UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

TESI DI LAUREA

Definizione di un framework basato su ELK e Python per la gestione dei guasti in un contesto Industry 4.0

Definition of a framework based on ELK and Python for fault management in an Industry 4.0 context

Relatore

Prof. Domenico Ursino

Correlatore

Prof. Francesco Cauteruccio

Candidato

Michele Francesco Pio Ciavarella

ANNO ACCADEMICO 2021-2022

Sommario

La presente tesi ha come obiettivo quello di definire un framework basato sullo *stack ELK* e su *Python* per la gestione di guasti in un contesto di Industria 4.0. Questo lavoro è collocato all'interno del progetto *Intelligence 5.0*, iniziativa intrapresa dal *gruppo Biesse* in collaborazione con l'*Università Politecnica delle Marche*. Biesse è una multinazionale marchigiana leader nella tecnologia per la lavorazione di legno, vetro, pietra, plastica e metallo. Le attività descritte in questa tesi si concentrano sull'analisi di dati di log riguardanti diverse macchine di Biesse forniti dall'azienda stessa. Inizialmente è stato necessario un lavoro di data ingestion per estrapolare dati di interesse e per memorizzarli in forma strutturata in un sistema documentale. Successivamente, è stata portata avanti la parte di analisi dei dati con l'obiettivo di estrarre informazioni utili agli esperti di dominio per individuare eventuali malfunzionamenti delle macchine.

Keyword: Data Analytics, Log Analysis, Extract Trasform and Load, ELK, Python

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

L'*industria 4.0* nasce dalla cosiddetta *quarta rivoluzione industriale*, che sta portando alla produzione industriale interamente automatizzata e interconnessa. Questo termine indica la propensione dell'odierna automazione industriale ad inserire alcune nuove tecnologie produttive per migliorare le condizioni di lavoro, creare nuovi modelli di business, aumentare la produttività degli impianti e migliorare la qualità dei prodotti.

I *Big Data* sono, senza dubbio, uno dei principali cambiamenti che l'Industria 4.0 ha apportato. I vantaggi che possono offrire alle aziende sono numerosi. In particolare, possono consentire all'impresa di aumentare la produttività e, al contempo, di ridurre i costi. L'impiego di oggetti sempre connessi alla rete (*Internet of Things*) sta trasformando le tecnologie di produzione delle aziende, rendendole più flessibili ed efficienti. Le macchine accumulano grandi quantità di dati, utili per monitorare in tempo reale molte aree e, soprattutto, aiutano ad analizzare la situazione aziendale corrente e futura.

Questa tesi si pone l'obiettivo di definire un framework per la rilevazione dei guasti in un contesto di Industria 4.0. Tale lavoro costituisce una piccola parte del progetto *Intelligence 5.0*, iniziativa intrapresa dal gruppo *Biesse* in collaborazione con l'*Università Politecnica delle Marche*, con il sostegno della *Regione Marche* e del *MISE*. Il gruppo Biesse è una multinazionale marchigiana interessata alla produzione e distribuzione di macchinari per la lavorazione di legno, vetro, pietra, plastica e metallo. La sua iniziativa mira alla realizzazione di una rete di sensori applicati alle macchine per attività di diagnostica predittiva. Il sistema non solo fornisce analisi predittive su possibili guasti o malfunzionamenti, ma punta a realizzare un'interfaccia utente intelligente per consentire un utilizzo ottimale dei macchinari da parte dell'operatore.

In questo elaborato di tesi ci si concentra sull'analisi di dati di log provenienti dalle macchine di Biesse. La prima parte del lavoro riguarda l'attività di gestione dei dati. I log forniti sono semplici file di testo, quindi non adatti ad operazioni di data analytics. È stato, quindi, realizzato un prototipo di data ingestion system in cui questi dati vengono estrapolati dai log e memorizzati in forma strutturata all'interno di un sistema documentale. Queste attività sarebbero state difficili da eseguire senza l'ausilio dello *stack ELK* (in inglese "stack", "pila di oggetti"). La sigla "ELK" comprende le tre iniziali dei framework che compongono questa tecnologia, chiamati *Elasticsearch*, *Logstash* e *Kibana*.

La seconda parte del lavoro riguarda, invece, l'attività di analisi dei dati. Il sistema che analizza i dati è stato realizzato tramite il linguaggio di programmazione *Python* e permette di estrarre informazioni relative a varie proprietà di interesse delle macchine. Il tutto viene eseguito per permettere agli esperti di dominio di individuare, dai risultati ottenuti, eventuali malfunzionamenti delle macchine.

La presente tesi è composta da sette capitoli strutturati come di seguito specificato:

- Nel Capitolo 1 saranno introdotti lo stack ELK e, successivamente, il linguaggio di programmazione Python.
- Nel Capitolo 2 verranno presentati la realtà aziendale del gruppo Biesse e il progetto "Intelligence 5.0".
- Nel Capitolo 3 verrà descritta la struttura dei dati dei file di log forniti da Biesse; successivamente, saranno illustrate le attività di ETL eseguite sui dati.
- Nel Capitolo 4 verrà descritta l'architettura del framework realizzato, che è suddiviso in tre componenti che interagiscono tra loro.
- Nel Capitolo 5 verrà illustrata l'implementazione del framework.
- Nel Capitolo 6 verranno trattati l'utilizzo del framework e l'estrazione di knowledge pattern dai risultati ottenuti.
- Nel Capitolo 7 verrà affrontata una discussione critica in merito al lavoro svolto.

Infine, verranno tratte le conclusioni e verranno delineati alcuni possibili sviluppi futuri.

CAPITOLO 1

Introduzione ad ELK e Python

In questo capitolo vengono introdotti lo "stack ELK" e il linguaggio di Programmazione "Python". Per quanto riguarda ELK si discuterà nello specifico dei tre framework che lo compongono, ossia Elasticsearch, Kibana e Logstash, e di come può essere utilizzato nelle attività di analisi dei log. Invece, per ciò che concerne Python, si parlerà della sua storia, delle sue caratteristiche, dei motivi per cui utilizzarlo e dei suoi principali campi di applicazione.

1.1 ELK

ELK è uno stack composto da tre prodotti open source: *Elasticsearch*, *Logstash*, e *Kibana*, tutti sviluppati e gestiti dall'azienda *Elastic* (Figura 1.1). In particolare:

- *Elasticsearch* è un motore di ricerca e analisi basato su *Apache Lucene*;
- *Logstash* è una pipeline di elaborazione dati per acquisire dati da più fonti, trasformarli ed esportarli verso vari target, tra cui Elasticsearch;
- *Kibana* è un'applicazione che consente la visualizzazione e il reporting dei dati indicizzati in Elasticsearch.

Figura 1.1: Logo di Elastic

ELK sta diventando in pochi anni la più comune piattaforma open source di gestione dei log a livello globale (Figura 1.2). Infatti, questo stack è attualmente utilizzato da colossi come Netflix, Uber, Microsoft e da migliaia di altre aziende. Di seguito sono descritti nello specifico i tre componenti dello stack ELK.

Figura 1.2: ELK stack

1.1.1 Elasticsearch

Elasticsearch è un motore di ricerca e analisi distribuito ed è al centro di ELK stack. Dal momento del suo rilascio, ovvero nel 2010, Elasticsearch è diventato rapidamente il motore di ricerca più popolare nel suo ambito e viene impiegato soprattutto per l'analisi dei dati di log e per la ricerca full-text. È basato su *Apache Lucene*, utilizza *API RESTfull*, e lo standard *JSON* per rappresentare le informazioni.

Tramite l'utilizzo di API o di strumenti di importazione dati come Logstash, è possibile effettuare la raccolta, l'aggregazione e l'archiviazione dei dati in Elasticsearch. Esso, infatti, fornisce ricerca e analisi quasi in tempo reale per tutti i tipi di dati. Che si tratti di testo strutturato o non strutturato, dati numerici o dati geospaziali, è possibile archiviarli e indicizzarli in modo efficiente in maniera tale da supportare ricerche rapide. Una volta archiviati i dati, per poter aiutare l'utente nel loro utilizzo, è possibile, tramite delle API, creare query per l'interrogazione del sistema senza dover conoscere il codice Java su cui è basato Elasticsearch.

Elasticsearch offre velocità e flessibilità per gestire i dati in un'ampia varietà di casi d'uso; questi ultimi comprendono:

- aggiungere una casella di ricerca ad un'app o a un sito Web;
- archiviare e analizzare log;
- usare il machine learning per modellare automaticamente il comportamento dei dati in tempo reale;
- automatizzare i flussi di lavoro aziendali utilizzando Elasticsearch come motore di storage;
- gestire, integrare e analizzare le informazioni spaziali utilizzando Elasticsearch come sistema informativo geografico (GIS).

I principali vantaggi nell'utilizzo di Elasticsearch sono i seguenti:

- la sua natura distribuita consente ad esso di elaborare grandi volumi di dati in parallelo, trovando rapidamente le migliori corrispondenze per le query;
- fornisce supporto per vari linguaggi tra cui Java, Python, PHP, JavaScript, Node.js, Ruby e molti altri;
- permette l'esecuzione di operazioni come il rilevamento delle anomalie quasi in tempo reale, grazie alla velocità di esecuzione delle operazioni di lettura e scrittura;

- fornisce un'elevata scalabilità orizzontale;
- è integrato con Kibana, permettendo la visualizzazione e il reporting dei dati, e con Logstash consentendo di eseguire facilmente attività di ETL sui dati di origine e di caricarli nel cluster di Elasticsearch.

