applicazione di modelli multipli per ottenere una soluzione più accurata. In particolare, sotto il termine Ensable Learning si celano più tecniche, in questo caso, quella su cui si costruisce l'algoritmo del Random Forest, è la tecnica *Bagging* o *Bootstrap Aggregation* che consiste nell'addestrare più modelli dello stesso tipo su dataset differenti che sono ricavati dal dataset di training attraverso operazioni di campionamento casuale con remissione. L'algoritmo della Foresta Casuale, come si può dedurre dal nome, utilizza l'albero decisionale come modello individuale e quindi l'algoritmo prevede la creazione di molti alberi decisionali su differenti sottoinsiemi del dataset di training e il risultato finale non sarà altro che la classe restituita con maggior frequenza dagli alberi creati [73]. L'idea è che ogni singolo albero potrebbe fare previsioni non accurate, ma combinando tante previsioni, la soluzione sarà certamente migliore. Questo accade perché il Bagging calcola la media di molti classificatori instabili, ma non distorti e questo riduce la varianza. Gli alberi decisionali, in questo senso, sono i candidati perfetti, poiché possono catturare relazioni complesse tra le variabili e quindi sono poco distorti e, allo stesso tempo, si risolve il problema dell'Overfitting con la riduzione della varianza. Il modello finale, infatti, sarà più generale e meno specifico rispetto al risultato ottenuto dai singoli alberi decisionali [74]. Inoltre, il Random Forest, introduce un'ulteriore componente di casualità oltre a quella dovuta alla creazione di sotto partizioni casuali del dataset di training e cioè considera solo una parte di features per suddividere ciascun nodo dell'albero (*feature bagging*). Tipicamente, se p sono gli attributi del dataset, l'algoritmo ne seleziona $m = \sqrt{p}$ oppure $m = \log_2 p$. Ad esempio, se il dataset di training è caratterizzato da 16 attributi, l'algoritmo potrebbe considerare solo 4 scelti in modo casuale per dividere il nodo. Di conseguenza, l'attributo di splitting sul nodo non sarà il miglior attributo tra tutti quelli del dataset, ma sarà il migliore rispetto al sottoinsieme di features selezionate. Questo meccanismo comporta una riduzione della correlazione tra coppie di alberi e ciò si traduce in una riduzione della varianza della media

[73][74]. Chiamando B il numero degli alberi, i passi dell'algoritmo possono essere riassunti nel seguente modo:

$Per\ b = 1, ...B$:

1. Estrae un campione bootstrap Z di dimensione N dal training set.
2. Crea un albero sulla base di questo campione ripetendo in modo ricorsivo i seguenti passaggi per ogni nodo fino al raggiungimento della dimensione minima dei nodi:
   (a) Seleziona m attributi tra i p esistenti ($m < p$)
   (b) Seleziona l'attributo migliore per lo splitting tra gli m selezionati
   (c) Divide il nodo

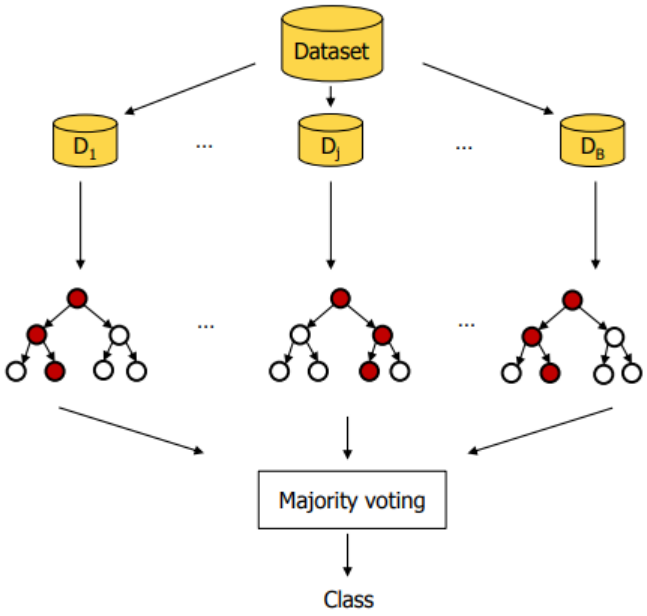


Figura 3.14: Funzionamento del Random Forest [71]

Si può concludere che i vantaggi di questa metodologia sono:

- Effettua una predizione a bassa varianza e stabile rispetto al variare dei dati di input.
- Migliora le prestazioni del singolo albero perché combina molti alberi assieme.
- Non è un metodo che comporta evidenti rallentamenti nella sua applicazione poiché ogni albero si crea sempre su un sottoinsieme di dati e di attributi.
- Non si generano problemi di Overfitting.

- Non risente eccessivamente della presenza di noise e outliers.

- Misura l'importanza delle features, cioè fornisce una stima delle caratteristiche più importanti nella classificazione.

Il punto di debolezza del metodo riguarda il fatto che il risultato perde la facile interpretabilità che invece caratterizza l'algoritmo del Decision Tree. Inoltre, il Bagging, se da un lato comporta la riduzione della varianza, dall'altro aumenta la distorsione e dunque la capacità del modello di attribuire le etichette correttamente peggiorerà leggermente [71][75].

3.3 K-Nearest Neighbors

Uno degli algoritmi più conosciuti e più semplici nel mondo del machine learning è senza dubbio il *K-Nearest Neighbors*, detto anche *KNN*. La particolarità di questo algoritmo di apprendimento supervisionato è che, a differenza degli altri classificatori non prevede la creazione di un modello, ma la fase di training consiste semplicemente nella memorizzazione dei valori assunti dalle caratteristiche e delle etichette. Il suo funzionamento si basa sul calcolo della distanza che c'è tra il record di cui si vuole predire l'etichetta e i K elementi del dataset a lui più vicini. Il record verrà etichettato sulla base delle labels dei K neighbors selezionati. Poiché l'algoritmo si basa sul concetto di distanza, è importante normalizzare i dati in fase di preprocessing affinché la sua misura non sia dominata da uno degli attributi presenti nel dataset. Ad esempio, se c'è l'attributo altezza, esso ha presumibilmente un range di valori compreso tra 1,5 m e 2.0 m, ma nello stesso dataset potrebbero esserci attributi come il reddito, con un range compreso tra 10K e 1M di euro e quindi questa differenza nei range assunti dalle variabili influirebbe notevolmente sul calcolo della distanza. Il modello, per il suo funzionamento richiede:

- Un set di dati storici etichettati che saranno utilizzati per il calcolo della distanza dall'unseen case.

- Una metrica per calcolare la distanza tra i record. Spesso si usa la distanza Euclidea $d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$

- Il valore di K che rappresenta il numero di vicini che si vuole considerare nel calcolo della distanza. Ad esempio, se $K = 1$, il record è attribuito alla classe del suo vicino. In Figura 3.15 si può osservare un esempio in cui si ricercano i vicini di un "unknown record" selezionandone dapprima uno (a), poi due (b) ed infine 3 (c).

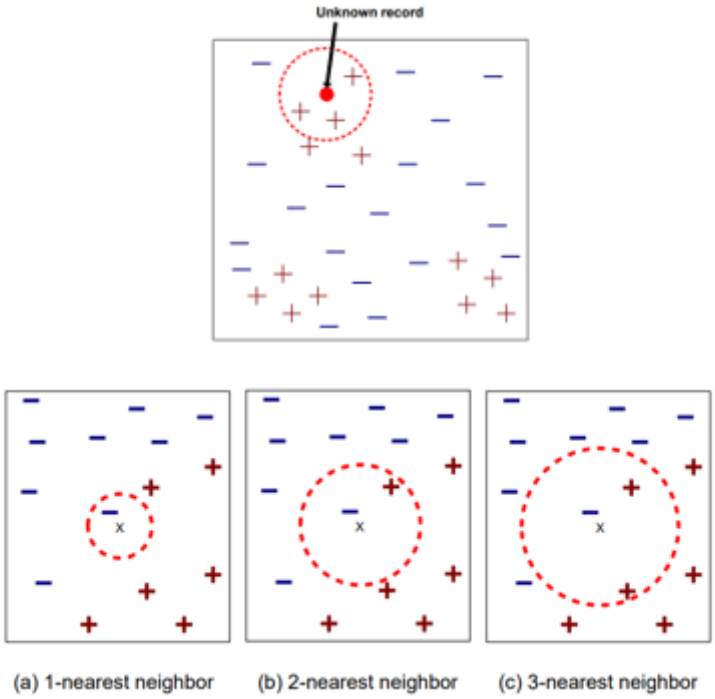


Figura 3.15: K-Nearest Neighbors

Il modello, ricevuti questi input, calcola la distanza tra l'unknown record e i punti del dataset di training, ordina le distanze in modo crescente e seleziona i primi K elementi più vicini, dopodiché attribuisce l'etichetta al record sulla base delle classi dei neighbors.

La label è assegnata secondo un processo di majority voting tra i K elementi più vicini al parametro da stimare. Se si attribuisce lo stesso peso a tutti i vicini, l'algoritmo risulta molto sensibile al parametro K e c'è anche l'inconveniente dovuto alla predominanza delle classi con più oggetti. Per ovviare a questi problemi si può pesare il contributo di ogni vicino con la distanza, moltiplicandolo per il fattore $w = \frac{1}{d^2}$ in modo da dare più importanza ai punti più vicini.

La difficoltà principale di questo algoritmo consiste nel trovare il valore di K ideale da passare come input. Se K è abbastanza elevato si riduce il rumore che compromette la classificazione, ma c'è maggiore probabilità di includere punti appartenenti ad altre classi. Se, al contrario, K è troppo piccolo, l'approccio risulta sensibile al rumore poiché ci si concentra su una regione limitata e il classificatore conoscerà molto poco la distribuzione generale. Tra i metodi più utilizzati per trovare il valore di K ottimale c'è la Cross-Validation di cui si parlerà in seguito.

Il KNN è un algoritmo che può richiedere tempi di esecuzione molto elevati. Il tempo necessario sarà direttamente proporzionale al valore di K scelto come input e dipende, ovviamente, anche dalla dimensionalità del dataset che influisce

negativamente sulle prestazioni dell'algoritmo, non a caso si parla di *curse of dimensionality*, cioè la maledizione della dimensionalità [65].

3.4 Classificazione Bayesiana

La *classificazione Bayesiana* è un tipo di classificazione che si basa sul calcolo della probabilità che un determinato record appartenga ad una classe invocando il *Teorema di Bayes* (che definisce la probabilità condizionata di un evento rispetto ad un altro). Anche in questo caso, l'algoritmo ha necessità di crearsi un modello in fase di training per poter predire le nuove etichette sui dati sconosciuti.

Indicando con $C = C_1, C_2, C_3, \ldots, C_s$ l'insieme delle classi e con $X = x_1, x_2, \ldots x_k|$ il record che si deve etichettare caratterizzato da k attributi, si vuole calcolare la probabilità che il record X appartenga alla classe C e quindi si vuole stimare $P(C|X)$ cioè la probabilità che, avendo osservato X, esso appartenga a C. Questo valore deve essere calcolato per ogni classe e, successivamente, si attribuisce il record alla classe a cui è associato il massimo valore di $P(C|X)$. In altre parole, si assegna ad X l'etichetta b tale che

$$P\left(C_b|X\right) = max_{i=1,\ldots s} P(C_i|X)$$

Secondo il *Teorema di Bayes*, la probabilità che, avendo osservato X, esso appartenga a C si calcola nel seguente modo:

$$P\left(C|X\right) = \frac{P\left(X|C\right) * P(C)}{P(X)}$$

Dove:

- $P(C)$: è la **probabilità a priori** cioè la probabilità, indipendentemente dall'osservazione, che il prossimo record appartenga alla classe C. Questo valore è facilmente calcolabile come rapporto tra numero di record che hanno la classe C nel training set e numero di record totali: $P(C) = \frac{N_C}{N}$
- $P(X)$: è la **probabilità assoluta** di X, cioè la probabilità che il prossimo oggetto da classificare sia X ed è un valore costante per ogni classe.
- $P\left(X|C\right)$: è la **probabilità condizionata** di X dato C, cioè la probabilità che il prossimo record sia X sotto l'ipotesi che la sua classe di appartenenza sia C. Il problema è riuscire a stimare questo valore. Per farlo, si utilizza l'assunzione *Naive Bayes* che sfrutta l'ipotesi semplificativa di *indipendenza degli attributi* $x_1, x_2, x_3, \ldots, x_k$ (l'effetto di un attributo su una data classe è indipendente dai valori degli altri attributi):

$$P\left(x_1, x_2, x_3, \ldots, x_k|C\right) = P\left(x_1|C\right) P\left(x_2|C\right) \ldots P(x_k|C)$$

Dove $P(x_k|C) = \frac{|x_{kC}|}{N_C}$ cioè il rapporto tra il numero di record che hanno attributo k che assume il valore x_k e con etichetta C e il numero totale di record con etichetta C.

Si consideri come esempio il dataset di training in Figura 3.16, il modello si costruisce sui dati etichettati e si calcola la probabilità di ciascuna classe e la probabilità condizionata sugli attributi $P(x_k|C)$:

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

outlook	
P(sunny|p) = 2/9	P(sunny|n) = 3/5
P(overcast|p) = 4/9	P(overcast|n) = 0
P(rain|p) = 3/9	P(rain|n) = 2/5
temperature	
P(hot|p) = 2/9	P(hot|n) = 2/5
P(mild|p) = 4/9	P(mild|n) = 2/5
P(cool|p) = 3/9	P(cool|n) = 1/5
humidity	
P(high|p) = 3/9	P(high|n) = 4/5
P(normal|p) = 6/9	P(normal|n) = 2/5
windy	
P(true|p) = 3/9	P(true|n) = 3/5
P(false|p) = 6/9	P(false|n) = 2/5

P(p) = 9/14
P(n) = 5/14

Figura 3.16: Esempio classificazione Bayesiana [71]

Si consideri poi un nuovo record non etichettato da classificare: $X = rain, hot, high, false$
Si deve calcolare $P(C|X)$ per entrambe le classi:

Classe P

$$P(X|P)*P(P) = P(rain|P)*P(hot|P)*P(high|P)*P(false|P)*P(P) = 0.010582$$

Classe N

$$P(X|N)*P(N) = P(rain|N)*P(hot|N)*P(high|N)*P(false|N)*P(N) = 0.018286$$

Il valore di $P(N|X)$ è maggiore rispetto a $P(P|X)$, per cui il record è etichettato con classe N. Il più significativo vantaggio del classificatore Bayesiano consiste nella sua semplice e veloce applicabilità. Inoltre, ha prestazioni ottime se vale l'assunzione di indipendenza, però, essa risulta spesso decisamente irrealistica e, in questi casi, le previsioni non sono accurate ed affidabili [71].

3.5 Support Vector Machine

Il *Support Vector Machine*, chiamato anche *SVM*, è un algoritmo di apprendimento automatico supervisionato che è utilizzato per scopi di classificazione. Il suo obiettivo è quello di trovare l'iperpiano che suddivida meglio i dati nelle rispettive classi. Come gli altri algoritmi di Classificazione, è necessaria la creazione di

un modello a partire da una porzione di dati etichettati che sono utilizzati per l'individuazione nello spazio delle diverse aree a cui appartengono i punti del dataset. L'obiettivo è individuare aree che siano il più possibili distanti l'una dall'altra. Per meglio comprendere il suo funzionamento occorre introdurre alcuni concetti [76][77]:

- **Iperpiano**: formalmente, un iperpiano è definito come un sottospazio di dimensione $n-1$ in uno spazio di dimensione n. Ad esempio, se si considera il piano cartesiano a due dimensioni (x, y) , l'iperpiano sarà rappresentato da una retta che separa i punti in due semipiani. In uno spazio tridimensionale si avrà un piano che separa i punti in due semispazi. Se le dimensioni aumentano anziché parlare di linea retta o piano si parla di "iperpiano". In uno spazio n-dimensionale, l'iperpiano è rappresentato dall'equazione lineare $a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$ e i due semispazi saranno individuati da:

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n < b$$

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n > b$$

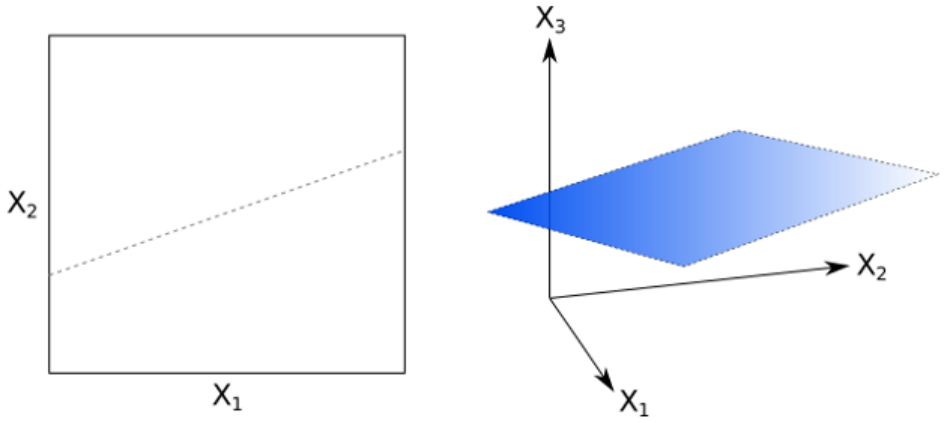


Figura 3.17: Iperpiano [76]

- **Support Vectors**: sono i punti che si trovano più vicini all'iperpiano e quindi sono i punti che si avvicinano maggiormente ad un'altra classe. Essi influenzano la posizione e l'orientamento dell'iperpiano e quindi, cambiando i support vectors, si cambia anche il posizionamento dell'iperpiano.

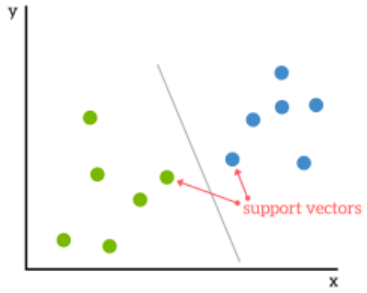


Figura 3.18: Support Vectors [76]

- **Margine**: è la distanza tra i support vectors di due classi differenti più vicine all'iperpiano.

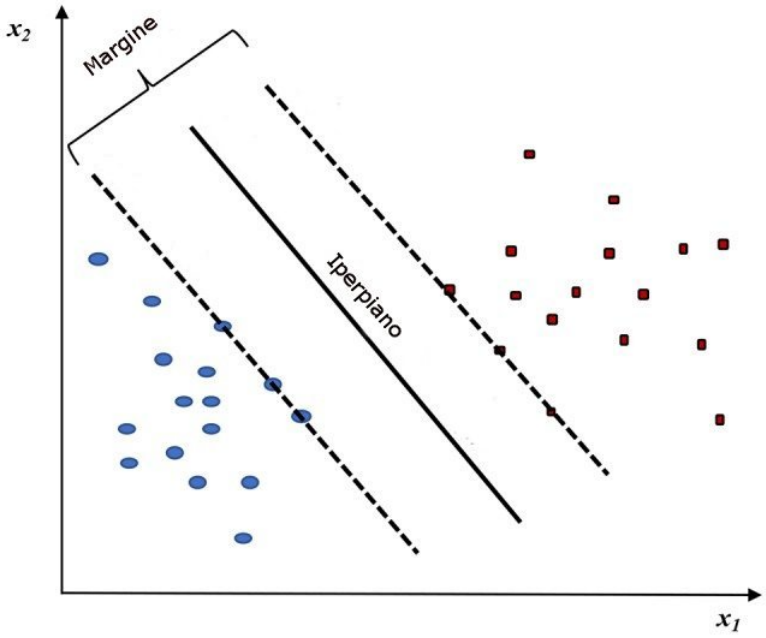


Figura 3.19: Margine [76]

L'obiettivo dell'algoritmo è quello di trovare l'iperpiano che meglio divida i punti in classi. Per fare ciò esegue i seguenti step [76]:

1. Cerca un iperpiano linearmente separabile per suddividere i valori di classi diverse. Gli iperpiani possono essere moltissimi (Figura 3.20) e l'algoritmo seleziona quello che massimizza il margine per migliorare l'accuratezza del modello. In altre parole, l'obiettivo è selezionare il piano che ha la distanza massima tra i punti delle classi. Questo criterio fa sì che i punti futuri saranno classificati correttamente con maggiore probabilità. In Figura 3.21 si possono osservare due differenti iperpiani trovati a partire dagli stessi

dati. I due iperpiani sono caratterizzati da margini differenti, uno minore e l'altro maggiore. Si sceglie quello che massimizza il margine per migliorare l'accuratezza del modello.

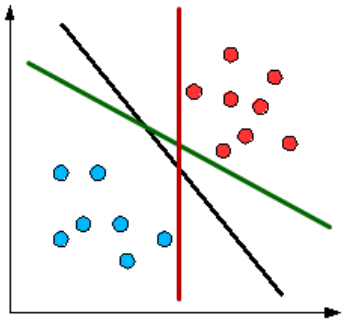


Figura 3.20: Possibili iperpiani [76]

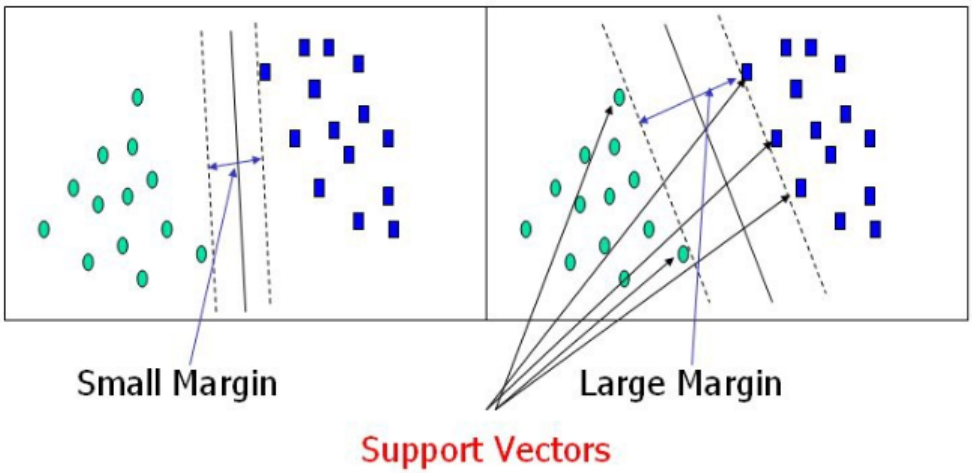
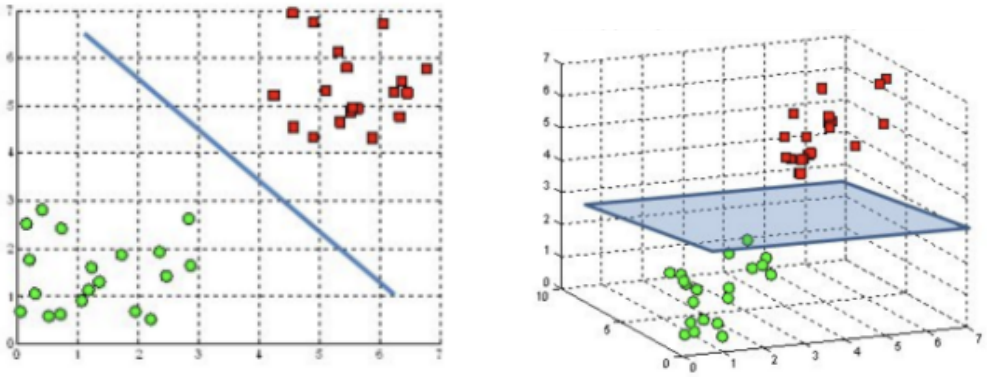


Figura 3.21: Margini differenti [77]

2. Se l'iperpiano linearmente separabile non esiste, allora si utilizza una mappatura non lineare e quindi si aumentano le dimensioni dei dati (da due a tre, da tre a quattro e così via). Successivamente, si trova l'iperpiano che suddivida meglio i dati. In Figura 3.22 sono messi a confronto due iperpiani, il primo è in uno spazio 2D, il secondo 3D.

Entrando più nel dettaglio, se il dataset in questione non è separabile linearmente (come quello rappresentato in Figura 3.23), si può ricorrere al metodo o trucco del *Kernel* che permette la creazione di modelli non lineari.



(a) *Spazio 2D* **(b)** *Spazio 3D*

Figura 3.22: Ricerca dell'iperpiano [77]

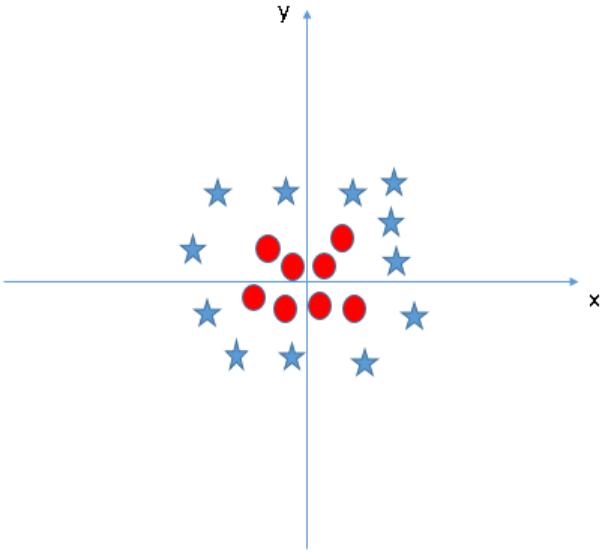


Figura 3.23: Dati non linearmente separabili

Il metodo *Kernel* prevede l'aggiunta di una nuova dimensione e cioè trasforma i dati ricevuti come input nella forma richiesta in quei casi in cui non si riesca a trovare un iperpiano linearmente separabile. Il *Kernel* è definito come:

$$K(x, y) =< f(x), f(y) >$$

dove K indica la funzione del *Kernel*, x e y sono i vettori di input di dimensione n. La funzione f è utilizzata per mappare l'input dallo spazio n-dimensionale a quello m-dimensionale, dove $m > n$. Il simbolo <> rappresenta il prodotto scalare. Le tipologie di *Kernel* possono essere riassunte come segue [76]:

- **Kernel lineare**: è il più semplice e funziona particolarmente bene per i problemi di classificazione sui testi.

$$K(x_i, y_i) = x_i * y_i$$

- **Kernel polinomiale**: è caratterizzato da una costante c e da un grado di libertà d. Se d assume il valore 1 allora il $Kernel$ polinomiale coincide con quello lineare.

$$K(x_i, y_i) = (x_i * y_i + c)^d$$

- **Kernel RBF**: detto anche $Kernel$ gaussiano, contiene un parametro γ. Più il valore di γ è piccolo, più il modello è simile ad un SVM lineare, più è elevato più il modello sarà influenzato dai vettori di supporto.

$$K(x_i, y_i) = e^{(-\gamma||x_i - y_i||^2)}$$

L'utilizzo del $Kernel$ rende questo tipo di algoritmo molto versatile e adatto a svariati contesti in cui la separazione delle classi non è di tipo lineare. Inoltre, poichè nel processo decisionale sono considerati solo un sottoinsieme di punti, cioè i vettori di supporto, l'SVM risulta efficiente dal punto di vista dell'utilizzo della memoria. Gli svantaggi principali dell'algoritmo riguardano il fatto che i risultati non sono facilmente interpretabili e quindi occorre ricorrere a strumenti di visualizzazione grafica [76].

3.6 Cross-Validation

La *Cross-Validation* o validazione incrociata è una tecnica statistica che è utilizzata per suddividere il dataset in training e test e quindi per validare il modello. Essa suddivide in modo casuale i dati in K porzioni della stessa dimensione e, ad ogni iterazione, la K-esima parte è utilizzata come test, mentre la restante parte come training. In altre parole, la Cross-Validation, dopo aver suddiviso il dataset in K porzioni, applica l'algoritmo di classificazione su tutte le possibili K combinazioni di training e test per valutare quale tra queste garantisce maggior accuratezza. Si supponga, ad esempio, di avere un dataset D e di dividerlo con $K = 3$, si ottengono tre sotto partizioni casuali della stessa dimensione: $S1$, $S2$, $S3$. La Cross-Validation sarà caratterizzata da 3 iterazioni:

1. Run 1: $S1$ è utilizzato come test, $S2$ ed $S3$ sono utilizzati come training. In questa iterazione è calcolato anche l'errore su $S1$ che si denota come $err(S_1)$.

2. Run 2: $S1$ ed $S3$ sono utilizzati come training ed $S2$ come test. L'errore calcolato sarà $err(S_2)$.

3. Run 3: $S1$ ed $S2$ sono utilizzati come training ed $S3$ come test. L'errore calcolato sarà $err(S_3)$.

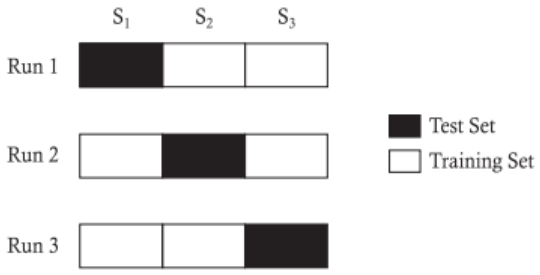


Figura 3.24: Cross-Validation [70]

L'errore totale si ottiene sommando $err(S_1)$, $err(S_2)$, $err(S_3)$ e dividendo per il numero totale di record. Questo approccio è definito *Three-Fold Cross-Validation*, poiché K è assunto pari a 3, ma, generalizzando, questo metodo prende il nome di **K-Fold Cross-Validation** e l'errore totale sarà calcolato come:

$$err_{test} = \frac{\sum_{i=1}^{k} err_{sum}(i)}{N}$$

Il valore ottimale di K da utilizzare nella K-Fold Cross-Validation dipende dal numero di caratteristiche del dataset. Un valore di K piccolo comporta un training set di ridotta dimensione e ciò si traduce in una stima più ampia dell'errore. Al contrario se si stabilisce un K più elevato si otterranno ad ogni iterazione dei training set di dimensione maggiore e ciò comporta una riduzione del bias nella stima dell'errore.

Un caso estremo della K-Fold Cross Validation è l'approccio **Leave-one-out**, in cui K è settato pari al numero dei record totali ($K = N$) e quindi ad ogni iterazione si utilizzerà esattamente un'unica istanza come test e le rimanenti $N - 1$ come training. In questo tipo di approccio, si utilizza la dimensione massima possibile di training set, ma allo stesso tempo il metodo potrebbe risultare molto costoso in termini computazionali se il dataset ha dimensioni elevate. La K-Fold Cross Validation non garantisce, però, che in ogni fold siano rappresentante le corrette proporzioni di dati appartenenti a classi diverse. Nel caso in cui, si voglia rispettare questa proporzione, si applica la **Stratified Cross Validation**: con questo metodo, ogni fold generato contiene la stessa percentuale di dati per ogni etichetta del dataset totale. Dunque, ogni etichetta sarà ben rappresentata in ogni partizione [70].

Capitolo 4

Risultati Sperimentali

4.1 Strumenti utilizzati

Si presentano ora i risultati sperimentali ottenuti applicando le metodologie discusse nel capitolo precedente.

Prima però, si effettua una breve panoramica sugli strumenti utilizzati a supporto di queste analisi.

Python Il linguaggio di programmazione utilizzato è Python che si presta particolarmente al tipo di analisi effettuato in quanto esistono numerose librerie dedicate alla manipolazione dei dati e all'implementazione degli algoritmi di Data Mining. Si tratta di un linguaggio ad alto livello orientato ad oggetti. La piattaforma sulla quale ci si è appoggiati è Anaconda che include le principali librerie utili per la data science. Tale piattaforma comprende anche l'ambiente di sviluppo Jupyter che è il notebook virtuale impiegato per implementare gli algoritmi e per la visualizzazione grafica dei risultati. Le librerie utilizzate sono:

- Pandas: è una libreria che fornisce strumenti per manipolare i dati, in particolar modo quelli che presentano una struttura chiamata Dataframe che consiste in una sorta di tabella molto simile ad un database. Il Dataframe è provvisto di indice ed è particolarmente comodo da utilizzare per effettuare le analisi. Inoltre, la libreria possiede tools per la lettura e scrittura di dati in diversi formati (JSON, CSV, ecc) [78].

- Matplotlib: è una libreria che implementa la creazione dei grafici [79].

- Scikit-learn: è la libreria per l'apprendimento automatico. Essa contiene i moduli e le funzioni per implementare gli algoritmi di Classificazione, Clustering e Regressione [80].

JSON I dati manipolati sono in formato JSON (JavaScript Object Notation). Le caratteristiche di questo formato testuale consistono nel fatto che i dati sono contenuti all'interno di oggetti delimitati da parentesi graffe, sono rappresentati sotto forma di coppie chiave-valore e ciascuna coppia è separata da ",". Ogni chiave è seguita da ":" e sia le chiavi che i valori sono racchiusi tra le virgolette. Inoltre, il formato prevede anche le parentesi quadre le quali definiscono matrici che a loro volta contengono numerosi oggetti al loro interno. Per la sua semplicità, il formato JSON è uno strumento molto utilizzato per lo scambio dei dati tra applicazioni [81].

4.2 Caso di studio

Il caso di studio, come già accennato, riguarda il monitoraggio di un robot di un'importante azienda. I dati sono acquisiti da sensori appositamente utilizzati per controllare il funzionamento del robot, al fine di rilevare eventuali anomalie attraverso un programma di manutenzione predittiva basato sugli algoritmi di Classificazione. L'obiettivo è, a partire dai dati riguardanti la corrente consumata (misurata in Ampere) dal robot durante vari cicli produttivi, valutare se gli algoritmi di Classificazione sono in grado di predire i guasti, individuando i valori anomali di corrente e quindi identificando tempestivamente ed in modo automatico i malfunzionamenti che potrebbero essere dovuti all'errato tensionamento della cinghia. Questo tipo di programma offre un prezioso supporto all'azienda che può intervenire prima che si verifichi un vero e proprio guasto, migliorando la propria efficienza. In questo modo, infatti, si interviene sul macchinario solo nei casi in cui sia strettamente necessario, riducendo i fermi produttivi e i costi legati a manutenzioni non indispensabili.

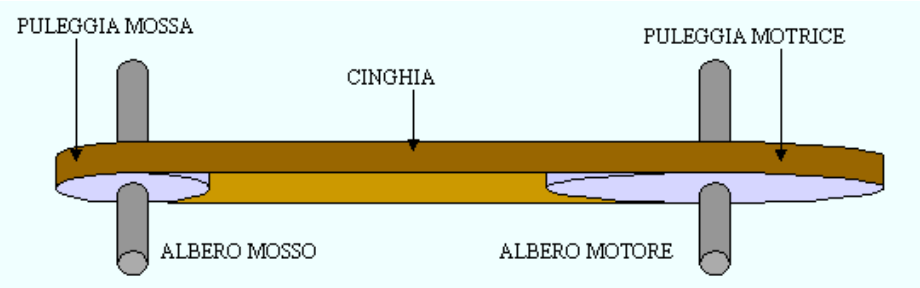


Figura 4.1: Funzionamento di una cinghia [82]

Le cinghie sono utilizzate, insieme alle pulegge, per trasmettere potenza tra due alberi, anche a livelli elevati. Esse non sono altro che strisce di cuoio o di altri materiali flessibili che vengono avvolte ad anello intorno alle pulegge, una delle

quali è definita "motrice" mentre l'altra è messa in movimento dall'attrito con la cinghia.

Esistono diversi tipi di cinghie, ad esempio quelle piatte, tonde, trapezoidali, scanalate e sincrone. Il loro vantaggio è che permettono la trasmissione di potenza in modo uniforme e senza rumore, assorbono urti e variazioni improvvise del carico e necessitano di una manutenzione minima, al contrario dei classici sistemi di trasmissione ad ingranaggi. Lo svantaggio principale è che sono caratterizzate da una bassa rigidità e resistenza per via dei materiali di cui sono composte [82].

Il problema che si vuole analizzare è conosciuto come *Bel Tensioning*, in quanto la causa più rilevante del malfunzionamento delle cinghie risiede in un tensionamento improprio, motivo per il quale risulta importante valutare con attenzione questo aspetto poiché una tensione troppo bassa causa slittamenti, surriscaldamento e una prematura usura della cinghia. Al contrario, l'eccessivo tensionamento danneggia le cinghie, i cuscinetti e gli alberi. La tensione deve essere una giusta via di mezzo che corrisponde al valore più basso per il quale la cinghia non scivola e non stride sotto la massima condizione di carico, anche se esiste un ampio range intorno a questo valore ottimale che garantisce un funzionamento corretto [83].

Per monitorare il funzionamento della cinghia si osservano i valori di corrente consumata dal robot poiché l'effetto di un errato tensionamento si ripercuote sui consumi energetici.

Il robot è caratterizzato da cicli di produzione della durata di circa 24 secondi. Durante questo periodo, sono effettuate 11967 rilevazioni, cioè circa una ogni 2 ms. In particolare, i dataset oggetto dello studio sono due: *Gray* e *White*. Essi sono in formato JSON MIMOSA ed entrambi sono stati ricavati a partire dal monitoraggio dello stesso robot. I due dataset hanno la medesima struttura: ogni ciclo è univocamente identificato dal timestamp e, per ogni timestamp, si hanno a disposizione 11967 rilevazioni di corrente.

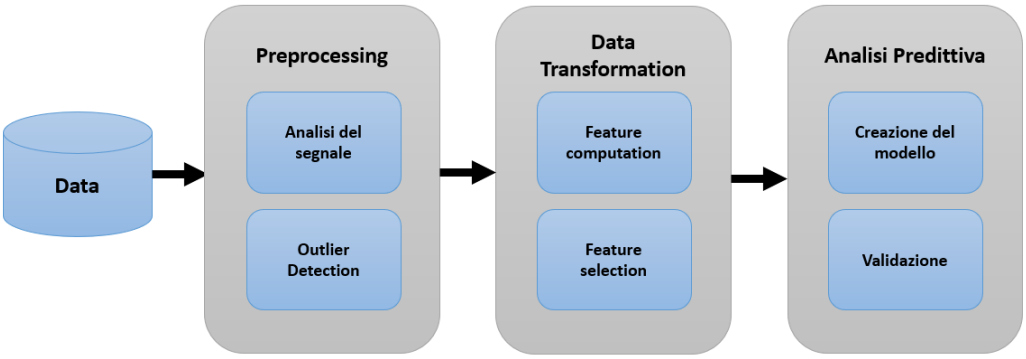


Figura 4.2: Architettura del modello predittivo

Nella prima parte di questa ricerca, ci si concentra sulla fase di *Preprocessing*

in cui si osserva e si studia il segnale di corrente e si individuano i valori anomali. Successivamente, si passa alla fase di *Data Transformation* in cui il segnale è suddiviso in split e si calcolano le features che saranno poi selezionate in base al grado di correlazione. Infine, si presentano i risultati ottenuti in fase di *Analisi predittiva* applicando gli algoritmi di Classificazione presentanti nel capitolo precedente, al fine di costruire modelli di predizione adeguati che saranno poi validati per capire quali meglio soddisfino i requisiti.