1.1.2 Logstash

Logstash è una pipeline di elaborazione dati lato server open source che permette di acquisire dati, trasformarli e inviarli alla destinazione desiderata. I dati sono spesso sparsi in molti sistemi e in molti formati. Logstash permette di importare facilmente dati non strutturati da molteplici fonti, come registri di sistema e registri di siti Web, il tutto in modalità streaming continua. Una grande utilità di Logstash è quella di poter ripulire i dati prima di essere inviati alla destinazione.

Per poter trasformare facilmente i dati ed eseguire attività di ETL, Logstash prevede l'utilizzo di un'ampia varietà di filtri, senza la necessità di dover costruire esternamente pipeline di trasformazione dei dati. Mentre Elasticsearch è solitamente l'output di riferimento per Logstash, esso non è l'unico che può essere scelto. È disponibile, infatti, un elevato numero di formati di output che consentono di instradare i dati con grande flessibilità dove si vuole, a seconda del caso d'uso.

1.1.3 Kibana

Kibana è uno strumento di visualizzazione ed esplorazione dei dati utilizzato per l'analisi dei log, delle serie temporali, dei dati geospaziali, per l'analisi della sicurezza, per il monitoraggio delle applicazioni e altri casi d'uso. Esso consente l'analisi dei dati attraverso la creazione di grafici a barre, grafici a torta, tabelle, istogrammi e mappe. Grazie alla presenza delle dashboard (Figura 1.3, è possibile combinare questi elementi visivi, che possono essere condivisi tramite browser, per fornire visualizzazioni analitiche per grandi volumi di dati in tempo reale. Kibana, inoltre, fornisce una stretta integrazione con Elasticsearch, che lo rende la scelta di default per la visualizzazione dei dati archiviati in Elasticsearch.

Kibana consente l'analisi visiva dei dati provenienti da uno o più indici Elasticsearch. Gli indici vengono creati quando Logstash acquisisce dati non strutturati da file di log e altre origini e li converte in un formato strutturato per le funzionalità di archiviazione e ricerca di Elasticsearch. L'interfaccia di Kibana consente agli utenti di interrogare i dati negli indici Elasticsearch e, quindi, visualizzare i risultati. Gli utenti possono scegliere tra diversi tipi di grafici, modificare le aggregazioni dei dati e filtrarli in base a ciò che si vuole ottenere.

1.1.4 Installazione ed avvio di ELK stack

In questa sezione vengono mostrati l'installazione ed un primo avvio dello stack ELK sul sistema operativo *Windows*. La versione di Elasticsearch, Kibana e Logstash utilizzata per questo scopo è la `7.17.2`, rilasciata nel marzo del 2022. L'unico prerequisito per l'installazione dello stack sul sistema operativo è la necessità di disporre di *Java JDK*. Una volta installato Java, bisogna accedere al link `elastic.co/downloads` e scaricare i tre componenti di ELK nel formato `.zip`. Dopo che tali componenti sono stati estratti nella posizione desiderata, l'installazione è completata ed è possibile l'avvio.

Elasticsearch

Inizialmente, procediamo con l'avvio di Elasticsearch. Per farlo bisogna avviare il terminale Windows, posizionarsi nella cartella `elasticsearch-7.17.2/bin` ed eseguire

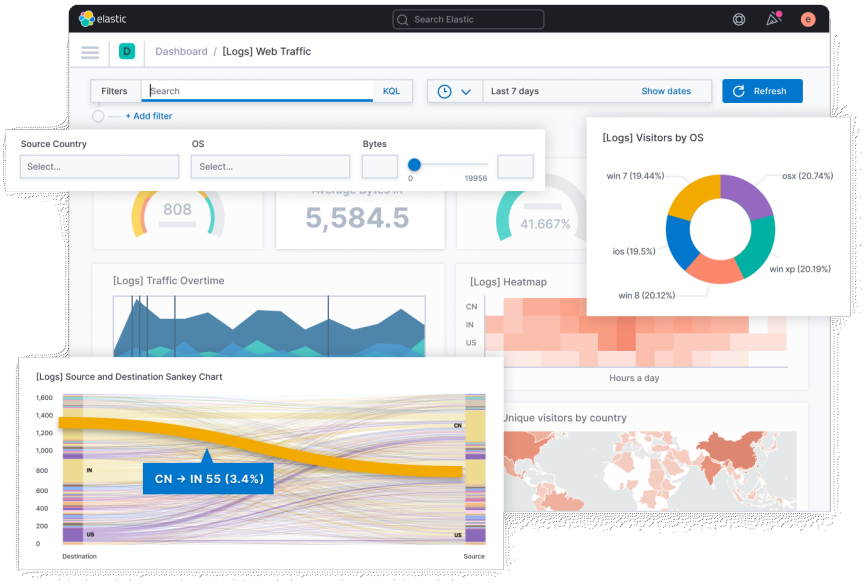


Figura 1.3: Esempio di dashboard in Kibana

`elasticsearch.bat`. Dopo aver atteso alcuni secondi, per certificare l'effettivo avvio occorre digitare, servendosi di un qualsiasi browser, l'indirizzo `localhost:9200`. Se l'avvio è avvenuto nella maniera corretta, vengono visualizzate una serie di informazioni, come il nome del nostro sistema o la versione di Elasticsearch (Figura 1.4).

```
{
  "name" : "DESKTOP-RC47958",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "-_Vie24XTWymt-p2U3PuNQ",
  "version" : {
    "number" : "7.17.2",
    "build_flavor" : "default",
    "build_type" : "zip",
    "build_hash" : "de7261de50d90919ae53b0eff9413fd7e5307301",
    "build_date" : "2022-03-28T15:12:21.446567561Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Figura 1.4: Avvio di Elasticsearch

Kibana

Nel passo successivo ci occupiamo dell'avvio di Kibana. Come già fatto per Elasticsearch, bisogna avviare il terminale, posizionarsi nella cartella `kibana-7.17.2/bin` ed eseguire `kibana.bat`. Anche in questo caso, per potersi accertare che il tutto funzioni correttamente, occore andare sull'host locale, ma questa volta sulla porta *5601*. Se l'avvio è andato a buon fine, saremo in grado di visualizzare la home page di Kibana, come mostrato in Figura 1.5.

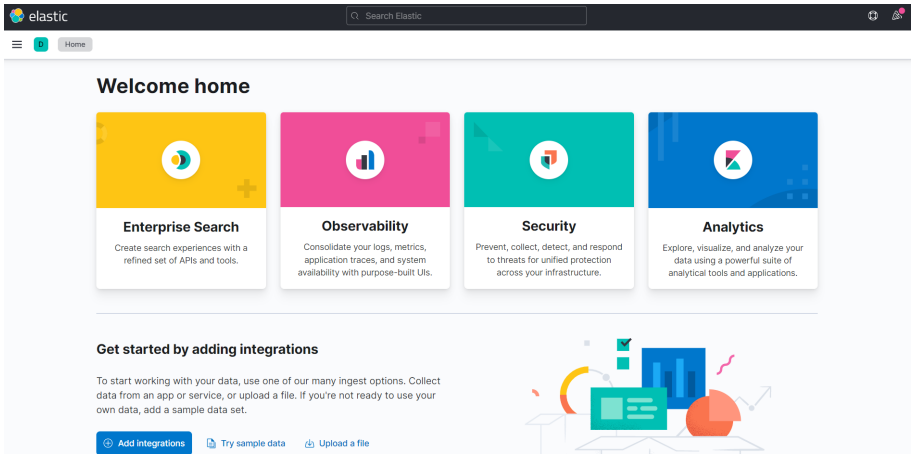


Figura 1.5: Avvio di Kibana

Logstash

Ora vediamo come è possibile utilizzare Logstash con Elasticsearch e Kibana. Prima di eseguire il file `logstash.bat` nella cartella `logstash-7.17.2/bin`, come fatto in precedenza, occorre aggiungere i path `elasticsearch-7.17.2/bin`, `kibana-7.17.2/bin` e `logstash-7.17.2/bin` nella variabili di ambiente del sistema. Ciò viene fatto per permettere di eseguire `logstash.bat` in qualsiasi punto del file system utilizzando il terminale. Il passaggio successivo consinste nel creare un file di configurazione con l'estensione `.conf`, adeguatamente predisposto per poter segnalare a Logstash cosa fare. Questo file può essere creato in qualsiasi posizione del file system; non è necessario che debba essere all'interno della cartella `logstash-7.17.2`. All'interno del file ci sono tre importanti blocchi: input, filter e output:

- all'interno della sezione "input" vengono poste le istruzioni necessarie a Logstash per prelevare i dati;
- nel blocco "filter" possono essere creati una serie di filtri che consentono di creare, rimuovere e trasformare campi, oltre a modellare quelli già esistenti;
- la sezione "output" contiene le istruzioni che indicano a Logstash dove esportare gli eventi, dopo che questi sono stati creati e trasformati tramite i filtri.

La sezione relativa a "filter" è opzionale, quindi per questo semplice esempio viene omessa. Nella Figura 1.6 è possibile vedere un esempio di un file `.conf` nel quale l'input viene fornito semplicemente tramite lo standard input. All'interno della sezione output si può notare che i dati vengono inviati al server Elasticsearch; inoltre, viene fornito un indice dei dati grazie all'istruzione `index => "index-for-logstash"`.

Per poter avviare Logstash occorre, dopo aver aperto il terminale, posizionarsi nel punto in cui è stato creato il file `.conf` e digitare: `logstash -f filename.conf`. All'interno del comando è necessario un suffisso per segnalare la modalità di esecuzione del framework. Il suffisso utilizzato in questo caso è `-f`, che fornisce in ingresso proprio un file di configurazione segnalando a Logstash cosa fare.

Dopo l'avvio bisogna digitare sul terminale un testo (dal momento che viene utilizzato lo standard input) che verrà preso in input da Logstash. Una volta che tutto ciò avrà avuto esito positivo ci si dovrà spostare tramite browser su Kibana, in particolare nella sezione che ci permette di creare un modello di indice. Infatti, si potrà creare un nuovo indice dei dati scegliendo il nome *index-for-logstash*, che è stato precedentemente esplicitato nel file `.conf`. Infine, per visualizzare i dati inseriti, occorre posizionarsi nella sezione *Discover* di Kibana.