4.3 Preprocessing

Andamento del segnale

Il primo importante step consiste nello studio dei dati e quindi nell'osservazione dell'andamento del segnale che caratterizza ciascun ciclo di produzione del robot. All'interno del dataset *Gray* si hanno a disposizione 6019 cicli, mentre in *White* 5729. I timestamp che identificano univocamente ciascun ciclo si riferiscono in entrambi i casi ad un intervallo temporale compreso tra 24-02-2020 e 04-03-2020. I record sono etichettati con le labels 0, 10, 15.

Nelle Tabelle 4.1 e 4.2 è indicata la distribuzione delle etichette all'interno dei dataset *Gray* e *White*, in entrambi i casi si evince che la classe più numerosa è la 10:

Etichetta	# Cicli	Dataset %
0	1287	21,4%
10	3419	56,8%
15	1313	21,8%

Tabella 4.1: Gray

Etichetta	# Cicli	Dataset %
0	1216	21,2%
10	3275	57,2%
15	1238	21,6%

Tabella 4.2: White

Nella Figura 4.3 è raffigurato l'andamento del segnale di corrente durante un ciclo di lavorazione. Si può notare che può essere diviso in quattro aree: la prima fase è caratterizzata da oscillazioni, che, nella seconda fase, si stabilizzano e il segnale assume il valore di 1 A. La terza fase ha avvio con il segnale della corrente

che crolla per poi iniziare nuovamente ad oscillare. Infine, nella quarta ed ultima fase, il segnale torna a stabilizzarsi e assume un valore pari a -1 A.

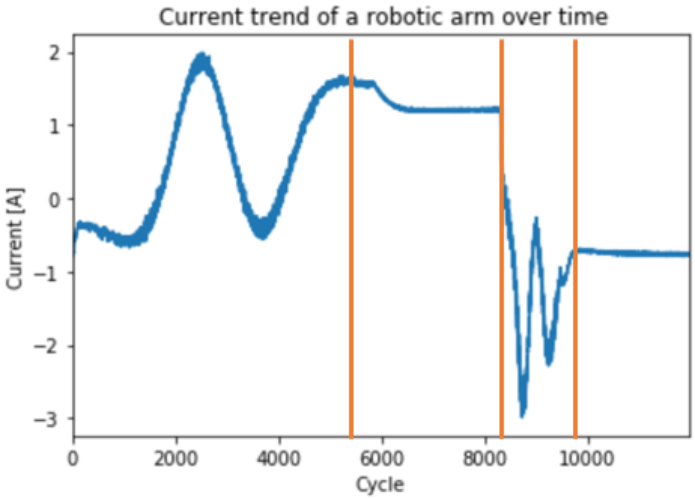


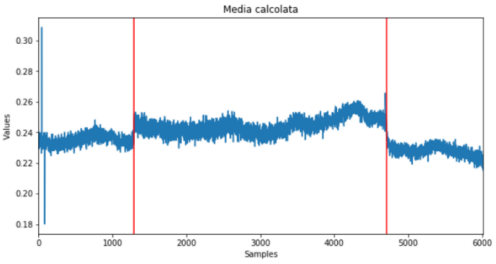
Figura 4.3: Segnale corrente di un ciclo produttivo del robot

I dataset *Gray* e *White* presentano, oltre ai valori di corrente di ciascun ciclo, alcune features. Si è deciso di calcolare nuovamente i valori di alcune tra queste per effettuarne un confronto. Di particolare interesse è risultata la media poiché ha reso possibile l'individuazione di alcuni valori anomali. Si sono osservate anche altre caratteristiche quali ad esempio minimo, massimo, deviazione standard, curtosi, asimmetria e mediana, ma non sono emerse informazioni aggiuntive rispetto a quelle già identificate attraverso l'analisi della media. Per questo motivo, di seguito, si presentano le osservazioni messe in luce dallo studio di questa caratteristica nei due dataset.

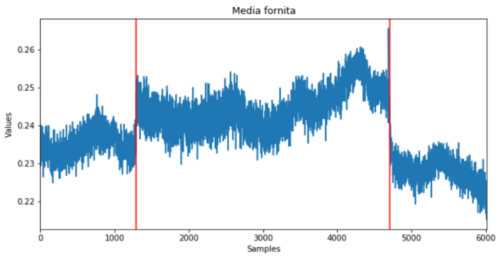
Gray Per ogni ciclo si è calcolata la media e la si è plottata (Figura 4.4 (a)). La Figura 4.4 (b), invece, rappresenta i valori medi di corrente per ogni ciclo forniti all'interno del dataset. Si possono notare subito alcune differenze:

- Il range di valori assunti dalla media calcolata è [0.18, 0.31], mentre quello assunto dalla media fornita dal dataset è [0.21, 0.27].

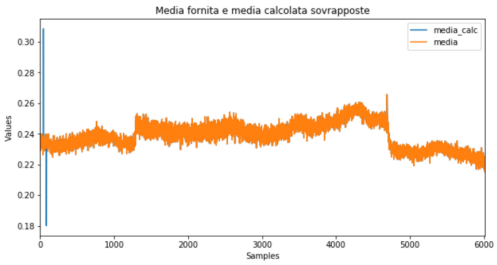
- La Figura 4.4 (c) mostra la sovrapposizione dell'andamento della media calcolata e di quella fornita e si deduce chiaramente la presenza di due valori anomali nei primi cicli. In particolare, si tratta della riga 43 (a cui corrisponde il timestamp 24-02-2020 14.28.07) e della riga 84 (24-02-2020 15.00.49) i cui valori calcolati non coincidono con quelli forniti. Per questo motivo, si è deciso di eliminare le due righe, considerate errori, per le successive analisi.



(a) *Media Gray calcolata*



(b) *Media Gray fornita*



(c) *Media calcolata e fornita sovrapposte*

Figura 4.4: Osservazione della media in Gray

Inoltre, è interessante notare come sia chiaro il passaggio da una label all'altra (indicato in Figura 4.4 (a) e (b) con le linee rosse verticali): dal ciclo 0 al 1286 sono racchiusi i record con label 0, dal 1287 al 4705 quelli con label 10 e i successivi hanno label 15. Al termine di ciascuna classe vi è un vero e proprio salto. Analizzando più nel dettaglio la media, ci si accorge che al confine tra label 10 e label 15, c'è un valore anomalo: un picco. Esso coincide con il ciclo numero 4690 (03-03-2020 16:36:15).

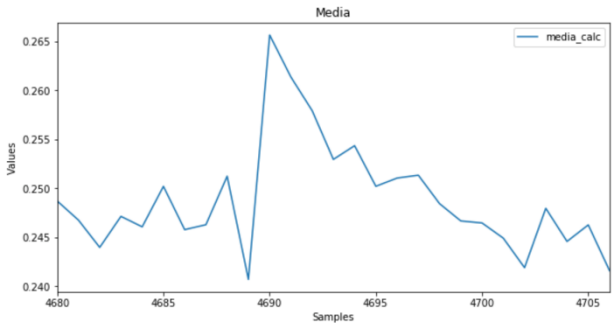


Figura 4.5: Valore anomalo Gray (ciclo: 4690)

La causa di questo picco potrebbe essere ricondotta al fatto che il monitoraggio dei cicli di lavorazione si è interrotto con il record 4689 (timestamp: 27-02-2020 08:27:33) per poi riprendere, dopo qualche giorno, con la misurazione numero 4690 (timestamp 03-03-2020 16:36:15) come si vede in Figura 4.6.

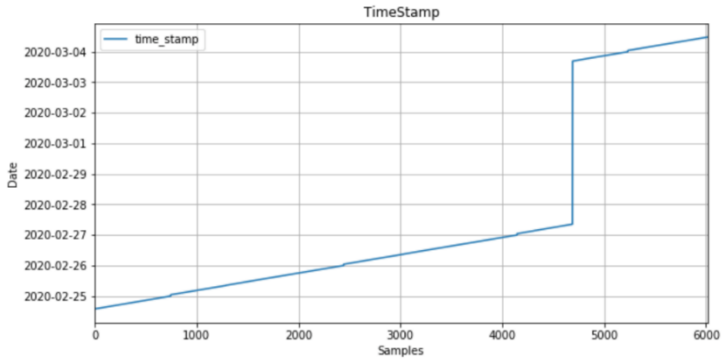


Figura 4.6: Timestamp Gray

In ultima analisi, si riporta il risultato della regressione lineare in Figura 4.7 applicata alla media calcolata. Si può notare che l'inclinazione della retta di regressione è leggermente negativa. In corrispondenza dei record 43 e 84 è possibile vedere i due cicli considerati come errori che sono stati poi rimossi per le successive analisi.

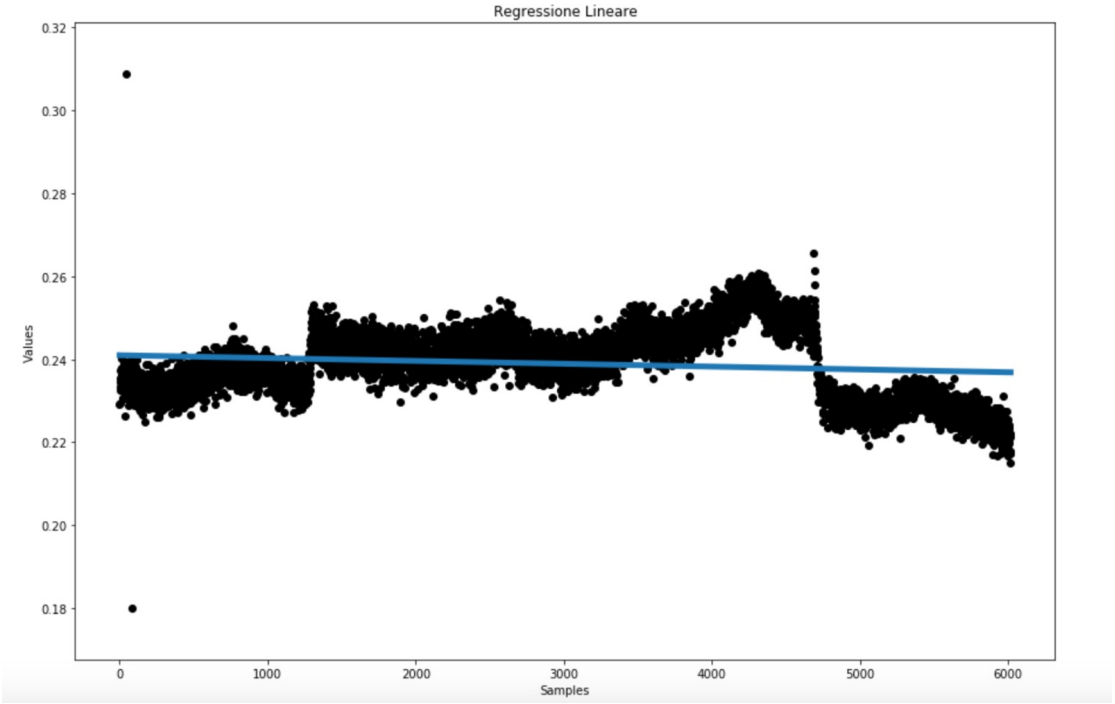


Figura 4.7: Regressione sulla media di Gray

White I valori della media calcolata e di quella fornita sul dataset *White* coincidono, quindi si riporta solamente il grafico relativo alla sovrapposizione delle due features in Figura 4.8. Le linee rosse verticali rappresentano il cambiamento di etichetta: dal ciclo 0 al 1215 ci sono i record etichettati con 0, dal 1216 al 4490 quelli con label 10 ed i successivi hanno label 15.

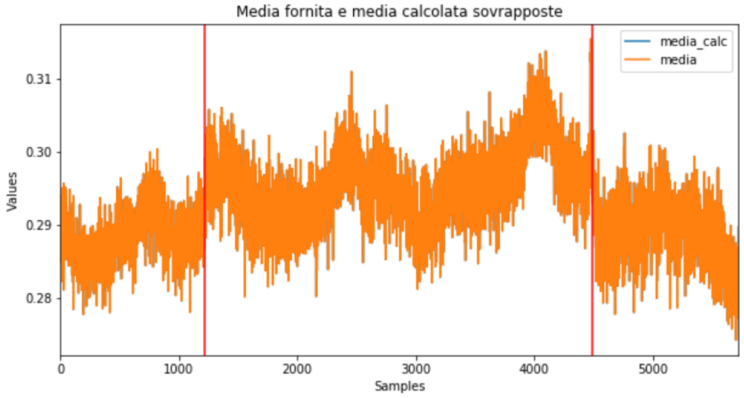


Figura 4.8: Media White

Anche in questo dataset si osserva un picco anomalo alla fine dell'etichetta 10, in particolare esso si trova in corrispondenza del ciclo numero 4476 (03-03-2020 17.33.56). Come si era già evidenziato per il dataset *Gray*, si assiste nuovamente ad un'interruzione del monitoraggio dei cicli: la misurazione 4475 è effettuata il 27-02-2020, mentre la successiva, cioè quella che riguarda il picco, risale a qualche giorno dopo. Si deduce, quindi, che il valore anomalo sia dovuto a questo fattore ed esso può essere osservato più nel dettaglio in Figura 4.10.

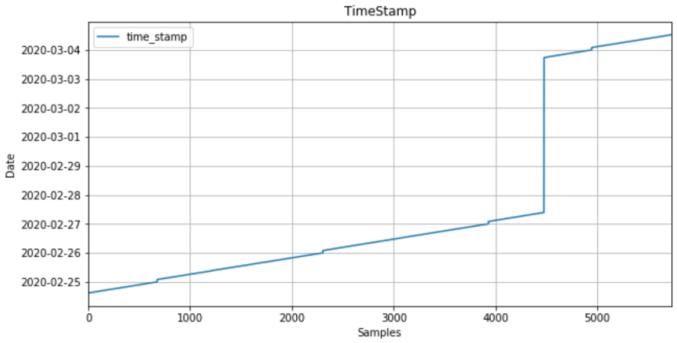


Figura 4.9: Timestamp White

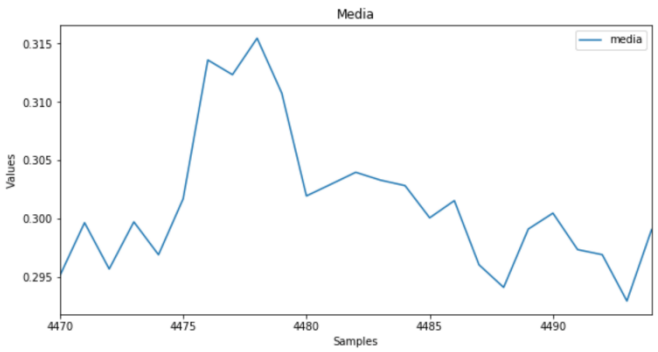


Figura 4.10: Valore anomalo White (ciclo: 4476)

In ultima analisi, si è applicata la regressione lineare sulla media e si nota in Figura 4.11 che l'inclinazione della retta di regressione in questo caso è leggermente positiva, al contrario di quanto emerso nel dataset *Gray*.

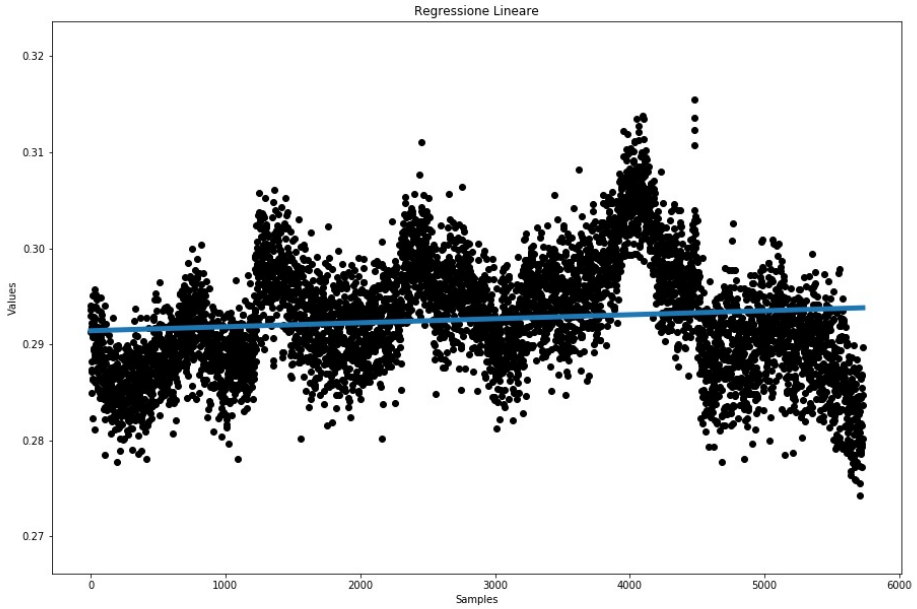


Figura 4.11: Regressione sulla media di White

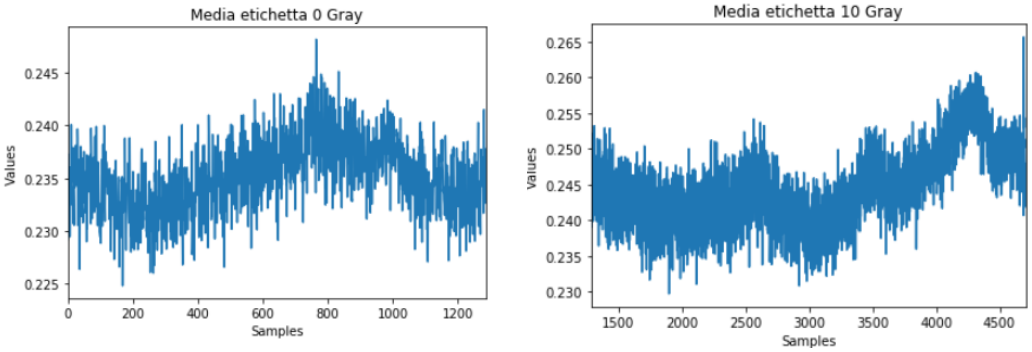
Analisi separata per etichette

Successivamente all'analisi preliminare sull'andamento del segnale, si passa ad uno studio più approfondito dei dati separatamente per etichetta per confrontare i valori appartenenti a classi diverse. Anche in questo caso si è osservato l'andamento della media.

Gray I risultati ottenuti in questa fase si riferiscono ai dai privi dei due cicli considerati errori e trovati allo step precedente.

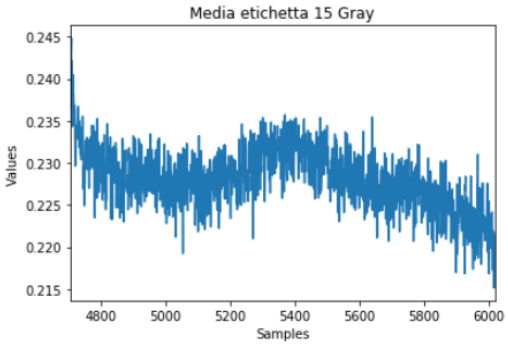
La media assume valori differenti nelle tre classi:

- Nella classe 0 varia tra 0.225 A e 0.245 A (valore medio di corrente per la classe: 0.2352)
- Nella classe 10 varia tra 0.23 A e 0.26 A (valore medio di corrente per la classe: 0.2445)
- Nella classe 15 varia tra 0.245 A e 0.215 A (valore medio di corrente per la classe: 0.2281)



(a) *Etichetta 0*

(b) *Etichetta 10*



(c) *Etichetta 15*

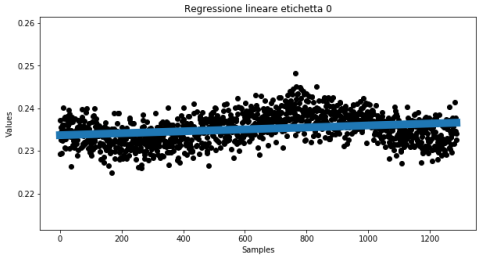
Figura 4.12: Media Gray per etichette

Dai grafici in Figura 4.12 si può notare che i record con etichetta 0 e 10, hanno valori di media che oscillano in modo piuttosto regolare. Questo fenomeno diventa meno visibile nella classe 15, per la quale si assiste ad un leggero andamento decrescente.

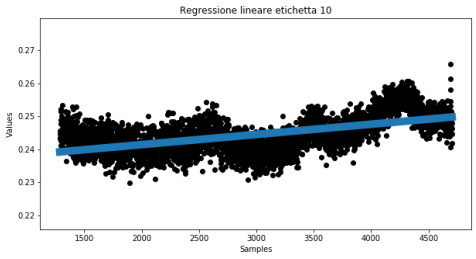
In ogni caso, come si poteva già notare dalla Figura 4.4, i cicli a cui è associato un maggior consumo di corrente sono quelli con etichetta 10, mentre i cicli a cui è associato il consumo di corrente più basso appartengono alla classe 15. La classe 0 sembrerebbe contenere i cicli intermedi, il cui consumo di corrente medio è una via di mezzo tra quello della classe 10 e 15.

Infine, anche in questo caso si è applicata la regressione lineare sulla media e si nota che le classi 0 e 10 riportano un andamento leggermente crescente, in particolar modo l'etichetta 10 è caratterizzata da una maggior pendenza positiva della retta di regressione rispetto alla classe 0. Al contrario, la retta di regressione ha pendenza negativa nella classe 15.

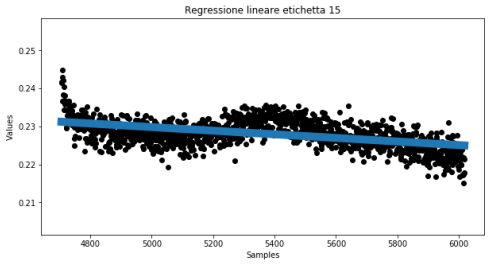
White Esattamente come nel dataset *Gray*, anche in *White* si assiste ad una distribuzione dei record tale per cui la classe 10 corrisponde ai cicli che consumano più corrente, mentre i record etichettati con 0 e 15 si riferiscono a cicli con un



(a) *Etichetta 0*



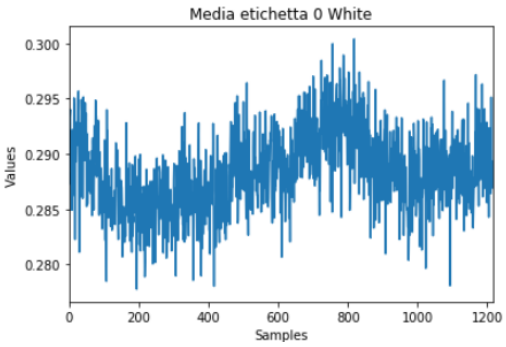
(b) *Etichetta 10*



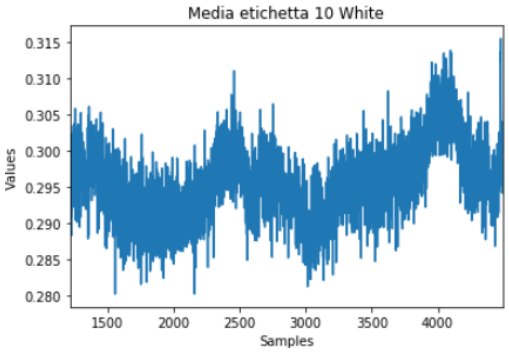
(c) *Etichetta 15*

Figura 4.13: Regressione lineare Gray per etichetta

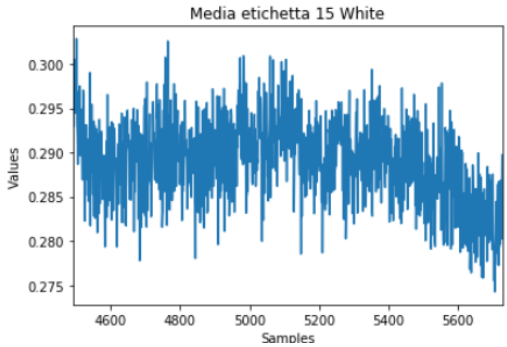
inferiore consumo energetico. Si può osservare la media in Figura 4.14 nel dettaglio.



(a) *Etichetta 0*



(b) *Etichetta 10*



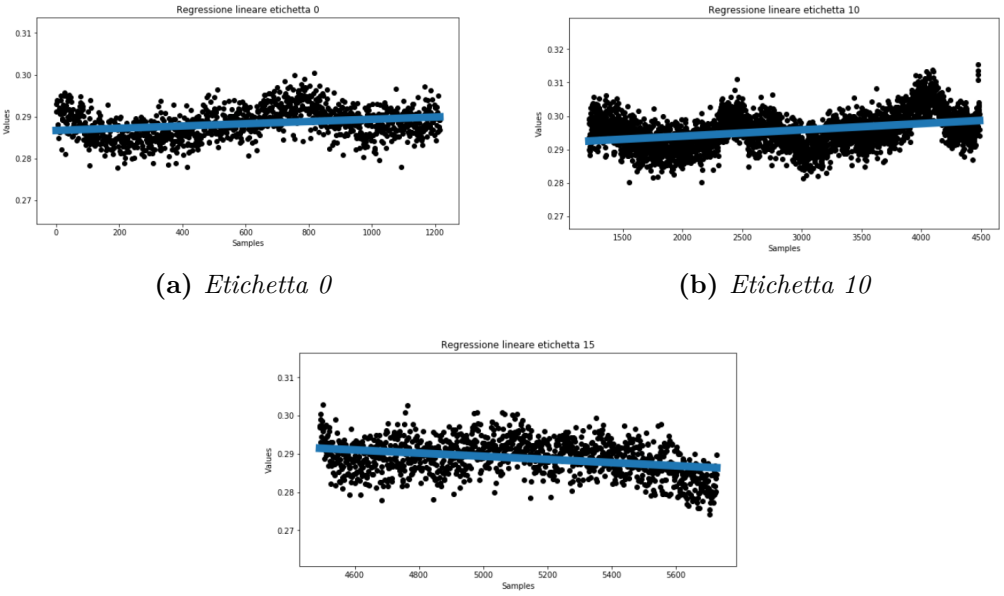
(c) *Etichetta 15*

Figura 4.14: Media White per etichette

Dall'osservazione dei grafici si deduce che la media:

- Nella classe 0 varia tra 0.28 A e 0.3 A (valore medio di corrente per la classe: 0.2884)
- Nella classe 10 varia tra 0.28 A e 0.315 A (valore medio di corrente per la classe: 0.2956)
- Nella classe 15 varia tra 0.27 A e 0.3 A (valore medio di corrente per la classe: 0.2889)

Dai grafici in Figura 4.14 si può notare che si identificano oscillazioni nella media delle classi 0 e 10 che si attenuano nella classe 15. Infine, si è applicata la regressione lineare che anche in questo caso conferma lo stesso andamento presente nelle classi del dataset *Gray*.



(a) *Etichetta 0*

(b) *Etichetta 10*

(c) *Etichetta 15*

Figura 4.15: Regressione lineare White per etichetta

4.4 Data Transformation

Questa fase è caratterizzata da due step: *smart data computation* e *feature selection*. Durante la *smart data computation* il segnale di corrente è suddiviso in split per meglio catturarne la variabilità. Il segnale, nel caso in esame, è stato suddiviso in 24 segmenti di ugual dimensione come rappresentato in Figura 4.16.

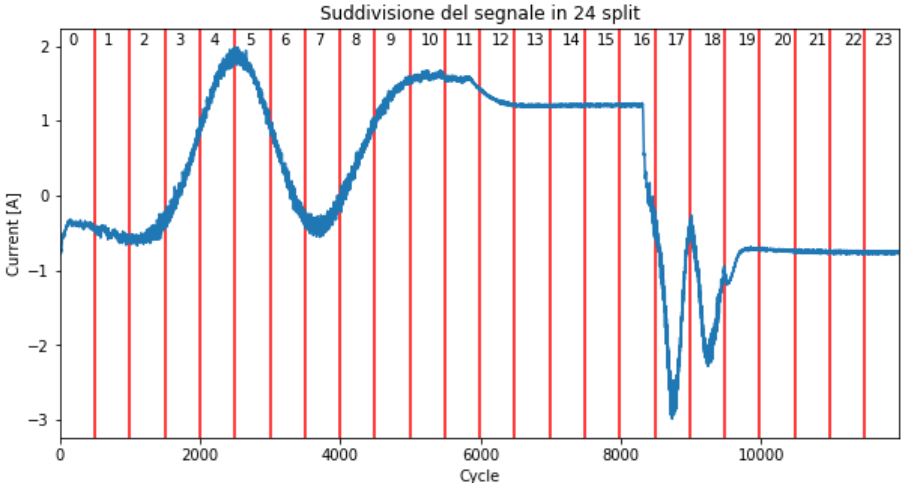


Figura 4.16: Suddivisione del segnale in 24 split

All'interno di ciascuno split sono state calcolate 14 features:

- Media, deviazione standard
- Minimo, massimo
- Primo quartile, mediana, terzo quartile
- Curtosi
- Asimmetria
- RMS (valore efficace del segnale di corrente)
- Somma in valore assoluto dei valori di corrente
- Element over mean
- Energia assoluta
- Mean absolute change (differenza tra coppie di valori consecutivi di segnale)

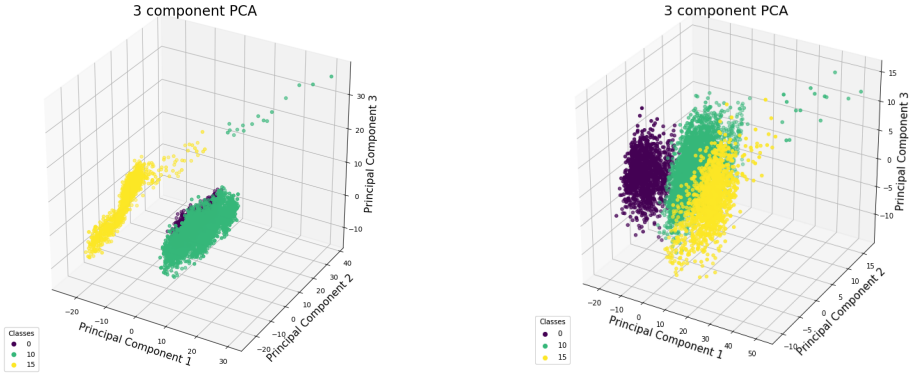
Poiché calcolare tutte le caratteristiche in ciascuno split genera un numero eccessivo di attributi e, poiché alcuni di questi sono tra loro correlati e quindi potrebbero introdurre rumore all'interno dei modelli, si passa ad una fase di *feature selection*, durante la quale si decide quali attributi rimuovere sulla base del *Mean Absolute Correlation* (MAC). Questo test prevede il calcolo della correlazione tra tutte le features e l'eliminazione di quelle che hanno un MAC superiore a 0.5. In questo modo si riducono gli attributi da prendere in considerazione e non si rischia di compromettere la precisione dei modelli. In Tabella 4.3 si mostra, per ciascun dataset, il numero di features considerate al termine della feature selection (si

ricorda che, avendo calcolato 14 features per ogni split, si parte con un totale complessivo di 336 attributi).

Dataset	# Attributi considerati	% Attributi considerati
Gray	299	89%
White	249	74,1%

Tabella 4.3: Attributi considerati dopo la feature selection

Infine, è interessante osservare come i punti si distribuiscano nelle varie classi e ciò è reso possibile dalla rappresentazione PCA (*Principal Component Analysis*). Questa tecnica è solitamente impiegata per la riduzione del numero degli attributi in fase di data reduction per semplificare i dati di origine, ma può anche essere utile per rappresentare graficamente un dataset caratterizzato da un gran numero di caratteristiche. La PCA effettua una trasformazione lineare delle variabili sulla base della varianza, cioè considera solo le features caratterizzate da una varianza elevata, riducendo così la complessità del problema. Gli attributi selezionati, chiamati *Principal components*, sono quelli che hanno maggior contenuto informativo e quindi rappresentano e semplificano l'oggetto del dataset. In questo contesto, si utilizza la PCA per rappresentare su tre dimensioni (x, y, z) i dati contenuti in *Gray* e *White* selezionando tre principal components (che sono indicate sui tre assi) e rappresentando attraverso i colori le etichette 0, 10 e 15. Nella Figura 4.17 è raffigurata la PCA relativa ai due dataset e si può notare come i cicli siano ben separati e coesi nelle proprie classi di appartenenza. Questo aspetto renderà più agevole l'applicazione degli algoritmi di Classificazione.



(a) *PCA Gray* **(b)** *PCA White*

Figura 4.17: Distribuzione dei dati nelle etichette

4.5 Analisi Predittiva

Terminata la fase di "preparazione" dei dataset si passa alla **Predictive Analysis**: l'obiettivo è quello di applicare diversi algoritmi di Classificazione sui dati storici etichettati per creare modelli predittivi in grado di prevedere l'etichetta di classe dei nuovi dati provenienti dai sensori. Per la realizzazione dei modelli di previsione si parte dai dataset etichettati *Gray* e *White* e si suddividono in set di training e di test utilizzando la Cross-Validation. Per ogni test effettuato, si calcolano le metriche di *Accuratezza*, *Precisione*, *Richiamo* ed $F - measure$ per valutare i modelli in modo da poter confrontare le prestazioni dei diversi algoritmi e selezionare quello che meglio si adatti al caso di studio. Se i modelli individuati sono affidabili, essi permettono di riconoscere tempestivamente l'arrivo di cicli anomali, altrimenti, se il modello non identifica correttamente i valori anomali non può essere considerato valido. In altre parole, l'obiettivo dei classificatori è quello di monitorare il comportamento del robot per capire se sta funzionando correttamente o meno. Gli algoritmi impiegati in questa fase sono quelli introdotti al capitolo precedente: Decision Tree, Random Forest, Classificazione Bayesiana, K-Nearest Neighbors e Support Vector Machine. Essi sono stati implementati utilizzando la libreria Scikit-learn di Python.

4.5.1 Grid Search

Ciascun algoritmo riceve come input un certo numero di iperparametri e, per valutare quelli ottimi, si è utilizzato lo strumento della Grid Search. Gli iperparametri sono caratteristiche del modello il cui valore deve essere impostato prima di eseguire l'algoritmo (si pensi, ad esempio, al valore di K nell'algoritmo KNN). La Grid Search ha come obiettivo quello di trovare i valori ottimi degli iperparametri per massimizzare l'accuratezza delle previsioni utilizzando la Brute-force search e la Cross-Validation: suddivide il dataset completo in fold attraverso la Cross-Validation e applica su tutte le combinazioni di train e di test l'algoritmo di classificazione in questione n volte, dove n rappresenta il numero di tutte le combinazioni possibili degli iperparametri. Ad esempio, impostando un numero di fold pari a 5 nella Cross-Validation e considerando un algoritmo che riceve come input due iperparametri A e B, si decide di testare alcuni possibili valori per ognuno di essi:

$$A = 10, 100$$

$$B = 0.1, 0.2, 0.5, 1.0$$

La Grid Search considera tutte e 5 le combinazioni di train e test e applica su ognuna di esse l'algoritmo di classificazione 8 volte corrispondenti alle seguenti combinazioni degli iperparametri:

- A=10, B=0.1
- A=10, B=0.2
- A=10, B=0.5
- A=10, B=1
- A=100, B=0.1
- A=100, B=0.2
- A=100, B=0.5
- A=100, B=1

Infine, per ciascun modello creato (nell'esempio sono 40), calcola il punteggio ottenuto per valutarne le prestazioni e restituisce il risultato migliore ottenibile. In Python, la Grid Search è facilmente applicabile utilizzando la libreria Scikit-learn che implementa la funzione GridSearchCV. Essa riceve come input i seguenti elementi:

- Un oggetto da stimare (in questo caso il classificatore).
- Un dictionary con gli iperparametri e i rispettivi valori da testare all'interno del classificatore in esame.
- Un criterio, chiamato *scoring* con cui valutare il modello. Nelle analisi in questione si è utilizzato un insieme di metriche di valutazione: *Accuratezza*, *Precisione*, *Richiamo* ed $F-measure$. Occorre ricordare, però, che l'*Accuratezza* è un indicatore globale del modello, mentre *Precisione*, *Richiamo* ed $F-measure$ sono relativi alle singole classi. Per ciascuna di queste tre metriche si è deciso di prendere in considerazione la media delle tre classi in modo tale da avere, per ciascun modello, quattro indicatori globali.
- Il numero di fold in cui suddividere il dataset con la Cross-Validation. La GridSearchCV implementa la Stratified K-Fold Cross-Validation e nelle analisi effettuate si è fissato un K pari a 10.

Di seguito si presenta una panoramica dei parametri testati per ciascun algoritmo.

Decision Tree

- **Criterion = {“gini”, “entropy”}**: rappresenta la metrica utilizzata per misurare la qualità della divisione del nodo. Può essere selezionato l'indice di impurità di Gini oppure l'Entropia.
- **Splitter = {“best”, “random”}**: rappresenta la strategia utilizzata per separare i nodi. In particolare, "best" sceglie la suddivisione migliore e “random” sceglie la suddivisione casuale migliore.
- **Max_depth**: rappresenta la massima profondità dell'albero e quindi si testano una serie di valori numerici interi. Una profondità troppo elevata comporta il fenomeno dell'Overfitting, mentre una profondità scarsa genera Underfitting. La profondità dipende anche dalla complessità del problema e quindi nei vari dataset sono testati valori differenti in considerazione di questo aspetto.

Random Forest

- **N_estimators**: rappresenta il numero di alberi nella foresta. Più aumenta il numero di alberi, maggiore sarà la complessità computazionale, ma le prestazioni migliorano.
- **Criterion = {"gini", "entropy"}**: è un parametro specifico del singolo albero e, come già accennato per quanto riguarda il Decision Tree, è una misura della qualità della divisione del nodo.
- **Max_depth**: come per il Decision Tree, rappresenta la massima profondità dell'albero.

Classificazione Bayesiana La classificazione Bayesiana non necessita della Grid Search in quanto non riceve come input alcun parametro. In questo caso si è semplicemente applicata a parte la Cross-Validation utilizzando la funzione cross_validate della libreria Scikit-learn ed impostando un K pari a 10.

K-Nearest Neighbors

- **N_neighbors**: è il numero dei vicini da considerare per il calcolo delle distanze.
- **Weights = {“uniform”,”distance”}** : è la funzione di peso utilizzata per la previsione. Se si utilizza “uniform” tutti i punti sono ponderati allo stesso modo, altrimenti se si sceglie “distance” si darà un peso a ciascun punto che è

tanto maggiore tanto più il punto è vicino e tanto minore tanto più il punto è lontano.

Support Vector Machine

- **Kernel={"linear","poly","rbf"}**: specifica il tipo di kernel da utilizzare nell'algoritmo.