```
input {
    stdin {}
}

output {
    elasticsearch {
        hosts => ["localhsot:9200"]
        index => "index-for-logstash"
    }
}
```

Figura 1.6: Esempio di file di configurazione Logstash

1.2 Python

Python (Figura 1.7) è un linguaggio di programmazione dinamico, ad alto livello e orientato agli oggetti, utilizzabile per molti tipi di sviluppo software. Esso offre un forte supporto all'integrazione con altri linguaggi e programmi e possiede un'estesa libreria standard. Python gira su Windows, Linux/Unix, Mac OS X, OS/2, Amiga, palmari Palm e cellulari Nokia; inoltre è possibile utilizzarlo anche sulle macchine virtuali Java e .NET. È distribuito con licenza Open-Source approvata dalla OSI, rendendolo gratuito e aperto a tutti.

Figura 1.7: Logo di Python

Secondo l'indice *TIOBE*, molto utilizzato per queste classifiche, Pyhton risulta essere il linguaggio di programmazione più popolare fino a Luglio 2022, come mostrato in Figura 1.8. Questo indice, che è un indicatore di popolarità dei linguaggi di programmazione, si basa sulle ricerche eseguite dagli utenti su 25 motori di ricerca, tra cui Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube e Baidu. È importante sottolineare che l'indice TIOBE non riguarda il *miglior* linguaggio di programmazione o il linguaggio in cui sono state scritte la *maggior parte delle righe di codice*.

1.2.1 Storia

La storia di Python è strettamente legata al linguaggio di programmazione *ABC*, sviluppato ad Amsterdam, presso il *CWI* (Centrum Wiskunde & Informatica). Python è stato concettualizzato alla fine del 1980. Il suo creatore, Guido van Rossum (Figura 1.9), a quel tempo lavorava in un progetto al CWI, chiamato *Amoeba*, un sistema operativo distribuito. Per sottolineare l'importanza che avuto ABC per Guido Van Rossum, in un'intervista con Bill Venners ha dichiarato: "Nei primi anni 80, ho lavorato come implementatore in un team building di un linguaggio chiamato ABC presso il Centrum voor Wiskunde en Informatica (CWI). Non so quanto bene la gente conosca l'influenza di ABC su Python. Cerco di menzionare l'influenza di ABC perché sono in debito con tutto ciò che ho imparato durante quel progetto e con le persone che ci hanno lavorato". Come dichiarato nella stessa intervista,

Jan 2022	Jan 2021	Change	Programming Language	Ratings	Change
1	3	↑	Python	13.58%	+1.86%
2	1	↓	C	12.44%	-4.94%
3	2	↓	Java	10.66%	-1.30%
4	4		C++	8.29%	+0.73%
5	5		C#	5.68%	+1.73%
6	6		Visual Basic	4.74%	+0.90%
7	7		JavaScript	2.09%	-0.11%
8	11	↑	Assembly language	1.85%	+0.21%
9	12	↑	SQL	1.80%	+0.19%
10	13	↑	Swift	1.41%	-0.02%

Figura 1.8: Indice TIOBE Luglio 2022

il suo obiettivo era quello di provare a progettare un linguaggio di scripting semplice che possedesse alcune delle migliori proprietà di ABC, ma senza i suoi problemi.

Figura 1.9: Guido Van Rossum, creatore di Python

Per quanto riguarda l'origine del nome "Python", anche se la maggior parte delle persone potrebbe pensare che ci sia un legame con i serpenti, in realtà il nome fu scelto per la passione del suo inventore verso i *Monty Python*, un gruppo comico brittannico, e per la loro serie televisiva *Monty Python's Flying Circus*.

Guido Van Rossum pubblicò la prima versione del codice Python (Versione 0.9.0) nel febbraio del 1991. Questa versione includeva già la gestione delle eccezioni, le funzioni e i tipi di dati principali di liste, *dict*, *str* e altri. Era anche orientato agli oggetti e aveva un sistema di moduli.

Python Versione 1.0 è stato rilasciato nel gennaio del 1994. Le principali novità incluse in questa release furono gli strumenti di programmazione funzionale *map*, *filter* e *reduce*.

Sei anni e mezzo dopo, nell'ottobre del 2000, è stato introdotto Python 2.0 che venne sviluppato per altri 8 anni nelle Versioni 2.x prima che venisse rilasciata la successiva versione

principale: Python 3.0 (noto anche come "Python 3000" e "Py3K"). L'enfasi in Python 3 fu sulla rimozione di costrutti e moduli di programmazione duplicati, soddisfacendo così, o avvicinandosi a soddisfare, la legge 13 dello Zen di Python: "Ci dovrebbe essere un, e preferibilmente solo un, modo ovvio per farlo".

1.2.2 Caratteristiche

Python è un linguaggio di programmazione di alto livello. Questa definizione viene utilizzata per quei linguaggi fortemente astratti, le cui istruzioni si avvicinano di più al linguaggio naturale che a quello macchina. Le istruzioni di questi linguaggi per poter essere eseguite, hanno ovviamente bisogno di essere convertite in linguaggio macchina: ciò viene fatto attraverso l'interpretazione e la compilazione. Il programmatore, infatti, non deve occuparsi degli aspetti di "basso livello" legati all'hardware; il programma interprete di Python si occupa di tradurre le righe scritte in Python in istruzioni direttamente comprensibili al computer (linguaggio macchina).

Le caratteristiche principali di Python sono le seguenti:

- Supporta diversi paradigmi di programmazione, come quello object-oriented (con supporto all'ereditarietà multipla), quello imperativo e quello funzionale, ed offre una tipizzazione dinamica forte. Possiede una ricca libreria built-in che, unitamente alla gestione automatica della memoria e ad un'efficiente gestione delle eccezioni, fa di esso uno dei linguaggi più usati.
- Risulta essere molto comodo e semplice da usare. Python, infatti, nelle intenzioni di Guido van Rossum, è nato per essere un linguaggio immediatamente intuibile. La sua sintassi è, in generale, pulita e snella, così come i suoi costrutti.
- È un linguaggio pseudocompilato: un interprete si occupa di analizzare il codice sorgente (file testuali con estensione `.py`) e, se sintatticamente corretto, di eseguirlo. In Python, non esiste una fase di compilazione separata (come avviene in C, per esempio) che generi un file eseguibile partendo dal sorgente.
- La sua natura di linguaggio pseudocompilato rende Python mutipiattaforma. Una volta scritto un sorgente, esso può essere interpretato ed eseguito sulla maggior parte delle piattaforme più popolari, siano esse di casa Apple, Microsoft o Linux. È necessaria soltanto la versione giusta dell'interprete installata sulla macchina.
- È un linguaggio general purpose: Python può essere utilizzato per la realizzazione di applicazioni di vario genere e per la risoluzione di un'ampia gamma di problemi.
- È free software ed open source: il download dell'interprete per la propria macchina e l'uso del linguaggio stesso nelle proprie applicazioni è completamente gratuito. Inoltre, il codice sorgente di Python può essere visto da tutti e può essere liberamente modificato e ridistribuito, secondo le regole di una licenza pienamente open source.

1.2.3 Perché Python

Python è un linguaggio di programmazione molto utilizzato, in questa sezione si cercherà di capirne i motivi e si analizzeranno i punti di forza e le sue molteplici possibili applicazioni.

Anche se, ovviamente, di parte, sul sito della comunità italiana di Python (`https://www.python.it/`), gruppo molto vivace e attivo, sono elencati *cinque buoni motivi per scegliere Python*. Tali motivi possono essere ritenuti, in generale, corretti:

- *Facile da imparare*: la sintassi e la struttura del linguaggio sono abbastanza semplici da imparare e utilizzare. Questo soprattutto per chi si avvicina per le prime volte al mondo della programmazione, che incontrerà una sintassi non molto complessa da comprendere.
- *La versatilità è il suo forte*: Python è un linguaggio che può essere utilizzati per molteplici scopi. Utilizzando direttamente Python (nativo) o usando librerie/framework, si potranno coprire svariati ambiti che vanno dallo sviluppo web al machine learning, passando per GUI ed applicazioni server.
- *Funziona su molte piattaforme*: Python è multipiattaforma, gira senza problemi su principali sistemi operativi: GNU/Linux, MacOS, Windows e Unix.
- *La popolarità e la ricerca di personale*: Python è considerato il linguaggio di programmazione maggiormente in crescita in questi ultimi anni. Essendo molto utilizzato, è presente una grande comunità che potrebbe aiutare i nuovi utenti nel processo di apprendimento. Inoltre, in ambito lavorativo c'è anche grande richiesta di programmatori in Python, o anche esperti in specifiche librerie.
- *Librerie e framework*: Uno dei punti di forza di Python è quello di avere una vastissima quantità di librerie e framework. Il linguaggio, però, include al suo interno tanti strumenti che permettono di svolgere varie attività, come l'accesso ai database, la lettura e la scrittura di file (ad esempio i file Json), la creazione di interfacce grafiche, l'accesso al sistema operativo per operazioni su file e cartelle, ed altre svariate tipologie di applicazioni.

Un altro vantaggio non menzionato in questi *cinque buoni motivi*, è sicuramente, la sua natura open source; infatti Python è completamente free e il suo uso e la sua distribuzione sono liberi da copyright. Inoltre, supporta, come molti altri linguaggi, diversi paradigmi di programmazione, come la programmazione ad oggetti, la programmazione procedurale e la programmazione funzionale. Infine, Python è integrabile con altri linguaggi: esistono interpreti che consentono l'integrazione con diversi altri linguaggi. *IronPython*, ad esempio, consente di utilizzare Python all'interno del framework .NET, usandone le sue funzioni, e interagendo con altri linguaggi .NET.

Applicazioni

Come già esplicitato in precedenza, grazie alla vasta quantità di framework e di librerie, sia built-in che di terze parti, è possibile utilizzare Python in diversi ambiti e applicazioni.

Si tenga presente che esistono molteplici possibilità per lo sviluppo Web sia ad alto che a basso livello. Per realizzare siti ed applicazioni web sono disponibili diversi framework come:

- *Django*: uno dei framework web più usati, che offre svariati strumenti per la realizzazione di siti e applicazioni web;
- *Flask*: un "microframework" che permette di creare in maniera veloce siti semplici;
- *Web2py*: un altro ottimo framework non complicato da usare.

È possibile usare Python anche per accedere ai database. Sono supportati nativamente diversi database (relazionali e non relazionali). Sostanzialmente, è possibile interfacciarsi con qualsiasi database esistente; tra i principali vi sono: Oracle MySQL, PostgreSQL, SQL Server e SQLite.

Come altre possibili applicazioni si segnalano la possibilità di realizzare interfacce grafiche (GUI) usando tutti i maggiori toolkit, come Tkinter, PyQt, PyGtk e wxPython, e lo sviluppo software, fornendo importanti strumenti e framework, come Trac, Buildbot e Roundup.

1.2.4 Pyhton nella data analysis e nel data mining

Importanti settori in cui viene spesso preferito Python rispetto ad altri linguaggi è certamente quello della *data analysis* e quello del *data mining*. In un sondaggio pubblicato a maggio 2018 dall'autorevole portale *KDNuggets*, nella categoria "Top Analytics, Data Science, Machine Learning Tools", è emerso che Python è utilizzato dal 65,2% dei circa 2000 intervistati, contro il 52,7% di RapidMiner e il 48,5% di R, i suoi due principali concorrenti. La sua popolarità deriva probabilmente dalla relativa facilità d'uso e dall'enorme ecosistema costituito da un numero molto vasto di librerie in questo ambito.

Il data mining (DM) si occupa della preparazione dei dati ottenuti da varie fonti informative (ad esempio, database o file di testo), nonché della modellazione dei dati utilizzando una varietà di tecniche, a seconda dell'obiettivo che si vuole raggiungere (ad esempio, classificazione, clustering, regressione, regole associative, etc.). Per raggiungere gli obiettivi prefissati, il DM utilizza tecniche di *machine learning* (ML) per estrarre nuove conoscenze dalle informazioni esistenti. Il DM, al giorno d'oggi, comprende vari ambiti, tra i quali la statistica, i big data e la visualizzazione dei dati. Fondamentale, nel processo di analisi dei dati, è il preprocessing, che mira a pulire, integrare, trasformare e migliorare la qualità dei dati grezzi originali in modo che possano essere utilizzati e manipolati dagli algoritmi di modellazione.

Di seguito poniamo il focus su quattro gruppi di librerie, ovvero data preparation, data visualization, machine learning e deep learning, che ci permetteranno di comprendere meglio l'importanza di Python nel settore.

Data preparation

Attualmente la libreria Python migliore e più utilizzata nel campo della *data preparation* è *Pandas*. Essa ha una vasta gamma di funzionalità per formati di dati di input/output, come Excel, CSV, Python/NumPy, HTML, SQL, etc. Inoltre, Pandas ha potenti funzionalità di interrogazione, di calcolo statistico e di visualizzazione di base. Ha una ricca documentazione, ma una sintassi un po' confusa, che viene spesso indicata come il suo difetto più significativo.

Data visualization

Le principali librerie utilizzate per la *data visualization* sono *Plotly*, *seaborn* e *Matplotlib*. Le ultime due risultano essere più semplici da imparare e da utilizzare per i principianti e Matplotlib consente di realizzare grafici più complessi e più personalizzabili. Tra le tre la più completa sembra essere Plotly. Il suo principale difetto è una sintassi relativamente poco intuitiva, che rende più difficile l'apprendimento. Tuttavia, il difetto è compensato con una documentazione molto ricca che fornisce numerosi esempi. Un suo vantaggio è anche quello di poter integrare i suoi grafici nelle pagine web con *Dash*, realizzando, come suggerisce il nome del framework, delle dashboard.

Machine Learning

Scikit-learn è la libreria Python più popolare per il machine learning. Buone alternative ad essa, anche se decisamente meno utilizzate, sono *mlxtend*, una nuova e piccola libreria che include solo pochi algoritmi di base, e *Shogun*, che è scritta principalmente in C++, ma

è disponibile un wrapper Python per tutte le sue funzionalità. Scikit-learn propone vari algoritmi come il *Support Vector Machine*, il *Random Forests* e il *K-Neighborhoods* e coopera con alcune librerie numeriche e scientifiche di Python, quali *NumPy* e *SciPy*. Inoltre, dei punti a suo favore sono la vasta comunità che la utilizza e una documentazione intuitiva.

Deep Learning

Una delle librerie più utilizzate nel deep learning è *Tensorflow*. Sviluppata da *Google Brain*, possiede vaste funzionalità, una buona documentazione e con essa il codice è molto personalizzabile. Altre due librerie molto popolari nel settore sono *Keras*, che è costruita su TF, e *Pytorch*, libreria sviluppata di recente da Facebook con una comunità rapidamente in crescita.

CAPITOLO 2

Descrizione del contesto di riferimento

In questo capitolo si descrive la realtà aziendale del gruppo Biesse e il progetto Intelligence 5.0, che è stato intrapreso dal gruppo in collaborazione con l'Università Politecnica delle Marche, con il sostegno della Regione Marche e del MISE. In particolare, vengono descritti l'ambito tecnologico del progetto, la sua finalità e i suoi obiettivi realizzativi.

2.1 Biesse

Biesse (Figura 2.1) è una multinazionale che progetta, produce e distribuisce sistemi e macchine per la lavorazione di legno, vetro, pietra, metallo, materiali plastici e compositi per i settori furniture, housing & construction, automotive e aerospace.

L'azienda è stata fondata a Pesaro nel 1969 da Giancarlo Selci ed è quotata nel segmento Star di Borsa Italiana dal 2001. Biesse è ormai da anni una presenza costante nei principali mercati mondiali; infatti, grazie ai suoi 13 stabilimenti produttivi, opera in più di 160 Paesi e circa l'85% del suo fatturato è realizzato all'estero. Inoltre, annovera fra i suoi clienti aziende di riferimento nelle proprie Industry e prestigiosi brand del design italiano ed internazionale.

Figura 2.1: Logo di Biesse

Dal 1969 il gruppo Biesse è specializzato nella produzione e commercializzazione di macchine per la lavorazione del legno. I settori verso cui sono rivolti i suoi prodotti e le sue tecnologie in questo ambito sono l'industria del mobile, del serramento e dei componenti in legno per l'edilizia. Biesse propone, anche, soluzioni per la lavorazione di materiali tecnologici, per l'imballaggio, l'edilizia e la visual communication, lavorando materie plastiche espanse e compatte, materiali compositi e cartone.

Nel 1987 il gruppo ha iniziato a produrre macchinari per altri tipi di materiali, quali vetro, pietra e metallo. L'azienda, in quegli anni, è di fatto diventata un brand di riferimen-

to per aziende che operano in diversi comparti industriali, dall'edilizia all'arredamento, dall'automotive fino al settore dell'aeronautica.

Biesse produce diverse componenti tecnologicamente avanzate, come elettromandrini, teste a 5 assi e rinvii angolari, che permettono di avere performance elevate e vantaggio competitivo ai propri clienti. Tutti questi elementi fondamentali per la produzione di metallo, leghe metalliche, materiali compositi e legno sono destinati ai produttori di automotive, aerospace, consumer electronics, robotics, furniture & building.

2.2 Progetto Intelligence 5.0

Questo lavoro di tesi si concentra su una parte del progetto *Intelligence 5.0*, intrapreso dal *gruppo Biesse* in collaborazione con l'*Università Politecnica delle Marche*, con il sostegno della *Regione Marche* e del *MISE*. Il settore di applicazione è quello della fabbrica intelligente o digital manufacturing, in particolare dell'Intelligence 5.0, soluzione avanzata IoT basata su una rete di sensori collaborativi applicati ai componenti delle macchine a controllo numerico per attività di diagnostica predittiva. Il sistema non solo fornisce analisi predittive su possibili guasti o malfunzionamenti dei macchinari, ma – e qui entra in gioco lo step 5.0 – si pone come interfaccia utente intelligente dell'operatore per un utilizzo ottimale dei macchinari.

2.2.1 Descrizione del progetto

Grazie alle tecnologie introdotte dall'Industria 4.0, le imprese sono agevolate, in quanto hanno la possibilità di raccogliere un'elevata quantità di informazioni sui loro processi e prodotti, utili a migliorare e ampliare la loro offerta attraverso l'introduzione di nuovi servizi. In particolare, servizi di diagnostica e manutenzione predittiva possono portare enormi benefici in termini di riduzione dei costi di ciclo di vita e di aumento dell'affidabilità delle macchine. Purtroppo questi servizi sono ancora poco diffusi nelle industrie, poiché richiedono complesse tecnologie, ingenti investimenti e il coinvolgimento di data scientist, che spesso non sono presenti in azienda; inoltre, l'ambito di ricerca relativo alla prognostica delle macchine è ai suoi primi anni di vita e, quindi, non fornisce metodologie consolidate per la loro applicazione pratica. Bisogna, anche, considerare che il rischio di investimento è elevato, in quanto spesso è difficile monetizzare i servizi, poiché essi sono sviluppati considerando più le caratteristiche dei dati disponibili che i reali bisogni degli utilizzatori.

Da queste necessità nasce il progetto Intelligence 5.0, che ha come suo titolo completo *Intelligence 5.0: dai sistemi cyber-fisici per la creazione di macchine utensili "self-aware" ad innovativi modelli di servizi industriali evoluti*. Esso mira ad offrire una risposta organica alle attuali *esigenze di mercato dell'industria meccatronica*, mettendo in pratica un approccio olistico in grado di migliorare il livello di *affidabilità delle macchine e al contempo garantire l'ottimizzazione ed efficientamento dei processi produttivi* da esse abilitati, grazie all'utilizzo combinato ed integrato dei nuovi paradigmi dell' *Industria 4.0: Digital Twin (DT), Artificial Intelligence (AI), Knowledge Management (KM) e Augmented Reality (AR)*, ovvero di quelle tecnologie abilitanti (KET) che afferiscono alla sfera dell'informazione e della comunicazione (ICT), coerenti con la Strategia Nazionale di Specializzazione della *Fabbrica Intelligente*.