4.5.2 Risultati Classificazione

Nelle seguenti tabelle si riportano i valori di *Accuratezza*, *Precisione*, *Richiamo* e $F-measure$ per ciascun algoritmo testato separatamente per dataset. I valori delle metriche fanno riferimento alle soluzioni migliori trovate mediante la Grid Search e quindi impostando le combinazioni ottime dei parametri di input e validando il modello con la Stratified K-Fold Cross-Validation.

Algoritmo	Accuratezza	Precisione	Richiamo	F-measure
Decision Tree	1	1	1	1
Random Forest	1	1	1	1
Class. Bayesiana	0.9966	0.9960	0.9954	0.9957
KNN	1	1	1	1
SVM	1	1	1	1

Tabella 4.4: Risultati classificazione **Gray**

Algoritmo	Accuratezza	Precisione	Richiamo	F-measure
Decision Tree	1	1	1	1
Random Forest	1	1	1	1
Class. Bayesiana	0.9938	0.9936	0.9934	0.9935
KNN	1	1	1	1
SVM	0.9979	0.9976	0.9979	0.9977

Tabella 4.5: Risultati classificazione **White**

Gli algoritmi applicati risultano tutti molto performanti sui dati presi in considerazione. Questo fenomeno è facilmente riconducibile al fatto che i dataset presentino valori ben separati in ciascuna classe e di conseguenza i classificatori riescono con facilità ad apprendere le relazioni tra variabili ed etichette e quindi si costruiscono modelli predittivi altamente affidabili. Per capire effettivamente come si distribuiscano i dati all'interno delle classi nei due dataset, si è pensato di calcolare l'indice della *Silhouette* attraverso la funzione silhouette_score della

libreria Scikit-learn. Un valore di *Silhouette* prossimo ad 1 rappresenta una situazione in cui le classi sono coese e ben separate e cioè ogni record è assegnato correttamente al proprio gruppo, al contrario il valore peggiore è -1 che rappresenta una situazione in cui i punti appartenenti ad una determinata classe sono più simili ai punti di altri gruppi e quindi i clusters non sono ben separati e coesi. Dato un punto $i \in C_i$, si definiscono:

$$a_i = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

$$b_i = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

a_i rappresenta la distanza media tra i e tutti i punti che appartengono allo stesso gruppo e quindi rappresenta la coesione, al contrario b_i è la distanza media di i da tutti i punti che appartengono alle altre classi e quindi misura il grado di separazione. La *Silhouette* è definita con la seguente formula:

$$s_i = \frac{b_i - a_i}{max(a_i, b_i)}$$

In questo caso la *Silhouette* si calcola a partire dalle classi individuate dalle labels 0, 10 e 15 note all'interno del dataset. I risultati ottenuti sono rappresentati in Tabella 4.6.

Dataset	Silhouette
Gray	0.270
White	0.246

Tabella 4.6: Silhouette

Come si può notare dai valori della *Silhouette*, i dataset assumono un punteggio piuttosto elevato e quindi le classi sono ben separate e coese, soprattutto in *Gray* in quanto presenta un valore leggermente superiore di *Silhouette* rispetto a *White*.

Dopo aver testato gli algoritmi presentati precedentemente, si è deciso di soffermarsi principalmente sul Decision Tree e sul Random Forest poiché i risultati mostrano le loro elevate performance e anche i tempi di esecuzione richiesti non sono eccessivamente onerosi.

Decision Tree

Gray Il Decision Tree risulta molto performante sul dataset *Gray*. In questo caso si è testata una max_depth pari a 2, 3 e 4 e tramite la Grid Search si è notato che è sufficiente una profondità dell'albero pari a 3 per raggiungere un'*Accuratezza*

uguale ad 1: il classificatore riesce con soli tre test ad acquisire le relazioni nascoste tra variabili ed etichette. Questo fenomeno è senza dubbio dovuto alla scarsa complessità del problema che rende piuttosto semplice la separazione dei dati nelle tre classi. Inoltre, dalla Grid Search si può notare che, indipendentemente dal parametro criterion (che può assumere indifferentemente valore "gini" o "entropy" senza compromettere l'*Accuratezza* del modello), le soluzioni ottime sono quelle che presentano lo splitter di tipo "random" e non "best". Per disegnare l'albero, si prende come riferimento la combinazione di parametri ottimi {criterion: gini, max_depth: 3, splitter: random} e si ottiene il modello in Figura 4.18, suddividendo il dataset in set di training che contiene il 75% dei valori (4512) e in set di test con il restante 25% (1505):

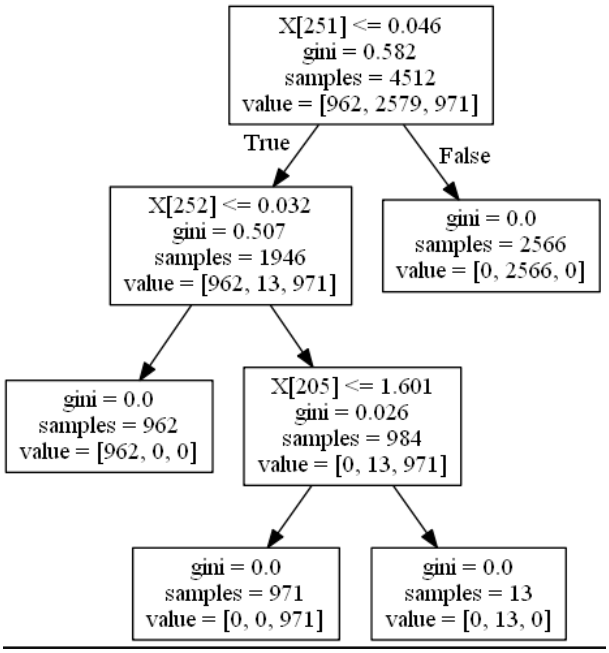


Figura 4.18: Decision Tree (Gray)

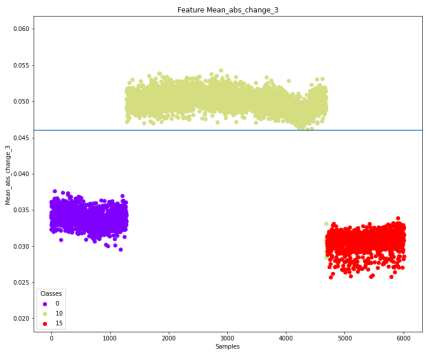
Osservando l'albero si possono dedurre quali siano gli attributi scelti per effettuare lo split sui nodi. Queste features sono quelle che permettono la suddivisione dei record nelle tre classi:

- Mean_abs_change_3 (feature numero 251): Se il valore della feature è maggiore a 0.046 allora sono identificati gli elementi appartenenti alla classe 10. Infatti, dal grafico in Figura 4.19 (a) si può notare come la classe 10 assuma valori completamente differenti da quelli assunti dai record con etichetta 0 e 15.
- Mean_abs_change_9 (feature numero 252): permette di classificare i record della classe 0, i quali presentano un valore di mean_abs_change_9 inferiore a

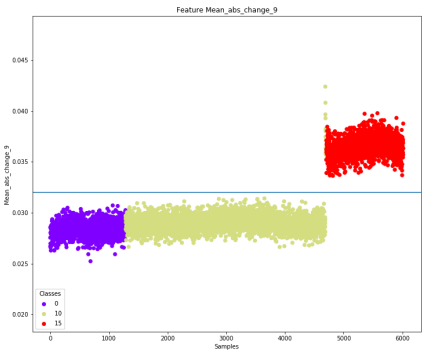
0.032.

- Median_4 (feature numero 205): separa gli ultimi elementi della classe 10 e identifica la classe 15. Se il record ha un valore di median_4 superiore a 1.601 allora sarà etichettato con 10, altrimenti con 15.

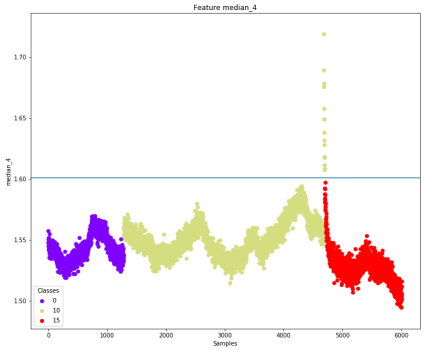
Dalla Figura 4.18 si può notare che l'albero presenta tutte foglie pure.



(a) *Mean_abs_change_3*



(b) *Mean_abs_change_9*



(c) *Median_4*

Figura 4.19: Features che dividono l'albero (Gray)

Infine, si osservano i valori di *Accuratezza*, *Precisione*, *Richiamo*, $F-measure$ e la matrice di confusione ottenuti sul test contenente il 25% dei valori dell'intero dataset. Si riconfermano le prestazioni ottime del modello:

- Matrice di confusione:
$$\begin{pmatrix} 324 & 0 & 0 \\ 0 & 878 & 0 \\ 0 & 0 & 303 \end{pmatrix}$$
- *Accuratezza*: 1

- *Precisione*: [1,1,1]
- *Richiamo*: [1,1,1]
- $F - measure$: [1,1,1]

L'*Accuratezza* sul dataset di test è 1 e significa che il modello predice correttamente tutti i record. Si è provato a variare la percentuale di dati inseriti nel training e nel test per vedere se è possibile creare un modello affidabile partendo da un ridotto numero di record conosciuti. Le predizioni sul test rimangono buone anche utilizzando solo il 10% dei dati come set di addestramento: l'*Accuratezza* sul test è pari al 99,78% e quella sul modello è pari a 1.

	10%Test 90%Train	10%Train 90%Test
Acc train	1	1
Acc test	1	0.9978

Tabella 4.7: Decision Tree Gray

White Il Decision Tree è molto performante sul dataset *White* e, con una serie di tentativi effettuati con la Grid Search, si può constatare che la profondità dell'albero richiesta per ottenere le massime prestazioni è pari a 4 e non 3 come nel dataset *Gray*. Sicuramente ciò è dovuto al fatto che i dati in *White* sono leggermente meno separati e coesi e quindi il classificatore deve effettuare più test. Anche in questo caso, le combinazioni di parametri ottime richiedono tutte lo splitter di tipo "random" e non "best". Utilizzando la prima combinazione ottima {criterion: gini, max_depth: 4, splitter: random}, un set di training contenente il 75% dei record (4296) e un set di test con il restante 25% (1433) si ottiene l'albero in Figura 4.20 come modello.

Gli attributi principali di splitting sui nodi sono:

- Mean_abs_change_5 (feature numero: 128): separa principalmente la classe 15 dalla 0 e dalla 10.
- Max_1 (feature numero: 174): Separa parte della classe 0 dalla 10.
- Mean_abs_change_8 (feature numero: 129): Separa la classe 0 della classe 10 in un test, mentre identifica definitivamente la 15 in un altro test.
- Mean_abs_change_3 (feature numero: 169): separa la classe 0 dalla 10.

Come si può notare dai test effettuati, la classe più semplice da identificare per il modello risulta essere la 15 (sono sufficienti 2 soli test), mentre è più difficile

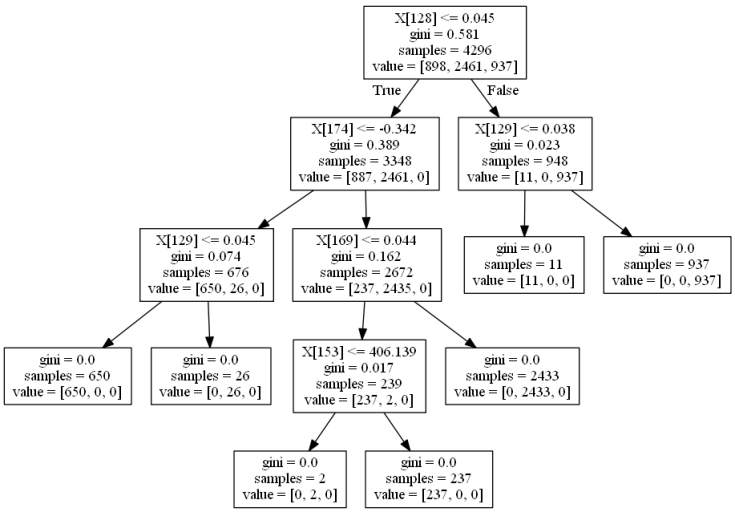
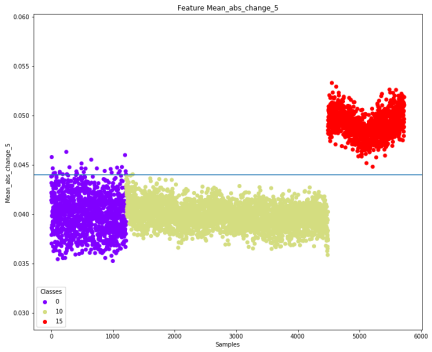
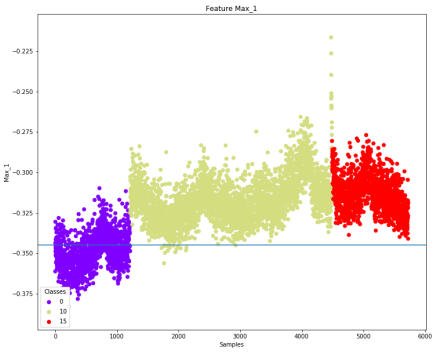


Figura 4.20: Decision Tree (White)

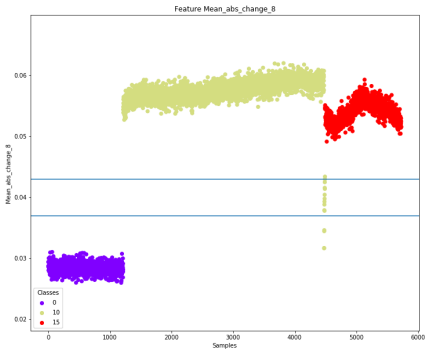
distinguere i record etichettati con 0 e 10. Anche in questo caso le foglie risultano essere pure.



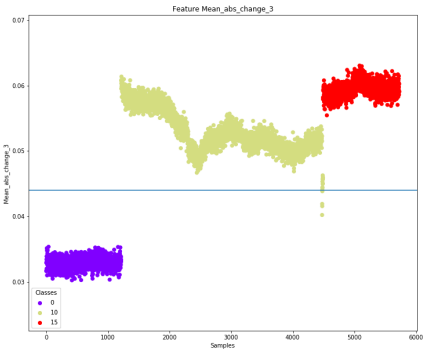
(a) *Mean_abs_change_5*



(b) *Max_1*



(c) *Mean_abs_change_8*



(d) *Mean_abs_change_3*

Figura 4.21: Features che dividono l'albero (White)

Infine, si osservano i valori di *Accuratezza*, *Precisione*, *Richiamo*, $F-measure$ e la matrice di confusione ottenuti sul test contenente il 25% dei valori dell'intero dataset. Si riconfermano le prestazioni ottime del modello:

- Matrice di confusione:
$$\begin{pmatrix} 313 & 0 & 0 \\ 0 & 862 & 0 \\ 0 & 0 & 330 \end{pmatrix}$$
- *Accuratezza*: 1
- *Precisione*: [1,1,1]
- *Richiamo*: [1,1,1]
- $F - measure$: [1,1,1]

Come per il dataset *Gray*, si sono testate diverse combinazioni di training e di test e il modello risulta affidabile anche quando è costruito sul 10% dei dati etichettati: l'*Accuratezza* sul test, infatti, peggiora in modo trascurabile assumendo un valore pari al 99,74% e l'*Accuratezza* del modello continua ad essere 1.

	10%Test 90%Train	10%Train 90%Test
Acc train	1	1
Acc test	1	0.9974

Tabella 4.8: Decision Tree White

Osservazioni Alla luce dei risultati presentati si può affermare che il Decision Tree è un algoritmo performante sui dati considerati, infatti i modelli che si creano risultano decisamente affidabili anche se creati a partire da un set limitato di dati etichettati.

Random Forest

Il Random Forest offre, in media, soluzioni migliori del Decision Tree proprio perché, combinando insieme più alberi, perfeziona le predizioni. Applicando la Grid Search sui due dataset per l'algoritmo in questione, si osserva che per *Gray* sono sufficienti 25 alberi con una profondità massima pari a 3 e per *White* 20 alberi con una profondità massima pari a 4 per ottenere un'*Accuratezza* uguale ad 1. Inoltre, anche in questo caso, si è deciso di applicare l'algoritmo con i parametri ottimi ottenuti con la Grid Search utilizzando diverse combinazioni di training e di test e

si è potuto verificare che il modello garantisce ottime prestazioni anche in presenza di pochi dati (10%) nel set di addestramento.

	10%Test 90%Train	10%Train 90%Test
Acc train	1	1
Acc test	1	0.9998

Tabella 4.9: Random Forest (Gray)

	10%Test 90%Train	10%Train 90%Test
Acc train	1	1
Acc test	1	0.9994

Tabella 4.10: Random Forest (White)

Osservazioni Si può concludere che anche il Random Forest, così come il Decision Tree, risulta essere un algoritmo valido per la creazione di modelli predittivi affidabili per i dati considerati.