Questo progetto punta ad adottare un approccio *customer-centric* per definire un modello di servizio diagnostico delle macchine in grado di rispondere alle esigenze degli utilizzatori (manutentori, customer service e operatori a bordo macchina) e tecniche di *augmented reality* per supportare l'operatore nello svolgimento delle attività garantendo la sua sicurezza. Il tutto verrà fatto sovrapponendo ed adattando i contenuti digitali (ad esempio, le istruzioni) alle parti fisiche della macchina visualizzando solo le informazioni necessarie.

L'obiettivo del progetto è quello di definire e sviluppare macchine di nuova generazione denominate *self-aware* che implementino un framework capace di:

- Acquisire i dati di lavorazione (informazioni di processo) e i dati sullo stato operativo dei vari sistemi/sottosistemi/componenti delle macchine in tempo reale. Essi vengono raccolti e immessi nello spazio "cibernetico" per l'analisi e l'elaborazione basata sull'esperienza.
- Elaborare questi dati utilizzando capacità di calcolo avanzate, anche di tipo cognitivo; queste conoscenze possono essere integrate con informazioni sulle prestazioni (KPI di produzione, di qualità etc.). Il progetto prevede di integrare strumenti come sistemi Multiagente o architetture di Sistemi Olonici con il paradigma dei CPS per rendere il sistema Cyberfisico il più possibile reattivo ed auto-adattivo ai cambiamenti dei processi produttivi. In questo modo, il processo di raccolta e analisi di grandi quantità di dati e l'estrazione di informazioni nascoste saranno un aiuto efficace per gli strumenti principali di reasoning dell'AI.
- Garantire l'efficienza e l'affidabilità della macchina, assicurare la riconfigurabilità del processo produttivo in base all'efficienza residua dei vari sistemi/ sottosistemi/ componenti delle macchine, in modo da migliorare l'efficienza del processo, riducendo al minimo i tempi di inattività. Tutto questo sarà possibile grazie alla identificazione e alla segnalazione di eventuali azioni da intraprendere (ad esempio, interventi correttivi sui parametri di processo, interventi di manutenzione, etc.) da parte della macchina. Inoltre, sarà possibile assicurare l'allocazione ottimale delle risorse in base alle esigenze di manutenzione.
- Fornire "insights" ai vari stakeholder nel ciclo di vita delle macchine connesse, come indicazioni per migliorare la qualità dei componenti e sottosistemi, aumentare l'efficienza complessiva dei processi, ottimizzare la logistica e fidelizzare i clienti attraverso servizi innovativi (riparazione, manutenzione, logistica).

Questo programma fa parte di un filone di ricerca avviato da Biesse nel 2017 e formalizzato con una serie di contratti di ricerca con UNIVPM. Dal 2017 ad oggi, l'azienda ha delineato la basi metodologiche e tecniche per l'organizzazione del programma Intelligence 5.0; inoltre, ha svolto ricerche sullo stato attuale della tecnologie delle macchine e sviluppato diversi prototipi e soluzioni assolutamente visionari sui temi dell'Intelligenza Artificiale e dell'Interazione Uomo-Machina adattiva e context-aware. Ciò dimostra l'importanza che l'azienda attribuisce alla ricerca e allo sviluppo per l'Industria 4.0, in quanto vuole rendere le soluzioni automatizzate alla portata di tutti, dalle piccole aziende alle grandi realtà produttive.

2.2.2 Ambito tecnologico

Gli obiettivi del progetto sono inerenti allo sviluppo di *Tecnologie dell'informazione e della comunicazione (ICT)* e si riferiscono al settore applicativo *Fabbrica Intelligente*, in quanto esso mira a sviluppare soluzioni all'interno dell'ambito *Industria intelligente e sostenibile, energia e ambiente*, considerando e integrando le seguenti sotto-tematiche applicate alla realtà industriale:

- soluzioni per la gestione integrata della manutenzione;
- soluzioni ICT per la valorizzazione e condivisione della conoscenza all'interno delle fabbriche;

- tecnologie e applicazioni di Realtà Virtuale/Aumentata per la gestione del prodotto-processo sistema;
- modelli di business innovativi basati sull'offerta integrata di prodotto-servizio;
- Cyber-Physical Systems (CPS) per la fabbrica intelligente;
- sistemi di supervisione e controllo dei processi industriali;
- interazione intelligente uomo-macchina;
- strumenti di simulazione integrati per il virtual commissioning di sistemi di produzione;
- macchine intelligenti;
- ICT per lo sviluppo model-based di macchinari;
- piattaforme integrate digitali per la configurazione di sistemi di produzione;
- strumenti per il supporto alle decisioni in ambienti complessi.

Nel progetto verranno adottate tecnologie legate a *Industrial Internet of Things (IIoT), Cyber Physical Systems, sistemi Cloud, Big Data Analytics*, che saranno utilizzate per la realizzazione di un modello matematico in grado di realizzare delle analisi avanzate. Verranno integrati l'intelligenza artificiale, il machine learning ed il software analytics con i dati raccolti negli impianti produttivi per creare modelli di simulazione digitale che si aggiornano quando cambiano i parametri dei processi produttivi o le condizioni di lavoro.

Ciò che si punta a fare è sviluppare un sistema di autoapprendimento che utilizza i dati raccolti da varie fonti, tra cui sensori che trasmettono le condizioni operative, esperti come ingegneri con esperienza nel settore, altre macchine simili e dati storici relativi al passato della macchina.

Per innalzare i livelli di performance delle aziende nelle fasi di produzione e di manutenzione, è diventata molto importante la simulazione, con la possibilità di realizzare un prototipo virtuale. Inoltre, con l'avvento delle piattaforme software, è possibile integrare dati in tempo reale con tutte le informazioni digitali che un'azienda ha su un determinato processo.

2.2.3 Finalità e obiettivo finale del progetto

La finalità del progetto Intelligence 5.0 è quella di migliorare l'affidabilità delle macchine, implementando un approccio olistico che garantisca l'ottimizzazione e l'efficienza dei processi produttivi, rispondendo così alle attuali esigenze del mercato di Biesse e più in generale alle necessità delle aziende clienti (principalmente produttori del settore legno-arredo, ma anche dell'industria del vetro e della pietra e di coloro che operano con i materiali tecnologici).

Le soluzioni proposte dal progetto rappresentano per il settore interessato elementi significativi in tema di innovazione e riguardano in particolare:

- L'introduzione di una nuova generazione di macchine "self-aware" dotate di un framework tecnologico per il rilevamento delle anomalie e la diagnostica predittiva che combina tecniche di simulazione Digital Twin (DT), algoritmi Machine Learning (ML) e regole knowledge-based (KB), in grado di mettere a sistema le pratiche e conoscenze aziendali. Attualmente non esistono sul mercato macchine che utilizzano il DT per la diagnosi o per la manutenzione predittiva. L'obiettivo è quello di sfruttare il concetto di DT per implementare un approccio di *Predictive Health Monitoring (PHM)* degli asset critici delle macchine.

- Sviluppo di un nuovo sistema di interfaccia utente collaborativo, self-aware e human-aware, capace di monitorare in tempo reale l'avanzamento dei processi. Grazie all'utilizzo dell'*eXtented Reality (XR)*, sarà possibile guidare l'operatore nell'esecuzione delle operazioni. Attualmente, ci sono molti studi che dimostrano come queste tecnologie potrebbero aiutare molto le industrie ma, nonostante ciò, esse sono state ancora adottate da pochissime aziende. Ciò dimostra che l'integrazione dei sistemi XR nella produzione rimane tuttora difficile e impegnativa, ma risulta essere essenziale nella trasformazione digitale della produzione.
- Aumento dell'efficienza dei processi produttivi. La nuova interfaccia uomo-macchina che verrà sviluppata sarà in grado di fornire ai diversi operatori informazioni istantanee, complete e intuitive sui processi di lavorazione. Ciò ridurrà significativamente i tempi di esecuzione e gli errori associati alle operazioni di setting, attrezzaggio e manutenzione macchina.
- Possibilità di adottare nuovi modelli di business grazie all'introduzione di nuovi servizi. Biesse sarà in grado di monitorare lo stato effettivo di macchine e processi in tempo reale e di identificare efficacemente le cause di eventuali problemi. Ciò consentirà di potenziare i servizi che *Biesse Service* attualmente fornisce ai clienti da remoto.

La produzione italiana occupa una posizione di rilievo nel mercato mondiale grazie all'alta qualità dei prodotti e alla elevata performance generale. Ciò consente ad esse di applicare prezzi più elevati nei mercati di esportazione, sia all'interno che all'esterno dell'Europa. Gli sviluppi tecnologici creati nell'ambito di questo progetto mirano ad aumentare la competitività rafforzando il capitale intangibile e migliorando la qualità e l'efficienza dei processi. Oggigiorno nelle imprese l'utilizzo dei dati e della conoscenza è spesso parziale. Invece, questo progetto punta proprio alla realizzazione di un processo di valorizzazione di tutti i dati aziendali, sfruttando le moderne tecnologie basate sui Big Data e sulla Data Science.

Ricapitolando, l'obiettivo finale del progetto Intelligence 5.0 è quello di fare ricerca e sviluppare una nuova generazione di macchine sensorizzate che implementino un interno framework HW/SW per il rilevamento delle anomalie basato su modelli Digital Twin (DT) che combinano tecniche di Machine Learning e regole knowledge-based. Esso verrà utilizzato per fornire un rilevamento precoce dei guasti, capacità diagnostiche predittive e affidabilità delle apparecchiature, oltre a fornire servizi di supporto remoto per il monitoraggio e l'ottimizzazione dei processi. Queste macchine saranno, inoltre, dotate di SmartHMI in grado di migliorare l'usabilità e la sicurezza degli utilizzatori.

2.2.4 Obiettivi realizzativi del progetto