4.5.3 Inserimento del rumore

I risultati finora ottenuti risultano particolarmente buoni poiché, dopo un confronto con l'esperto di dominio, si è constatato che i dati utilizzati sono stati creati ad hoc per le analisi. In particolare, è stato simulato manualmente il livello di tensionamento della cinghia riproducendo le tre situazioni possibili:

- Corretto tensionamento della cinghia
- Tensionamento eccessivo della cinghia
- Tensionamento insufficiente della cinghia

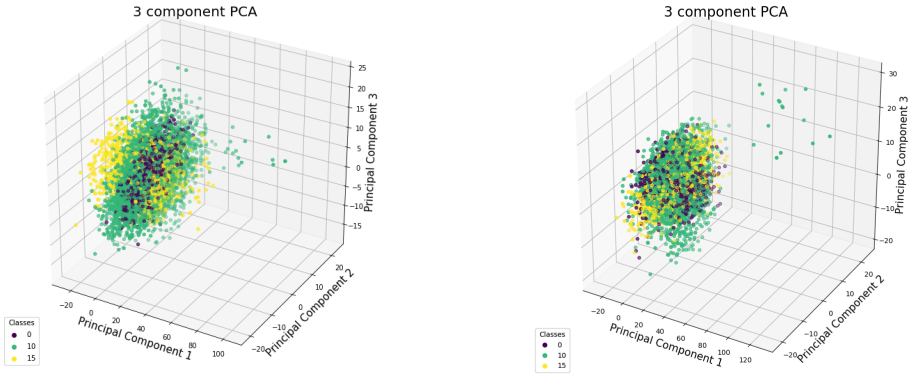
Osservando l'andamento della corrente media consumata in *Gray* e *White* (Figure 4.4 e 4.8), si ipotizza che i cicli appartenenti alla classe 0 riguardino il corretto funzionamento del robot e quindi un tensionamento adeguato delle cinghie. Al contrario, le etichette 10 e 15 corrispondo rispettivamente al sovra-tensionamento e al sotto-tensionamento della cinghia. Alla luce di queste considerazioni, i dati finora utilizzati sono difficilmente associabili ad un contesto reale e non rispecchiano il normale funzionamento del robot. Non avendo a disposizione un dataset reale, si è deciso di effettuare le medesime analisi su altri due dataset ottenuti a partire da *Gray* e *White* con l'inserimento di una certa quota di rumore. In particolare,

il rumore è stato inserito in modo tale da allineare le classi tra loro a partire da un gruppo di riferimento (in questo caso si è deciso di allineare le classi 10 e 15 alla classe 0). L'obiettivo di questa tecnica è quello di rendere le classi più vicine e quindi meno separate affinché non sia così evidente, come nei dataset puliti, il passaggio da una label all'altra. In questo modo si ottengono dati più simili a quelli che si potrebbero trovare in un contesto reale e si può quindi testare la robustezza degli algoritmi. Il rumore è stato inserito secondo il seguente criterio:

1. Si prende una classe di riferimento, ad esempio la classe 0.
2. Si considerano tutti gli n cicli che appartengono alla classe di riferimento: $x_0(t), x_1(t), ..., x_n(t)$.
3. Ad ogni istante t si calcola la media tra i valori assunti dagli n segnali di corrente all'istante t:
$$Avg_0(t) = \frac{1}{n} \sum_{i \leq n} x_i(t)$$
e $Avg_0(t)$ rappresenta la media dei valori assunti dai segnali di corrente all'istante t della classe 0 .
4. Si calcolano anche $Avg_{10}(t)$ e $Avg_{15}(t)$, cioè le medie dei valori assunti all'istante t dai segnali di corrente delle classi 10 e 15.
5. Si calcola lo scarto delle medie rispetto alla classe di riferimento:
$$S_{0-10}(t) = Avg_0(t) - Avg_{10}(t)$$
$$S_{0-15}(t) = Avg_0(t) - Avg_{15}(t)$$
6. Si considerano i cicli etichettati con 10 e 15, dove r è il numero di cicli contenuti nella classe 10, mentre p sono i cicli della classe 15:
$$x_0(t), x_1(t), ..., x_r(t)$$
$$x_0(t), x_1(t), ..., x_p(t)$$
7. Su ogni rilevamento di corrente all'istante t appartenete alla classe j, dove j rappresenta le classi non assunte come riferimento (e quindi o la 10 o la 15), si aggiunge rumore nel seguente modo:
   (a) Si applica la formula per l'allineamento: $x_i(t) = x_i(t) + S_{0-j}(t)$
   (b) Si definisce la funzione di probabilità della distribuzione uniforme per l'aggiunta di rumore:
   $$p(x) = \frac{1}{b-a}$$
   dove $b = -0.05 * x_i(t)$ e $a = 0.05 * x_i(t)$

(c) Si calcola $x_i(t) = x_i(t) + p(x)$

In Figura 4.22 si presenta la distribuzione dei record nei dataset ottenuti attraverso la rappresentazione PCA e si può notare come l'inserimento del rumore renda prevedibilmente più confusa la distinzione delle classi che appaiono meno separate e quindi molto più simili tra loro.



(a) *PCA Gray con rumore* **(b)** *PCA White con rumore*

Figura 4.22: Distribuzione dei dati con rumore nelle etichette

Se, infatti, si calcola la *Silhouette*, i dati con rumore presentano valori negativi e molto vicini allo 0 e ciò significa che le classi sono sovrapposte. In questi casi, il classificatore avrà necessità di maggiori risorse di tempo e computazionali per la creazione del modello in fase di training.

Dataset	Silhouette
Gray con rumore	-0.065
White con rumore	-0.067

Tabella 4.11: Silhouette dati con rumore

Infine, applicando la feature selection ai dati con rumore, ci si accorge che non è scartato nessun attributo in quanto il coefficiente di correlazione assume valori sempre inferiori a 0.5. Per questo motivo, i due dataset sono caratterizzati da 336 attributi che saranno considerati nelle analisi.

Di seguito si presentano i risultati ottenuti applicando gli algoritmi con l'utilizzo della Grid Search ai dati con rumore e, come si può notare, i valori di *Accuratezza*, *Precisione*, *Richiamo* e $F-measure$ sono in generale diminuiti rispetto al caso senza rumore. In particolare, i risultati ottenuti con la Classificazione Bayesiana

sono peggiorati in modo considerevole e l'algoritmo SVM non è stato applicabile in quanto i tempi di richiesti in fase di training sono risultati troppo elevati. Al contrario, si può osservare come il Decision Tree, il Random Forest e il KNN continuino ad essere performanti. Anche in questo caso ci si sofferma sul Decision Tree e sul Random Forest.

Algoritmo	Accuratezza	Precisione	Richiamo	F-measure
Decision Tree	0.9744	0.9694	0.9656	0.9674
Random Forest	0.9970	0.9967	0.9954	0.9960
Class. Bayesiana	0.7107	0.7188	0.7212	0.6932
KNN	0.9929	0.9899	0.9934	0.9916

Tabella 4.12: Risultati classificazione **Gray con rumore**

Algoritmo	Accuratezza	Precisione	Richiamo	F-measure
Decision Tree	0.8560	0.8330	0.8314	0.8318
Random Forest	0.9721	0.9772	0.9588	0.9674
Class. Bayesiana	0.5554	0.5866	0.6085	0.5530
KNN	0.9438	0.9504	0.9244	0.9356

Tabella 4.13: Risultati classificazione **White con rumore**

Decision Tree

Gray con rumore Considerando il dataset *Gray* con rumore, ci si rende conto che le prestazioni del modello sono prevedibilmente diminuite. La complessità del problema è aumenta e dunque non è più sufficiente una profondità ridotta dell'albero per la realizzazione di un modello predittivo soddisfacente. Infatti, in questo caso la profondità ottima trovata con la Grid Search risulta essere pari ad 8 e, indipendentemente dai valori assunti dagli altri parametri, una profondità superiore o inferiore ad 8 comporta un'*Accuratezza* sul test inferiore. Gli altri parametri ottimali risultano essere {criterion: gini, splitter: best}. Applicando questa combinazione al classificatore e utilizzando un training set con il 75% dei record e un test set con il restante 25% si ottiene il seguente modello in Figura 4.23.

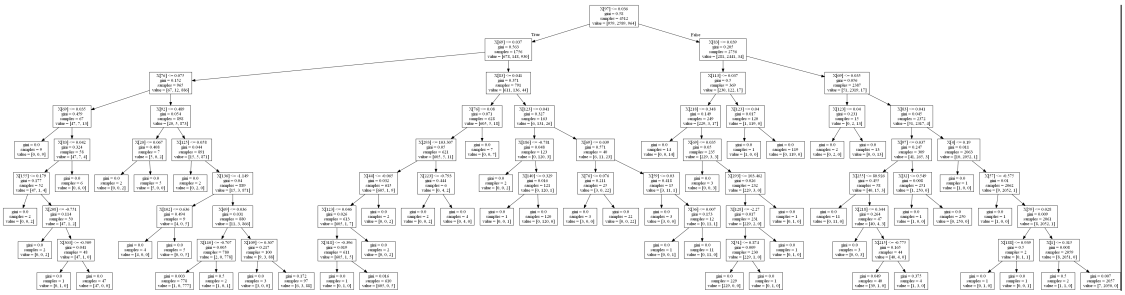
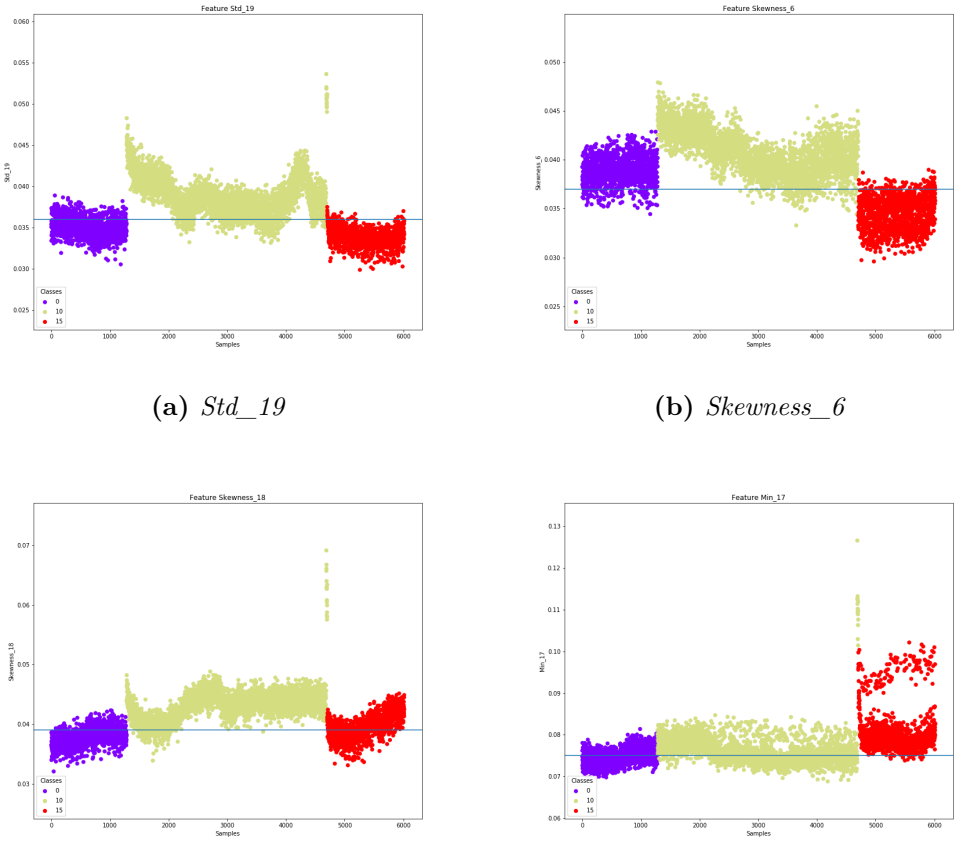


Figura 4.23: Decision Tree (Gray con rumore)

Prendendo ad esempio i primi attributi di splitting sui nodi del modello, si può notare come effettivamente sia più complesso trovare delle soglie che separino bene i dati poiché i valori delle features nelle tre classi sono molto più simili tra loro rispetto al caso senza rumore (Figura 4.24).



(a) *Std_19*

(b) *Skewness_6*

(c) *Skewness_18*

(d) *Min_17*

Figura 4.24: Features che dividono l'albero (Gray con rumore)

Di seguito si osservano i valori di *Accuratezza*, *Precisione*, *Richiamo*, $F-measure$ e la matrice di confusione ottenuti sul test contenente il 25% dei valori

dell'intero dataset. Le prestazioni del modello continuano ad essere buone benché siano comunque diminuite rispetto al caso benchmark in quanto alcuni record vengono assegnati alla classe non corretta:

- Matrice di confusione:
$$\begin{pmatrix} 285 & 11 & 11 \\ 2 & 872 & 3 \\ 8 & 4 & 309 \end{pmatrix}$$
- *Accuratezza*: 0.9740
- *Precisione*: [0.9661,0.9831,0.9567]
- *Richiamo*: [0.9283,0.9943,0.9626]
- $F - measure$: [0.9468,0.9886,0.9596]

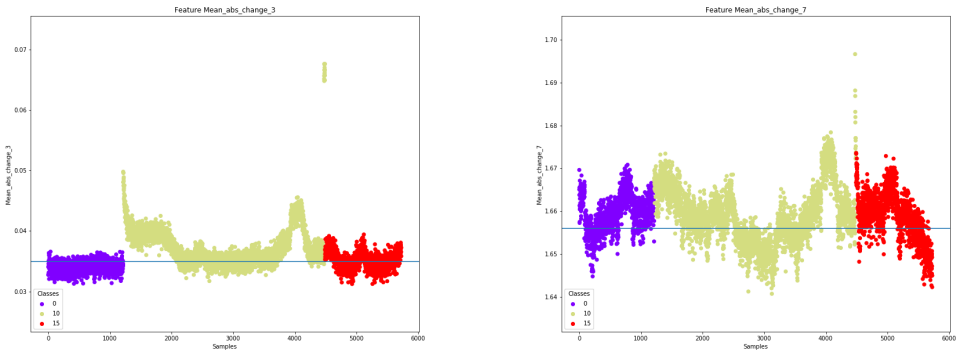
Anche in questo caso, è possibile ottenere un modello sufficientemente affidabile utilizzando una porzione limitata di dati etichettati come training: ad esempio, con il 10% dei dati nel set di allenamento, l'accuratezza sul test è del 93.26% mentre quella sul modello è pari al 99.66%. Quindi, nonostante l'inserimento del rumore, il modello non è affetto da Overfitting, in quanto le prestazioni sul test rimangono elevate anche a fronte di un modello realizzato a partire da pochi record etichettati.

	10%Test 90%Train	10%Train 90%Test
Acc train	0.9957	0.9966
Acc test	0.9817	0.9326

Tabella 4.14: Decision Tree Gray con rumore

White con rumore Le medesime analisi sono state effettuate sul dataset *White* con rumore, per il quale la massima profondità dell'albero ottimale ricavata con la Grid Search risulta essere pari a 11 combinata con i parametri {criterion: gini, splitter: best}. Per disegnare il modello si applica l'algoritmo con i parametri ottimali utilizzando un training set e un test set contenenti rispettivamente il 75% ed il 25% dei record. L'albero ottenuto come modello è molto profondo ed ampio e questo si traduce in previsioni meno accurate sul test.

Figura 4.25: Decision Tree (White con rumore)



(a) *Mean_abs_change_3* **(b)** *Mean_abs_change_7*

Figura 4.26: Features che dividono l'albero (White con rumore)

Anche in questo caso, così come per il dataset *Gray* con rumore, gli attributi di splitting dei nodi dell'albero presentano valori per ciascuna classe più simili tra loro e quindi è più complesso riuscire a trovare soglie che suddividano con facilità i record nelle classi di appartenenza (Figura 4.26).

A causa delle dimensioni dell'albero, l'*Accuratezza* sul test è peggiorata rispetto al caso senza rumore, così come le altre metriche:

- Matrice di confusione:

$$\begin{pmatrix} 260 & 34 & 18 \\ 45 & 723 & 58 \\ 17 & 55 & 223 \end{pmatrix}$$

- *Accuratezza*: 0.8415
- *Precisione*: [0.8074,0.8903,0.7458]
- *Richiamo*: [0.8333,0.8753,0.7559]
- $F - measure$: [0.8201,0.8827,0.7508]

In questo caso, riducendo le dimensioni del training si assiste al fenomeno dell'Overfitting, in quanto l'accuratezza sul dataset di allenamento aumenta, mentre sul test diminuisce. Ciò è dovuto al fatto che, riducendo la cardinalità del set di training, si genera un modello molto più specifico e meno generalizzabile.

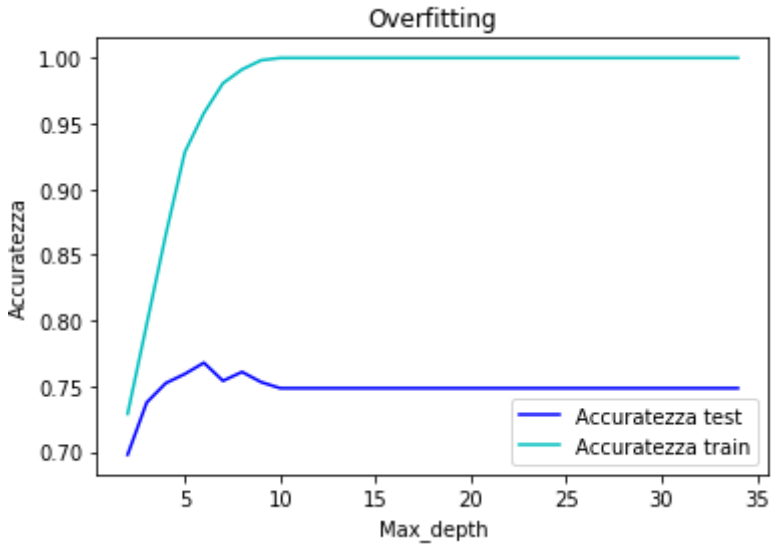


Figura 4.27: Fenomeno dell'Overfitting (White con rumore): Training con il 10% dei dati

Il fenomeno dell'Overfitting si affievolisce nel momento in cui si aumenta la dimensione del dataset di training (Figura 4.28) poiché, considerando più dati etichettati, si genera un modello meno specifico e di conseguenza più semplice da generalizzare su dati non etichettati, anche se l'*Accuratezza* sul test continua a non essere particolarmente elevata.

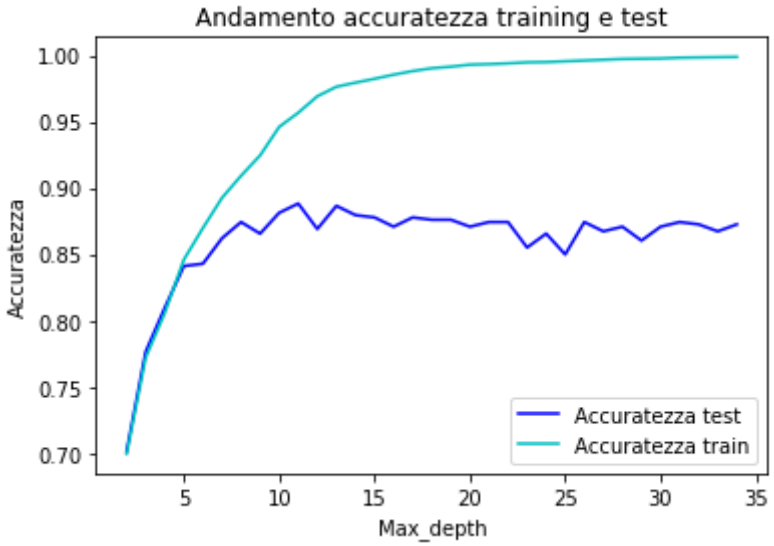


Figura 4.28: Andamento accuratezza training e test (White con rumore): Training con il 90% dei dati

	10%Test 90%Train	10%Train 90%Test
Acc train	0.9567	1
Acc test	0.8883	0.7485

Tabella 4.15: Decision Tree White con rumore

Random Forest

Nel dataset con rumore, applicando la Grid Search per trovare i parametri di input ottimali per il Random Forest, si ottiene per *Gray* un numero di alberi pari a 25 con una profondità massima di 11, mentre *White* necessita di 45 alberi con una profondità pari a 15.

Inoltre, anche in questo caso si è deciso di applicare l'algoritmo con i parametri ottenuti con la Grid Search, utilizzando diverse combinazioni di training e test e si è potuto verificare che il modello garantisce ottime prestazioni anche in presenza di pochi dati (10%) nel set di addestramento. Ciò è vero anche per il dataset *White* con rumore e quindi il Random Forest ha risolto i problemi di Overfitting del Decision Tree, migliorando l'*Accuratezza* delle predizioni in entrambi i dataset e in modo più significativo nel dataset *White*.

	10%Test 90%Train	10%Train 90%Test
Acc train	0.9998	1
Acc test	0.9950	0.9802

Tabella 4.16: Random Forest Gray con rumore

	10%Test 90%Train	10%Train 90%Test
Acc train	1	1
Acc test	0.9755	0.8747

Tabella 4.17: Random Forest White con rumore

Osservazioni Alla luce degli esperimenti effettuati, si può affermare che l'algoritmo che in generale offre prestazioni migliori è il Random Forest. Esso, infatti, permette la creazione di modelli affidabili creati anche solo sulla base di un ristretto numero di record etichettati (10%), realizzando predizioni con un'*Accuratezza* maggiore rispetto a quella fornita dal Decision Tree e dimostrandosi quindi in grado di riconoscere tempestivamente i dati in arrivo dai sensori anche nel caso dei dataset con rumore.