Il progetto è suddiviso, come è possibile vedere nella Tabella 2.1, in dieci obiettivi realizzativi che possono essere distinti sulla base del soggetto proponente (UNIVPM/BIESSE) e della tipologia, ovvero se appartengono alla categoria di *Ricerca Industriale* o di *Sviluppo Sperimentale*. In Figura 2.2 è possibile vedere gli output attesi per gli obiettivi realizzativi del progetto.

Di seguito viene presentata una breve descrizione per ogni obiettivo.

OR1 - Mappatura della conoscenza relativa alle macchine e ai processi e identificazione degli asset critici

Questo obiettivo punta alla realizzazione di una mappa del ciclo di vita delle macchine, tenendo conto dei processi produttivi, delle operazioni (come cicli, dinamiche, lavorazioni

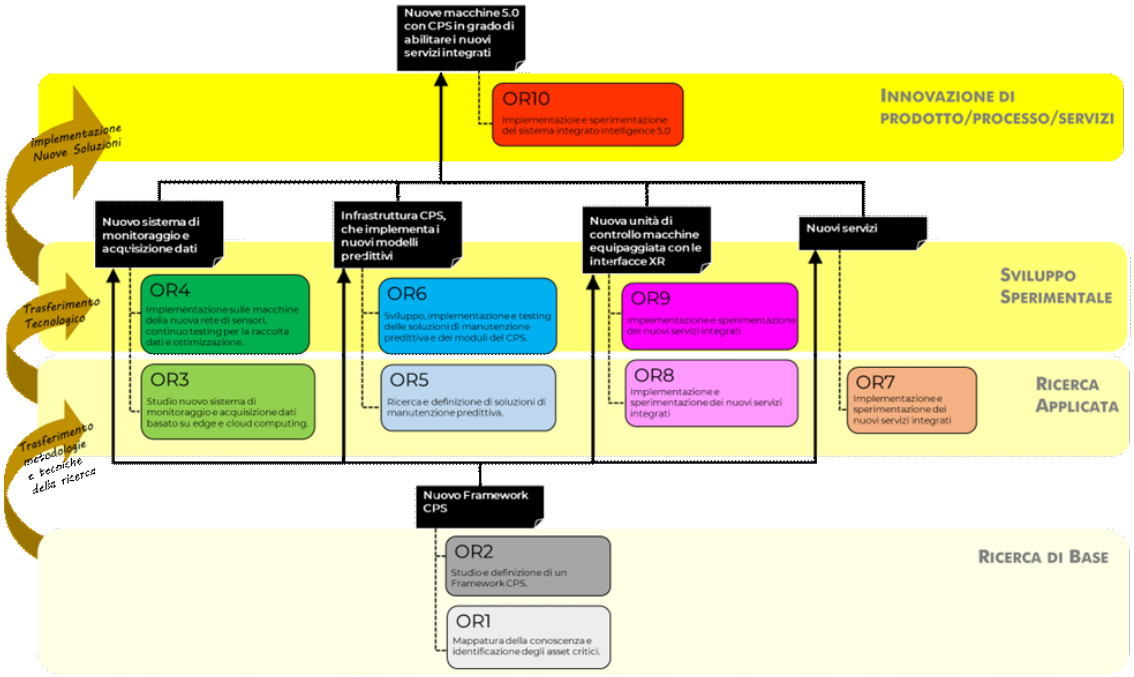


Figura 2.2: Output "Intelligence 5.0"

ecc.), delle loro componenti e del loro utilizzo. Questa conoscenza fornirà una struttura trasversale che faciliterà l'analisi dei guasti permettendone la loro previsione e identificando al contempo gli asset critici della macchina per il monitoraggio.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A1.1. Mappatura della conoscenza relativa alle macchine (singoli sistemi, componenti e sottosistemi), ai processi di produzione e alle lavorazioni. Identificazione delle informazioni scambiate e condivise tra i diversi reparti aziendali (ufficio tecnico, controllo qualità, service) nelle varie fasi del ciclo-vita di prodotto.
- A1.2. Identificazione dei componenti critici delle macchine.

OR2 – Studio e definizione di un Framework CPS

L'obiettivo consiste nel definire il Framework e nell'individuare l'architettura ottimale del Cyber Physical System (CPS). Esso richiede una definizione completa dell'intero ecosistema digitale, la modellazione della conoscenza multidominio necessaria per le simulazioni (macchine, processi di lavorazione e produzione), la selezione di tecnologie di supporto innovative e la formulazione di nuovi algoritmi per l'operatività del CPS.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A2.1. Identificazione dei domini applicativi di cyber-macchine e servizi integrati e definizione dei requisiti funzionali del CPS. Formalizzazione della conoscenza relativa a macchine e processi.
- A2.2. Definizione dell'architettura del CPS, benchmark e selezione delle tecnologie abilitanti il framework. Elaborazione di specifiche tecniche e requisiti di ogni singolo modulo dell'architettura e loro connessioni.

OR	Soggetto proponente	Tipologia obiettivo (SS/RI)	Titolo OR
OR1	BIESSE	RI	Mappatura della conoscenza relativa alle macchine e ai processi e identificazione degli asset critici.
OR2	UNIVPM	RI	Studio e definizione di un Framework CPS.
OR3	UNIVPM	RI	Studio e definizione di un sistema di monitoraggio e acquisizione dati basato su edge e cloud computing.
OR4	BIESSE	SS	Implementazione sulle macchine della nuova rete di sensori, testing continuo per la raccolta dati e ottimizzazione.
OR5	UNIVPM	RI	Ricerca e definizione di soluzioni di manutenzione predittiva.
OR6	BIESSE	SS	Sviluppo, implementazione e testing delle soluzioni di manutenzione predittiva e dei moduli del CPS.
OR7	BIESSE	RI	Ideazione di nuovi servizi integrati che sfruttano le potenzialità del sistema CPS oggetto del programma.
OR8	UNIVPM	RI	Studio e definizione di interfacce uomo-macchina adattative, multiutente e multi-piattaforma.
OR9	BIESSE	SS	Sviluppo e sperimentazione delle nuove interfacce utente.
OR10	BIESSE	SS	Implementazione e sperimentazione dei nuovi servizi integrati.

Tabella 2.1: Obiettivi realizzativi del progetto

OR3 - Studio e definizione di un sistema di monitoraggio e acquisizione dati basato su edge e cloud computing

L'obiettivo è sfruttare il potenziale sia dell'edge computing, che consente la diagnostica in tempo reale, sia del cloud computing, che memorizza, scambia ed elabora i dati e supera i limiti di memoria dei sistemi edge, per fornire informazioni sullo stato della macchina e sulla dinamica dell'intero processo.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A3.1. Ricerca di un'infrastruttura integrata di sensori "collaborativi" per l'acquisizione dei dati di funzionamento della macchina, dei componenti e sottosistemi e dell'intero processo.
- A3.2. Definizione dell'architettura nodo edge computing per il controllo dinamico di organi della macchina e degli algoritmi per la gestione degli allarmi. Definizione della struttura delle informazioni in output e delle specifiche di comunicazione con l'ambiente cloud.

OR4 - Implementazione sulle macchine della nuova rete di sensori, testing continuo per la raccolta dati e ottimizzazione

Questo obiettivo consiste nel far evolvere la macchina in funzione della rete dei sensori sviluppata in OR3. Tale processo riguarderà vari aspetti della macchina, a partire dall'installazione dei sensori fisici, alla loro connessione e interconnessione, all'introduzione del nodo edge e dei sensori virtuali, nonché alla revisione al codice PLC per ridefinire e ottimizzare il flusso di dati da estrarre ed elaborare alla luce dell'introduzione del nodo edge. Verrà quindi effettuata un'ampia fase di test per verificare la funzionalità e le prestazioni previste per la soluzione adottata.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A4.1. Progettazione dell'infrastruttura sensoristica e del nodo edge e loro integrazione in una nuova linea di macchine utensili per i tre settori di riferimento di Biesse.
- A4.2. Sperimentazione continua per verificare l'accuratezza delle misurazioni, la robustezza ed affidabilità del dato input/output, la latenza della comunicazione, la sincronizzazione tra variabili e parametri.

OR5 - Ricerca e definizione di soluzioni di manutenzione predittiva

L'obiettivo è identificare soluzioni per stimare la vita utile residua (RUL) degli asset critici delle macchine identificati durante l'OR1 per consentire la fornitura di servizi di manutenzione predittiva. Queste soluzioni implementeranno modelli predittivi basati su una strategia ibrida che utilizza sia approcci basati sui dati (come metodi statistici e algoritmi di machine learning), che modelli matematici basati sulla fisica. In questo modo, verrà sfruttata in modo combinato la conoscenza derivante dall'analisi dei big data e dall'asset di riferimento.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A5.1. Ricerca dello stato dell'arte dei modelli matematici e degli algoritmi di machine learning da applicare per l'elaborazione dei dati storici disponibili e delle informazioni provenienti dal sistema di acquisizione dati a bordo macchina ai fini della predizione del processo di usura.
- A5.2. Definizione dei modelli prognostici volti alla stima della RUL degli asset di interesse, al fine di abilitare servizi di manutenzione predittiva.

OR6 – Sviluppo, implementazione e testing delle soluzioni di manutenzione predittiva e dei moduli del CPS

L'obiettivo è quello di implementare i modelli e le strutture sviluppate in OR5. In questo senso, i modelli di machine learning che ricevono i dati dalla nuova rete di sensori saranno applicati alle macchine. Il processo di istruzione e sviluppo di tali modelli è continuo, poiché si basa sull'apprendimento, e i dati in ingresso aumentano nel tempo, incrementando l'accuratezza dei modelli predittivi.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A6.1. Implementazione dei modelli definiti in OR5 per il rilevamento e la predizione di anomalie, guasti e malfunzionamenti. Sviluppo dell'architettura CPS con logiche decentralizzate, implementazione della tecnologia di digital twin, completo sviluppo della cross-domain framework, elaborazione dati.

- A6.2. Prototipazione e testing dei moduli funzionali del sistema CPS per una linea di macchine e sperimentazione continua volta al miglioramento degli algoritmi per rendere i risultati delle elaborazioni affidabili e ridurre il rischio di costi connessi all'implementazione dei relativi servizi.

OR7 - Ideazione di nuovi servizi integrati che sfruttano le potenzialità del sistema CPS oggetto del programma

Questo obiettivo ha lo scopo di utilizzare in modo efficace le conoscenze e i dati acquisiti dalle attività degli OR precedenti; alla luce del lavoro svolto, e della creazione di un CPS operativo, si dovranno ipotizzare e studiare delle applicazioni adite a sfruttarne le potenzialità.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A7.1. Ricerca e individuazione di servizi integrati a supporto dell'ecosistema digitale: manutenzione predittiva, diagnostica, assistenza in remoto, progettazione macchine/componenti knowledge-based, controllo macchine/processi in remoto, ottimizzazione di processo.
- A7.2. Ideazione e definizione delle specifiche dei nuovi servizi in linea con il piano strategico aziendale.

OR8 – Studio e definizione di interfacce uomo-macchina adattative, multiutente e multipiattaforma

L'obiettivo consiste nell'esplorare e definire nuovi strumenti HMI collaborativi, multipiattaforma e multiutente per l'utilizzo e la diffusione delle conoscenze disponibili in diversi contesti applicativi (come, ad esempio, progettazione, produzione, manutenzione, sicurezza, logistica, etc.) per migliorare la qualità del lavoro e i livelli di competenza dei dipendenti (riduzione del lavoro mentale, aumento della sicurezza, apprendimento continuo). L'obiettivo è quello di rendere possibile l'utilizzo di servizi avanzati in vari contesti aziendali attraverso tali interfacce. Esse sarebbero, infatti, capaci di migliorare l'efficienza dei processi, di ottimizzare le prestazioni delle apparecchiature, di aumentare la produttività, di ridurre i rischi per l' operatore e di supportare le attività di pianificazione e gestione dei processi.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A8.1. Definizione dei requisiti funzionali delle interfacce utente finalizzate ad abilitare i servizi definiti in OR7. Benchmark delle tecnologie disponibili e identificazione delle specifiche dei dispositivi di interfaccia per la visualizzazione dei dati e per la fruizione, il controllo e la gestione delle informazioni adottate nei vari contesti d'uso. Definizione del concept delle interfacce secondo un approccio user-centered.
- A8.2. Studio e definizione di algoritmi di intelligenza artificiale per la gestione delle funzioni di adattamento dei contenuti e delle caratteristiche dell'interfaccia (ad esempio, funzionalità esposte, metodi di visualizzazione delle informazioni, modalità di iterazione), così da ottimizzare i servizi offerti, in base al contesto, al profilo dell'utente e alle funzionalità della macchina.

OR9 – Sviluppo e sperimentazione delle nuove interfacce utente

Questo obiettivo ha lo scopo di implementare le nuove funzionalità di interfaccia utente definite nell'OR8. Lo sviluppo di queste interfacce punta a migliorare la user experience

sotto vari aspetti; tramite delle istruzioni provenienti dalla macchina l'operatore potrà essere in grado di svolgere una sequenza di operazioni correlate all'utilizzo a tutto tondo della macchina stessa, dalla manutenzione alle lavorazioni fino alla formazione.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A9.1. Progettazione definitiva ed implementazione delle nuove interfacce utente per l'utilizzo dei servizi cloud insight.
- A9.2. Sperimentazione delle nuove interfacce utente per la valutazione della loro usabilità ed applicabilità.

OR10 - Implementazione e sperimentazione dei nuovi servizi integrati

L'obiettivo è attivare i servizi sviluppati e discussi in OR7 utilizzando le nuove interfacce utente sviluppate e implementate nei due obiettivi precedenti. Sulla base delle specifiche di progetto sviluppate in OR7, si passa all'implementazione e alla diffusione dei servizi integrati definiti in precedenza. A questo punto, devono essere definiti i test da effettuare per verificare il CPS, il framework e le capacità computazionali nell'edge e nel cloud.

L'obiettivo potrà essere raggiunto grazie allo svolgimento delle seguenti attività:

- A10.1. Implementazione di tutti i servizi integrati.
- A10.2. Sperimentazione dei servizi integrati con il supporto delle nuove interfacce utente e valutazione dell'intero framework, identificazione delle criticità e definizione degli interventi per il miglioramento dei sistemi.

2.2.5 OR5: Ricerca e definizione di soluzioni di manutenzione predittiva

Questo lavoro di tesi si concentra su una piccola parte del progetto generale Intelligence 5.0; in particolare è collocato all'interno dell'Obiettivo Realizzativo 5 e, pertanto, poniamo un focus su di esso. Come già detto in precedenza, per poter raggiungere tale obiettivo dovranno essere svolte due attività.

La prima attività dell'OR5 esaminerà e definirà soluzioni per determinare lo stato di salute degli asset critici operativi (componenti, organi, sistemi, etc.) in base al loro comportamento. In particolare, verranno sviluppati modelli di simulazione basati sulla modellazione matematica, sull'esperienza e sulle tecniche di elaborazione statistica. Inoltre, saranno studiati algoritmi di machine learning in grado di fornire previsioni basate sulle informazioni acquisite mediante prove di usura accelerata grazie all'OR3, e sull'elaborazione dei dati operativi storici delle macchine BIESSE collegate e formalizzate da OR1.

Per ogni asset critico, l'obiettivo è creare un modello di simulazione del processo di degrado/usura (digital twin), basato, da un lato, su modelli di degradazione dell'asset (tenendo conto delle proprietà del materiale e delle condizioni operative), e dall'altro su un approccio data-driven.

Le soluzioni ottimali da utilizzare verranno determinate in base al livello di conoscenza disponibile sui vari asset di interesse. In particolare, nel caso in cui risulti complesso modellare analiticamente il comportamento dell'asset di interesse, ma fosse disponibile un ampio set di dati di funzionamento delle macchine connesse, potranno essere definiti degli algoritmi basati su vari tipi di approcci di machine learning (supervisionati e non) per stimare modelli predittivi del guasto basati sull'analisi delle serie temporali (ad esempio, mediante modelli autoregressivi e a medie mobili). Per garantire la qualità del modello, si andranno ad individuare opportuni metodi di classificazione e dimensionality reduction (feature extraction, principal component analysis, linear discriminant analysis). I modelli di predizione così

individuati verranno testati e ottimizzati in modo iterativo, sfruttando i dati che verranno raccolti grazie alle successive attività di implementazione e testing che saranno intraprese grazie all'OR6.

Lo scopo della seconda attività è, invece, identificare soluzioni per prevedere il comportamento futuro degli asset di interesse, in modo da determinare la loro vita utile residua (RUL) e stabilire il momento più opportuno per effettuare la manutenzione o sostituire l'apparecchiatura.

L'obiettivo è quello di individuare una soluzione per determinare la finestra temporale per la sostituzione dei componenti di interesse. Verranno analizzati i domini di interesse al fine di identificare le variabili caratteristiche del problema. Sulla base delle variabili identificate, verranno definiti dei modelli matematici per stimare e prevedere la RUL e verranno definiti opportuni KPI per introdurre vincoli sia relativi al dominio che alla funzione obiettivo (massimizzazione del tempo di funzionamento, minimizzazione dei fermi macchina).

CAPITOLO 3

Struttura dei dati di partenza e attività di ETL

In questo capitolo vengono, innanzitutto, presentati la struttura dei dati dei file di log forniti dall'azienda Biesse e i dati di interesse da estrarre. Successivamente, nella seconda parte del capitolo, vengono descritte le attività di estrazione e di trasformazione dei dati utilizzando Logstash, per consentire, infine, una corretta indicizzazione su Elasticsearch.

3.1 Struttura dei dati di partenza

In questa sezione viene presentata la struttura dei dati di partenza messi a disposizione da Biesse nonché i dati di interesse per questo lavoro di tesi.

3.1.1 Descrizione funzionalità di logging

I dati messi a disposizione da Biesse per questo lavoro di tesi sono relativi alla registrazione di log provenienti da un *centro di lavoro CNC Rover* (un esempio di macchinario Rover viene mostrato in Figura 3.1). I macchinari CNC sono macchine utensili che utilizzano un *controllo numerico computerizzato* e sono impiegati in quasi tutti i settori della meccanica. I movimenti di queste macchine sono controllati da un dispositivo elettronico incorporato all'interno chiamato, appunto, controllo numerico. Questo tipo di macchina utensile è ideale per lavorazioni precise e molto lunghe, grazie all'utilizzo di specifici software che permettono la preimpostazione dei movimenti.

Figura 3.1: Centro di lavoro a controllo numerico ROVER A Biesse

I macchinari CNC sono utilizzati per rimuovere materiale e creare diversi tipi di oggetti; il processo inizia con un blocco di materiale grezzo e, con l'ausilio di speciali utensili (ad esempio, la fresa), il materiale viene rimosso fino ad ottenere l'oggetto finito. Le macchine CNC sono estremamente diffuse nelle lavorazioni industriali, in quanto possono trattare un'ampia gamma di materiali, tra cui ferro, plastica e legno.

Un elemento molto importante dei centri di lavoro CNC è l'*elettromandrino* (Figura 3.2). L'elettromandrino è un dispositivo elettromeccanico, montato su macchine utensili, dotato di motore interno per generare la rotazione di un qualsiasi utensile installato ed eseguire la lavorazione prevista. La sua caratteristica principale è quella di aumentare la precisione e la produttività delle macchine utensili. Gli elettromandrini possono essere classificati in base alla loro applicazione e alla modalità di costruzione; si distinguono, infatti, elettromandrini per la fresatura, per la tornitura e per la rettifica.

Figura 3.2: Esempio di elettromandrino

Lo scopo del logging relativo ai centri di lavoro CNC prodotti da Biesse è quello di registrare, per ogni lavorazione:

- l'elettromandrino (*TP*) coinvolto;
- le informazioni riguardo l'*utensile* utilizzato;
- la *velocità* target ed effettiva;
- le *temperature* dello statore e dei cuscinetti;
- la *corrente* media e di picco;
- la *vibrazione* media e di picco;
- il *tempo* complessivo di lavorazione.

Ad ogni lavorazione è assegnato un ID (*idChiave*) e per essa si registrano alcune o tutte le precedenti grandezze in più istanti, secondo il criterio sottostante:

- *avvio* → si registrano il TP coinvolto, le informazioni riguardo l'utensile utilizzato, la velocità target ed effettiva e le temperature dello statore e dei cuscinetti;
- termine *rampa di accelerazione* → si registrano la velocità target ed effettiva, le temperature dello statore e dei cuscinetti, la corrente di picco e la vibrazione di picco;
- elettromandrino a *regime* e vibrazione stabile → si registrano la velocità target ed effettiva, le temperature dello statore e dei cuscinetti, la corrente di picco e la vibrazione di picco;
- con l'elettromandrino a regime, ad *intervalli regolari* → si registrano la velocità target ed effettiva, le temperature dello statore e dei cuscinetti, la corrente media e di picco e la vibrazione media e di picco;
- al termine della rampa di spegnimento → si registrano la velocità target ed effettiva, le temperature dello statore e dei cuscinetti, la corrente di picco e la vibrazione di picco.

3.1.2 Descrizione dei file di log

I file di log messi a disposizione da Biesse si riferiscono a due elettromandrini, ovvero *TP1* e *TP11*, che possono trovarsi in quattro diversi stati, compresi tra 0 e 3.

Nella Figura 3.3 viene mostrato un file di log; analizzando tale file è possibile visualizzare le informazioni delle lavorazioni dei macchinari.

In generale, ciascuna riga del file di log riporta un timestamp incrementale (valore che segue *PLC*), espresso in millisecondi trascorsi dalle ore 00:00 del 01/01/2010. Il valore presente tra parentesi quadre identifica l'istante in cui una certa lavorazione si trova in un determinato stato.