Il Decision Tree è un algoritmo altrettanto valido e infatti i risultati ottenuti sono piuttosto soddisfacenti, anche se nel dataset *White* con rumore le prestazioni peggiorano leggermente e, se non si ha una quantità sufficiente di dati etichettati per il set di allenamento, allora c'è il rischio che si generi il fenomeno dell'Overfitting. Esso si manifesta poiché è stato inserito rumore nel dataset e perché è aumentato il numero di attributi considerati e questi fattori aumentano la complessità del problema.

L'aumento di complessità comporta il rischio della creazione di modelli poco distorti, ma con un elevata varianza e quindi il modello ha elevate capacità di apprendimento delle relazioni tra variabili ed etichette sui dati di training, ma ha scarsa capacità di generalizzazione.

In ogni caso, osservando i risultati ottenuti applicando la Grid Search e la Cross-Validation (Tabelle 4.4, 4.5, 4.12, 4.13), si può facilmente concludere che il Random Forest offre in media prestazioni più elevate in tutti i dataset considerati, migliorando le previsioni in particolar modo sul dataset *White* con rumore rispetto a quelle ottenute con il Decision Tree. Tuttavia, anche il KNN fornisce soluzioni decisamente soddisfacenti, superando le prestazioni del Decision Tree. In ogni caso, il Random Forest si conferma la scelta migliore per i dataset impiegati. In Figura 4.29 si riassumono i valori di *Accuratezza*, *Precisione*, *Richiamo* e $F-measure$ dei tre algoritmi che si sono dimostrati in generale più performanti.

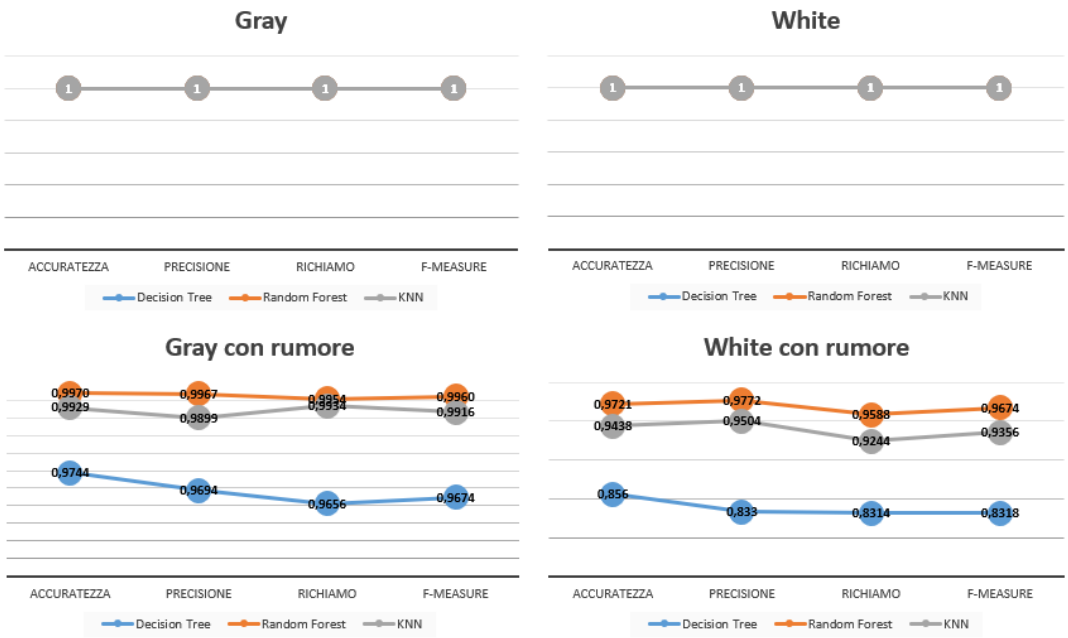


Figura 4.29: Riassunto algoritmi migliori

4.5.4 Classificazione incrociata

La classificazione incrociata consiste nell'applicazione degli algoritmi Decision Tree e Random Forest utilizzando come training e come test due differenti dataset. In particolare, si sono testate le seguenti combinazioni:

- Training su *Gray* e test su *White*
- Training su *White* e test su *Gray*
- Training su *Gray* con rumore e test su *White* con rumore
- Training su *White* con rumore e test su *Gray* con rumore

Si è pensato di sviluppare questo tipo di analisi perché i dataset *Gray* e *White* sono realizzati a partire dai valori di corrente generati dallo stesso robot. Per questo motivo dovrebbero risultare sovrapponibili e i dati dovrebbero essere caratterizzati dalla stessa distribuzione. L'obiettivo di questo esperimento è quello di valutare se si possono ottenere modelli predittivi affidabili che riconoscano i segnali di corrente anomali utilizzando come set di addestramento un dataset e considerando l'altro dataset a disposizione come set di test. L'analisi si basa sull'utilizzo dei dati prima della feature selection poiché *Gray* e *White*, in seguito alla selezione delle caratteristiche più importanti, presentano attributi differenti e quindi sarebbe stato errato confrontarli. Allo stesso tempo, però, il fatto di considerare features tra loro correlate introduce rumore che può compromette in parte le analisi effettuate. Fortunatamente, gli algoritmi selezionati sono piuttosto robusti in questo senso.

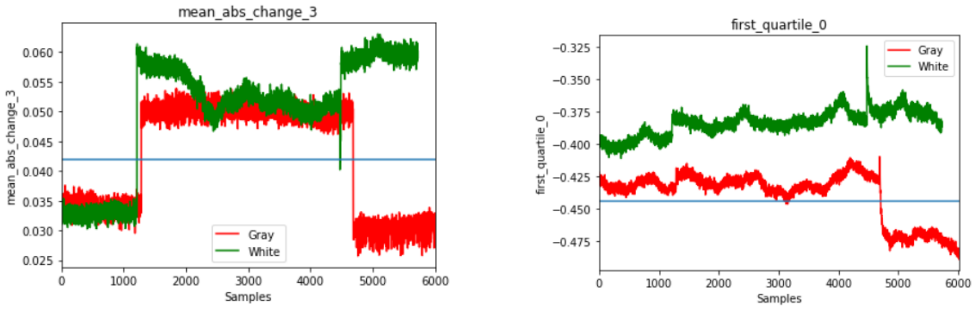
Decision Tree

Train Gray e test White Si applica la Grid Search sul dataset *Gray* per trovare la combinazione ottima di parametri da passare come input all'algoritmo. I parametri ottimali risultano essere {criterion: gini, max_depth: 2, splitter: best}. Passando alla fase di predizione sul test *White* si ottengono i seguenti risultati:

- Matrice di confusione:
$$\begin{pmatrix} 1216 & 0 & 0 \\ 2 & 3273 & 0 \\ 0 & 1238 & 0 \end{pmatrix}$$
- *Accuratezza*: 0.7836
- *Precisione*: $[0.9984, 0.7255, 0]$
- *Richiamo*: $[1, 0.9994, 0]$

- $F - measure$: $[0.9992, 0.8407, 0]$

L'*Accuratezza* sul test non è particolarmente soddisfacente poiché il modello non riesce ad identificare la classe 15 di *White* che è erroneamente etichettata con 10. La classe 15 in *White* assume valori più simili alla classe 10 di *Gray* e lo si può notare osservando l'andamento dei best splitting attributes dei nodi in Figura 4.30. L'*Accuratezza* del modello, invece, è elevata (99.73%).



(a) *Mean_abs_change_3* **(b)** *First_quartile_0*

Figura 4.30: Attributi che splittano l'albero (train Gray e test White)

Train White e test Gray Individuata la combinazione di parametri ottimi che risulta essere {criterion: gini, max_depth: 2, splitter: best}, si passa alla fase di predizione sul test e si ottengono i seguenti risultati:

- Matrice di confusione:

$$\begin{pmatrix} 1285 & 0 & 0 \\ 0 & 1146 & 2273 \\ 51 & 0 & 1262 \end{pmatrix}$$

- *Accuratezza*: 0.6137
- *Precisione*: $[0.9618, 1, 0.3570]$
- *Richiamo*: $[0.9619, 0.3352, 0.9612]$
- $F - measure$: $[0.9805, 0.5021, 0.5206]$

Come era facilmente prevedibile, in questo caso il classificatore non riesce a etichettare correttamente la classe 10 poichè in *Gray* gli elementi di questa classe sono più simili a quelli della classe 15 di *White*.

Train Gray e test White con rumore Applicando la combinazione di parametri ottima {criterion: gini, max_depth: 7, splitter: best} ottenuta mediante la Grid Search, si nota che, se si utilizzano i dati con il rumore, le prestazioni del modello peggiorano rispetto al caso senza rumore e si manifesta il fenomeno dell'Overfitting. Infatti, l'accuratezza sul modello è del 99,11%, mentre è solo del 56% sul test.

- Matrice di confusione:
$$\begin{pmatrix} 434 & 108 & 674 \\ 303 & 1910 & 1062 \\ 17 & 352 & 869 \end{pmatrix}$$
- *Accuratezza*: 0.5608
- *Precisione*: $[0.5756, 0.8059, 0.3335]$
- *Richiamo*: $[0.3569, 0.5832, 0.7019]$
- $F - measure$: $[0.4406, 0.6767, 0.4522]$

L'Overfitting è rappresentato in Figura 4.31 e si genera perchè il modello creato è troppo specifico per i dati contenuti in *Gray* e non è rappresentativo dei dati di *White*.

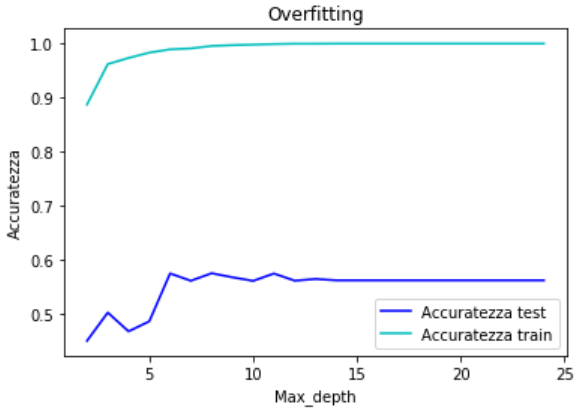


Figura 4.31: Overfitting train Gray test White con rumore

Train White e test Gray con rumore Applicando la combinazione di parametri ottima individuata dalla Grid Search {criterion: gini, max_depth: 11, splitter: best} le performance sul test caratterizzato dai record di *Gray* sono piuttosto scandenti, mentre l'accuratezza sul modello è elevata (95.95%). Si conferma quindi la presenza di Overfitting (Figura 4.32).

- Matrice di confusione:
$$\begin{pmatrix} 65 & 494 & 726 \\ 19 & 2255 & 1145 \\ 15 & 1053 & 245 \end{pmatrix}$$
- *Accuratezza*: 0.4437
- *Precisione*: $[0.37855, 0.5903, 0.0913]$
- *Richiamo*: $[0.0412, 0.7198, 0.1188]$
- $F - measure$: $[0.0743, 0.6486, 0.1032]$

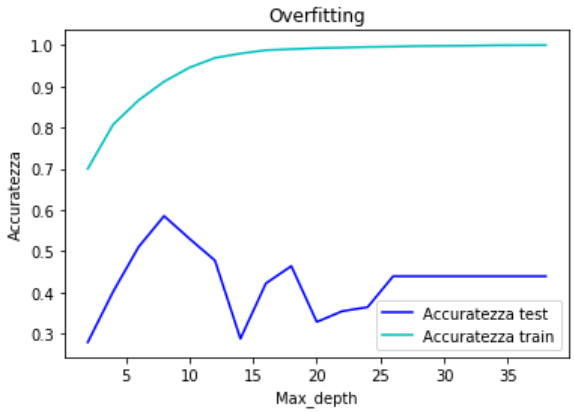


Figura 4.32: Overfitting train White test Gray con rumore

Osservazioni Dalle analisi effettuate, si deduce che i dataset *Gray* e *White* presentano delle differenze nella distribuzione dei dati e quindi realizzare modelli predittivi con il Decision Tree a partire da un solo dataset non è corretto perché i dati di un dataset non sono abbastanza rappresentativi dell'altro. Questo fattore porta alla creazione di modelli non adeguati e ciò accade anche nei dati privi di rumore, per i quali il classificatore confonde la classe 15 di *White* con la classe 10 di *Gray*. Infatti, nei due esperimenti effettuati con i dataset puliti, l'*Accuratezza* risulta non soddisfacente a causa del mancato riconoscimento di una sola classe. Con l'introduzione del rumore la situazione peggiora ed è difficile per i modelli identificare tutte le classi e quindi si genera un evidente fenomeno di Overfitting dovuto al fatto che i dati di training non sono rappresentativi dei dati nel test.

Random Forest

Come nelle analisi effettuate con il Decision Tree, si applica l'algoritmo Random Forest sul test dopo aver identificato mediante la Grid Search i parametri ottimali. Dopodiché si valuta l'*Accuratezza* sul modello e sul test.

Train Gray e test White I risultati ottenuti applicando la combinazione ottima di parametri {criterion: gini, n_estimators: 10, max_depth: 3} sono:

- Matrice di confusione:
$$\begin{pmatrix} 740 & 476 & 0 \\ 0 & 3275 & 0 \\ 0 & 1238 & 0 \end{pmatrix}$$
- *Accuratezza*: 0.7008
- *Precisione*: $[1, 0.6564, 0]$
- *Richiamo*: $[0.6085, 1, 0]$
- $F - measure$: $[0.7566, 0.77926, 0]$

L'*Accuratezza* sul modello è pari a 1, mentre sul test non è particolarmente elevata. Infatti, a differenza del Decision Tree che riusciva a predire correttamente la classe 0 e 10, il Random Forest predice correttamente solo la classe 10 e parte della 0.

Train White e test Gray I risultati ottenuti applicando la combinazione ottima {n_estimators: 100, criterion: gini , max_depth: 6 } sul test sono decisamente scadenti perché la maggior parte dei record sono attribuiti alla classe 0:

- Matrice di confusione:
$$\begin{pmatrix} 1259 & 26 & 0 \\ 2578 & 783 & 58 \\ 1313 & 0 & 0 \end{pmatrix}$$
- *Accuratezza*: 0.3393
- *Precisione*: $[0.2444, 0.9678, 0]$
- *Richiamo*: $[0.9797, 0.2290, 0]$
- $F - measure$: $[0.3912, 0.3703, 0]$

L'*Accuratezza* sul modello è molto elevata (pari ad 1) ed esso non si adatta affatto al test.

Train Gray e test White con rumore I risultati ottenuti sul test applicando la combinazione ottima {criterion: gini, max_depth: 20, n_estimators: 40} non sono buoni poichè la maggior parte dei record è attribuita alla classe 10. L'*Accuratezza* sul modello è pari ad 1, mentre sul test si ottengono i seguenti risultati:

- Matrice di confusione:

$$\begin{pmatrix} 3 & 1194 & 19 \\ 3 & 3236 & 36 \\ 0 & 1222 & 16 \end{pmatrix}$$

- *Accuratezza*:0.5682

- *Precisione*: $[0.5000, 0.5725, 0.2253]$

- *Richiamo*: $[0.0024, 0.9880, 0.0129]$

- $F - measure$: $[0.0049, 0.7249, 0.0244]$

Train White e test Gray con rumore In questo caso, inserendo come input i parametri {criterion: entropy, max_depth: 15, n_estimators: 45}, i record vengono assegnati principalmente alla classe 0 e alla 10. L'*Accuratezza* sul modello è pari ad 1, mentre i risultati sul test sono:

- Matrice di confusione:

$$\begin{pmatrix} 513 & 772 & 0 \\ 186 & 3226 & 7 \\ 840 & 472 & 0 \end{pmatrix}$$

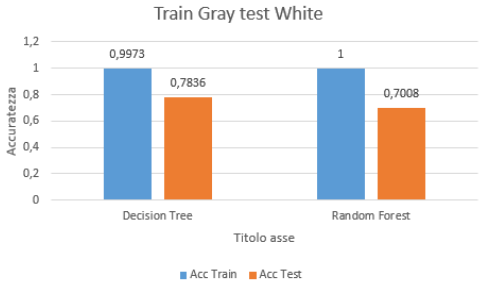
- *Accuratezza*: 0.62140

- *Precisione*: $[0.3333, 0.7215, 0]$

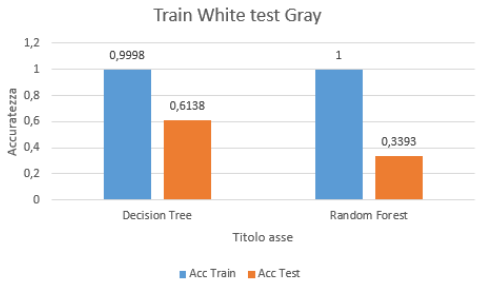
- *Richiamo*: $[0.3992, 0.9435, 0]$

- $F - measure$: $[0.3633, 0.8177, 0]$

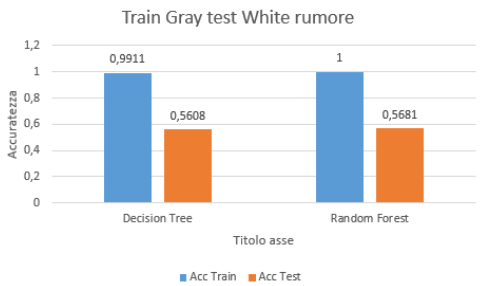
Conclusioni Nei grafici in Figura 4.33 sono riassunti i risultati ottenuti in questa analisi:



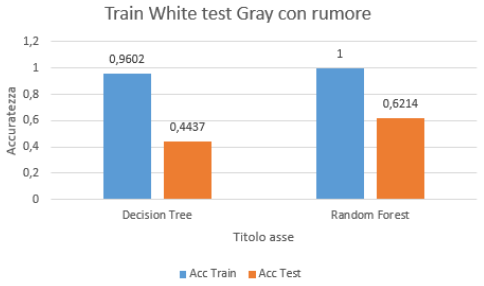
(a) *Train Gray test White senza rumore*



(b) *Train White test Gray senza rumore*



(c) *Train Gray test White con rumore*



(d) *Train White test Gray con rumore*

Figura 4.33: Accuratezza training e test

Concludendo, il Decision Tree offre modelli che garantiscono una previsione più corretta sul test nel caso in cui si considerino i dati puliti, al contrario, sui dati con il rumore risulta più adeguato utilizzare un modello creato con il Random Forest. Questo fenomeno è riconducibile al fatto che il Decision Tree sui dati rumorosi è affetto da Overfitting, mentre il Random Forest rende meno evidente questo problema. In ogni caso i risultati non sono buoni poiché i due dataset contengono differenze nella distribuzione dei dati e quindi non è opportuno realizzare modelli predittivi con questa modalità.

Capitolo 5

Conclusioni

In questa Tesi si è voluto realizzare un progetto di manutenzione predittiva utilizzando gli algoritmi di Classificazione e, a questo fine, se ne sono testati alcuni per dimostrare l'adeguatezza del loro impiego.

Inizialmente, gli esperimenti effettuati sui dataset *Gray* e *White* ottenuti dopo una fase di *Preprocessing* e *Data Transformation*, erano decisamente soddisfacenti e, per questo motivo, ci si è accorti che in effetti i dati erano stati creati ad hoc per gli esperimenti, simulando manualmente il livello di tensionamento della cinghia e misurando i valori di corrente consumata dal robot.

Per testare la robustezza degli algoritmi in una situazione più realistica, si è deciso di inserire rumore nei dataset con l'obiettivo di allineare i dati nelle classi, rendendole meno separate. Questo tipo modifica, ha creato dati più simili a quelli che potrebbero generarsi durante il normale funzionamento del robot.

Gli algoritmi testati si sono dimostrati affidabili anche se applicati sui dati con il rumore, ad eccezione della Classificazione Bayesiana e del Support Vector Machine che non è stato applicabile ai dati con rumore. In particolar modo, il Random Forest è risultato l'algoritmo più affidabile. Esso, infatti, è in grado di identificare i segnali anomali di corrente e di distinguerli da quelli che invece rispecchiano il normale funzionamento del robot, anche quando il modello si realizza a partire da un set limitato di dati etichettati. Questo risultato è molto soddisfacente perché dimostra che i classificatori possono essere effettivamente impiegati nell'individuazione delle anomalie che preannunciano il manifestarsi di un possibile guasto. In questo modo, gli addetti dell'azienda, possono intervenire prima che esso si manifesti.

In un secondo momento, si è effettuato l'esperimento della classificazione incrociata, con l'obiettivo di creare modelli predittivi basandosi sull'utilizzo di un determinato dataset e, successivamente, validandolo sull'altro dataset a disposizione.

I risultati ottenuti in questa fase non sono stati soddisfacenti poiché i dati in *Gray* e in *White* presentano alcune differenze nella loro distribuzione e quindi i

modelli realizzati non sono sufficientemente rappresentativi dei dati nel test.

In futuro, sarebbe interessante applicare la metodologia proposta a un dataset reale per verificare ulteriormente la bontà dei classificatori.

Infine, si vogliono mettere in luce alcune criticità legate alla Classificazione. Creare modelli per la manutenzione predittiva basandosi su tecniche di apprendimento supervisionato, implica la necessità di avere a disposizione un set di dati storici di cui si conoscono le etichette di classe. In realtà ciò non basta perché, per poter valutare le prestazioni nel tempo dei modelli predittivi impiegati, i nuovi dati devono anch'essi essere etichettati, in modo tale da poter calcolare le metriche che comunemente sono utilizzate per valutare i modelli. Infatti, l'*Accuratezza*, la *Precisione*, il *Richiamo* e l'$F-measure$ di cui si è fatto largamente uso in questo lavoro, si calcolano effettuando un confronto tra labels predette e labels originali. Generalmente, questa esigenza non può essere soddisfatta poiché il processo di etichettatura dei dati è decisamente impegnativo ed oneroso. Tuttavia, è fondamentale poter valutare le prestazioni nel tempo di un modello predittivo poiché i dati non sono quasi mai statici, ma mutano e si trasformano con il passare del tempo. Il modello di classificazione, però, si forma sulla base delle relazioni che in fase di induzione sono individuate sui dati etichettati noti al momento in cui si effettua il training. Dopodiché, si presuppone che queste relazioni permangano costanti nel tempo. Questa ipotesi è decisamente irrealistica, basti pensare al fatto che all'interno di un'azienda si possono cambiare i macchinari utilizzati oppure essi potrebbero essere soggetti ad usura e malfunzionamenti che alterano la distribuzione dei dati trasmessi al classificatore. Il modello, però, non avendo "mai visto prima" i nuovi dati, potrebbe non riconoscerli ed etichettarli in modo errato. Se non sono note le etichette di classe di questi nuovi dati è impossibile valutare la bontà del modello e quindi individuare quello che viene chiamato *concept drift*, cioè la deriva dei dati.

Un ulteriore interessante sviluppo futuro potrebbe riguardare l'arricchimento della metodologia proposta con lo studio di nuove metriche unsupervised per valutare nel tempo la bontà delle predizioni. Se si riuscissero ad ottenere metriche calcolabili su dati non etichettati per misurare la bontà della classificazione, allora sarebbe possibile individuare il concept drift e cioè l'arrivo di dati con differenti distribuzioni che potrebbe richiedere la creazione di un nuovo modello per effettuare le predizioni. Ad esempio, sarebbe interessante calcolare la *Silhouette* ogni volta che sono trasmessi nuovi dati. Tale metrica, infatti, valuta il grado di coesione e separazione tra clusters e quindi, se con l'arrivo di nuovi dati, il valore di *Silhouette* diminuisce, ciò significa che le classi sono meno separate e coese rispetto alla situazione precedente l'arrivo dei nuovi dati e quindi, probabilmente, si è di fronte al fenomeno del concept drift e occorre effettuare nuovamente il training del

modello.

Bibliografia

[1] Schwab Klaus. *La quarta rivoluzione industriale*. A cura di Franco Angeli. 2016 (cit. alle pp. 1–3).

[2] Cultura Nuova. URL: https://www.culturanuova.net (cit. a p. 1).

[3] SAPERE.IT. URL: http://www.sapere.it/ (cit. a p. 1).

[4] Treccani. *La seconda rivoluzione industriale, L'industrializzazione fra XIX e XX secolo*. URL: http://www.treccani.it/export/sites/default/scuola/lezioni/storia/SECONDA_RIVOLUZIONE_INDUSTRIALE_lezione.pdf (cit. a p. 2).

[5] SAPERE.IT. URL: https://www.sapere.it/sapere/strumenti/studiafacile/geografia-economica/Il-pianeta-uomo-e-la-tecnosfera/Il-sistema-industriale/La-seconda-rivoluzione-industriale-.html (cit. a p. 2).

[6] Varagnolo Roberto. URL: https://www.industry-4.it/ (cit. alle pp. 3, 13, 14).

[7] Mario Hermann, Tobias Pentek e Boris Otto. «Design Principles for Industrie 4.0 Scenarios: A Literature Review». In: (gen. 2015). DOI: 10.13140/RG.2.2.29269.22248 (cit. alle pp. 3, 14, 15).

[8] Bellini Mauro. *I SISTEMI CIBERFISICI*. URL: http://www.factoryofknowledge.net/ (cit. a p. 4).

[9] *Automazione Integrata*. 2015. URL: https://www.automazionenews.it/ (cit. a p. 4).

[10] *Industria Italiana*. 2017. URL: https://www.industriaitaliana.it/nel-cuore-dell-industry-4-0-i-cyber-physical-systems/ (cit. alle pp. 4–6, 8).

[11] *Automation Tomorrow*. 2019. URL: https://www.automationtomorrow.com/cyber-physical-system-cps/ (cit. a p. 6).

[12] Jens Amberg. *Industria 4.0 e sistemi cyber-fisici, Collocazione ed esempio*. 2019. URL: https://www.halstrup-walcher.de/halstrup-walcher-wAssets/docs/pressemeldungen/IT/2015_Fachartikel_Industria-4.0-Cambio-di-formato_IT.pdf (cit. a p. 6).

[13] Jay Lee, Behrad Bagheri e Hung-An Kao. «A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems». In: *SME Manufacturing Letters* 3 (dic. 2014). DOI: 10.1016/j.mfglet.2014.12.001 (cit. alle pp. 6, 7).

[14] Rebecca Mantovani. *Industria 4.0 e sistemi cyber-fisici, Collocazione ed esempio*. 2015. URL: https://www.focus.it/tecnologia/innovazione/tutto-quello-che-ce-da-sapere-sullinternet-of-things-in-x-domande-e-risposte (cit. a p. 9).

[15] Massimo Zanardini e Federico Adrodegari. «Internet delle Cose e servitizzazione: una nuova rivoluzione della manifattura». In: (mar. 2015) (cit. a p. 9).

[16] Figen Balo. «Internet of Things: A Survey». In: *International Journal of Applied Mathematics, Electronics and Computers* (dic. 2016). DOI: 10.18100/ijamec.267197 (cit. a p. 9).

[17] Angela Tumino. URL: https://blog.osservatori.net/it_it/cos-e-internet-of-things (cit. a p. 9).

[18] Dave Evans. *The Internet of Things: How the Next Evolution of the Internet is Changing Everything*. A cura di Cisco. 2011. URL: https://blog.osservatori.net/it_it/cos-e-internet-of-things (cit. a p. 9).

[19] IDC, cur. 2019. URL: https://www.idc.com/%20---%20https://www.idc.com/getdoc.jsp?containerId=prUS44596319 (cit. a p. 9).

[20] Todorovich Piero. 2020. URL: https://www.zerounoweb.it/%20---%20https://www.zerounoweb.it/analytics/big-data/internet-of-things-iot-come-funziona/ (cit. a p. 10).

[21] 2020. URL: https://www.focusindustria40.com/%20---%20https://www.focusindustria40.com/industrial-internet-of-things/ (cit. a p. 10).

[22] URL: https://www.copadata.com/%20---%20copadata.com/it/prodotti/piattaforma-software-zenon-per-lautomazione-industriale-energetica/zenon-differente/significato-di-iot-e-iiot-industrial-internet-of-things/zenon-supervisor-7-8/ (cit. a p. 11).

[23] 2019. URL: https://www.industry4business.it/%20---%20https://www.industry4business.it/ricerche/lera-della-servitizzazione-inizia-con-il-passaggio-da-prodotto-a-servizio/ (cit. a p. 12).

[24] Cosima Rizzi. 2020. URL: https://www.industry4business.it/%20---%20https://www.industry4business.it/servitization/cose-la-servitizzazione-e-come-sta-cambiando-le-strategie-delle-aziende/ (cit. a p. 12).

[25] Clemente Tartaglione Umberto Bettarini Mauro Di Giacomo. «FABBRICHE INTELLIGENTI». In: (2016). A cura di ARES 2.0 (cit. alle pp. 13, 15).

[26] Boston Consulting Group. «Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries». In: (2015) (cit. a p. 16).

[27] Christopher Austin e Fred Kusumoto. «The application of Big Data in medicine: current implications and future directions». In: *Journal of interventional cardiac electrophysiology : an international journal of arrhythmias and pacing* 47 (gen. 2016). DOI: 10.1007/s10840-016-0104-y (cit. alle pp. 17, 18).

[28] Luca Tremolada. A cura di Il SOle 24 Ore. URL: https://www.infodata.ilsole24ore.com/2019/05/14/quanti-dati-sono-generati-in-un-giorno/ (cit. a p. 18).

[29] Alessandro Piva. A cura di Osservatori.net digital innovation. 2019. URL: https://blog.osservatori.net/%20---%20https://blog.osservatori.net/it_it/le-5v-dei-big-data (cit. a p. 18).

[30] URL: https://www.wikiwand.com/%20---%20https://www.wikiwand.com/it/Big_data (cit. a p. 19).

[31] URL: https://www.osservatori.net/it_it/osservatori/comunicati-stampa/mercato-big-data-analytics-italia-valore-trend-comunicato (cit. a p. 21).

[32] URL: https://www.ingenium-magazine.it/big-data-e-business-data-analytics-mercato-doro-e-in-crescita-costante/ (cit. a p. 21).

[33] Timothy Grance Peter Mell. A cura di NIST. 2011. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf (cit. a p. 21).

[34] URL: https://www.hostingtalk.it/lezione-2-lorigine-storica-del-cloud-i-principali-sostenitori_-c000000sh/ (cit. a p. 22).

[35] Vincenzo Ambriola Caterina Flick. *DATI NELLE NUVOLE: ASPETTI GIURIDICI DEL CLOUD COMPUTING E APPLICAZIONE ALLE AMMINISTRAZIONI PUBBLICHE*. 2013 (cit. a p. 23).

[36] Maria Teresa Della Mura. URL: https://www.industry4business.it/industria-4-0/cosa-sono-e-qual-e-il-futuro-dei-cobot-i-robot-collaborativi-che-affiancano-gli-uomini-sulle-linee-di-produzione/ (cit. a p. 24).

[37] LeonelloTrivelli Gloria Cervelli Simona Pira. «Industria 4.0 senza slogan». In: (2017) (cit. alle pp. 25–27).

[38] Treccani. URL: http://www.treccani.it/enciclopedia/realta-aumentata_%5C%28Lessico-del-XXI-Secolo%5C%29/ (cit. a p. 25).

[39] URL: https://www.focusindustria40.com/realta-aumentata-industria-4-0/ (cit. a p. 25).

[40] URL: https://tecnologia.libero.it/industria-4-0-i-vantaggi-della-realta-virtuale-e-realta-aumentata-12634 (cit. a p. 25).

[41] Antonio Teti. In: (2016). URL: https://www.sicurezzanazionale.gov.it/sisr.nsf/wp-content/uploads/2016/06/cyber-security-e-investimenti-Teti.pdf (cit. a p. 27).

[42] URL: https://www.osservatori.net/it_it/osservatori/comunicati-stampa/mercato-sicurezza-informatica-italia-tecnologie-progetti-comunicato (cit. a p. 27).

[43] URL: https://www.innovationpost.it/2019/05/08/nuovi-materiali-e-soluzioni-innovative-ladditive-manufacturing-per-lindustria-4-0/ (cit. a p. 28).

[44] «La manifattura additiva. aLcune vaLutazioni economiche con particoLare riferimento aLL'industria itaLiana». In: (2014). A cura di Centro studi Confindustria. URL: http://www.confindustriasi.it/fabbrica4.0/Cap4.pdf (cit. a p. 28).

[45] URL: http://www.ip4fvg.it/focus-tecnologie-abilitanti-integrazione-orizzontale-e-verticale/ (cit. a p. 29).

[46] URL: https://mynext.it/2018/09/integrazione-verticale-e-orizzontale-dei-sistemi-cosa-significa/ (cit. a p. 29).

[47] URL: http://www.key-4.com/integrazione-orizzontale-e-verticale-nellindustria-4-0-scopriamo-insieme-cose/ (cit. a p. 29).

[48] URL: https://www.digital4.biz/supply-chain/operations-e-plm/manutenzione-predittiva-machine-learning-vantaggi/ (cit. a p. 29).

[49] Ali Rastegari e Antti Salonen. «Strategic maintenance management: Formulating maintenance strategy». In: 18 (gen. 2015) (cit. a p. 29).

[50] URL: https://www.studiofavari.com/2019/07/16/manutenzione-definizioni-uni/ (cit. a p. 29).

[51] URL: https://meccanicatecnica.altervista.org/la-manutenzione/ (cit. a p. 30).

[52] Skövde Runit AB. *Improved remaining useful life estimations for On-Condition parts in aircraft engines*. 2016. URL: https://www.diva-portal.org/smash/get/diva2:945577/FULLTEXT01.pdf (cit. a p. 30).

[53] URL: https://www.azienda-digitale.it/gestione-aziendale/manutenzione-preventiva-e-correttiva-in-cosa-differiscono/ (cit. a p. 30).

[54] URL: https://www.zerounoweb.it/resource-center/data-science-machine-learning/manutenzione-predittiva-cose-e-come-farla-con-intelligenza-artificiale-e-iot/ (cit. a p. 30).

[55] URL: https://leanmanufacturing10.com/it/manutenzione-correttiva-preventiva-e-predittiva-definizioni-e-differenze (cit. a p. 31).

[56] URL: https://www.industry4business.it/industria-4-0/predictive-maintenance-preservare-gli-asset-industriali-con-il-machine-learning/ (cit. a p. 31).

[57] Riccardo Muradore Francesco Cordoni Luca di Persio. «Machine Learning per la Manutenzione Predittiva». In: () (cit. a p. 32).

[58] URL: https://www.developersmaggioli.it/blog/machine-learning-la-scienza-delle-decisioni-automatiche/ (cit. alle pp. 32, 33).

[59] URL: https://www.intelligenzaartificiale.it/machine-learning/ (cit. a p. 33).

[60] URL: https://www.digital4.biz/supply-chain/operations-e-plm/manutenzione-predittiva-machine-learning-vantaggi/ (cit. a p. 33).

[61] Jiawei Han. *Data Mining: Concepts and Techniques 3rd Edition* (cit. alle pp. 34, 37, 41, 57).

[62] URL: https://www.bigdata4innovation.it/data-science/data-mining/data-mining-perche-le-aziende-oggi-non-possono-farne-a-meno/ (cit. a p. 34).

[63] URL: http://www.intelligenzaartificiale.it/data-mining/ (cit. a p. 35).

[64] Edmondo Peron Susi Dulli Sara Furini. *Data mining: Metodi e strategie*. A cura di Springer. 2009 (cit. a p. 35).

[65] Elena Baralis. *Data Mining Fundamentals* (cit. alle pp. 35–38, 41, 44–51, 69).

[66] URL: https://it.wikipedia.org/wiki/Dbscan (cit. a p. 49).

[67] Lior Rokachs Oded Maimon. *Data Mining and Knowledge Discovery Handbook*. A cura di Springer (cit. a p. 51).

[68] Nayer Wanas, Dina Said, Nabila Khodeir, Magda Fayek e Ahmed Gaffer. *DETECTION AND HANDLING OF DIFFERENT TYPES OF CONCEPT DRIFT IN NEWS RECOMMENDATION SYSTEMS*. Feb. 2019 (cit. a p. 52).

[69] Indre Zliobaite. «Learning under Concept Drift: an Overview». In: *CoRR* abs/1010.4784 (gen. 2010) (cit. a p. 52).

[70] Vipin Kumar Michael Steinbach Pang-Ning Tan. *Introduction to Data Mining*. A cura di McGraw Hill. 2006 (cit. alle pp. 53, 55, 59, 61–64, 76).

[71] Elena Baralis. *Classification fundamentals* (cit. alle pp. 55, 56, 58–60, 63, 65–67, 70).

[72] URL: https://lorenzogovoni.com/overfitting-e-underfitting-machine-learning/ (cit. a p. 65).

[73] URL: https://lorenzogovoni.com/random-forest/ (cit. alle pp. 65, 66).

[74] URL: http://www.r-project.it/_book/random-forest-rf.html/ (cit. alle pp. 65, 66).

[75] URL: http://frasca.di.unimi.it/MDSBF18/Lez7.pdf (cit. a p. 67).

[76] URL: https://lorenzogovoni.com/support-vector-machine/ (cit. alle pp. 71–75).

[77] URL: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (cit. alle pp. 71, 73, 74).

[78] URL: https://pandas.pydata.org/ (cit. a p. 77).

[79] URL: https://matplotlib.org/ (cit. a p. 77).

[80] URL: https://scikit-learn.org/stable/ (cit. a p. 77).

[81] URL: https://www.mrwebmaster.it/javascript/json-come-funziona_12795.html (cit. a p. 78).

[82] URL: https://www.ilprogettistaindustriale.it/cinghie-di-trasmissione/ (cit. alle pp. 78, 79).

[83] URL: https://www.ibtinc.com/how-to-tension-a-v-belt/ (cit. a p. 79).

Ringraziamenti

Giunta al termine di questo percorso, mi sembra doveroso ringraziare tutte le persone che mi sono state accanto.

Un grazie particolare lo vorrei dedicare alla Professoressa Tania Cerquitelli, a Paolo Bethaz e Riccardo Callà per essere stati di grande supporto nella stesura di questa Tesi.

Se oggi ho coronato il mio sogno lo devo a mia mamma, mio papà, mia sorella e ai miei nonni, che mi hanno sempre incoraggiata, sostenuta e soprattutto mi hanno dato la possibilità di seguire i miei sogni. Grazie per avermi insegnato il valore del sacrificio e dell'impegno perché si sa, quando le vittorie sono frutto di duro lavoro, hanno sempre un sapore in più.

Grazie ad Angelo, per tutti i consigli e la vicinanza che non mi ha fatto mancare mai. Grazie per essere in ogni momento la mia spalla e il mio punto di riferimento. Grazie per la spensieratezza e la felicità che mi regali ogni giorno.

Infine, grazie a tutti gli amici che ho incontrato in questo percorso. Grazie perché è merito vostro se porterò nel cuore questi anni, nonostante i sacrifici e il duro lavoro. Grazie per tutti i momenti belli e spensierati che hanno lasciato un segno indelebile. Grazie perché con voi, i momenti più duri sono diventati attimi di felicità che ricorderò sempre con grande nostalgia.