```
...
PLC : 388331204715.341 ## LogMonitorInverter(1): idChiave=436587 statoRot=0 [3762671] ##
PLC : 388331204715.343 ## LogMonitorInverter(1): TP11 numVibrom=2 [3762671] ##
PLC : 388331204715.345 ## LogMonitorInverter(1): UTE=FRESA300 (SROT=2) LUNG=97.849 DIAM=300.500 [mm] [3762671] ##
PLC : 388331204715.347 ## LogMonitorInverter(1): rpmSetPoint=6000.000 spindleRpm=0.000 [rpm] [3762671] ##
PLC : 388331204715.348 ## LogMonitorInverter(1): rpmSetPoint=6000.000 inverterRpm=0.000 [rpm] [3762671] ##
PLC : 388331204715.350 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=27.000 [°C] [3762671] ##
PLC : 388331204715.351 ## LogMonitorInverter(1): Time:  samples=0 in timeSamples=0.000 [sec] [3762671] ##
...
PLC : 388331375570.037 ## LogMonitorInverter(1): idChiave=436590 statoRot=0 [3779756] ##
PLC : 388331375570.039 ## LogMonitorInverter(1): TP11 numVibrom=2 [3779756] ##
PLC : 388331375570.041 ## LogMonitorInverter(1): UTE=FRE18DX (SROT=1) LUNG=166.000 DIAM=18.400 [mm] [3779756] ##
PLC : 388331375570.043 ## LogMonitorInverter(1): rpmSetPoint=19000.000 spindleRpm=0.000 [rpm] [3779756] ##
PLC : 388331375570.044 ## LogMonitorInverter(1): rpmSetPoint=19000.000 inverterRpm=0.000 [rpm] [3779756] ##
PLC : 388331375570.046 ## LogMonitorInverter(1): bearingsCelsius=26.000 statorCelsius=24.000 [°C] [3779756] ##
PLC : 388331375570.047 ## LogMonitorInverter(1): Time:  samples=0 in timeSamples=0.000 [sec] [3779756] ##
...
```

Figura 3.3: Esempio di file di log

La stringa *LogMonitorInverter(N)* indica che è stata attivata la struttura di monitoring collegata all'*inverter N* per l'accensione di un elettromandrino. Questa funzione è in output se viene rilevato un cambio di stato di rotazione dell'inverter.

All'accensione di un inverter (statoRot 0->1) vengono visualizzate alcune informazioni della lavorazione, nonché i valori istantanei delle grandezze monitorate. In particolare:

- *idChiave*: è la chiave progressiva del numero di lavorazione;
- *statoRot*: è lo stato di rotazione dello spindle, che si distingue in quattro valori:
  - 0: inverter fermo;
  - 1: inverter in rampa di accelerazione/decelerazione;
  - 2: inverter a regime;
  - 3: inverter in rampa di fermata per arresto;
- *numTP*: è il numero del TP interessato nella lavorazione;
- *numVibrom*: è il numero del vibrometro;
- *UTE*: è il nome dell'utensile installato sull'elettromandrino;
- *SROT*: è il verso di rotazione dell'utensile in base alla velocità comandata (destro se la rotazione è oraria vista dall'alto sul piano xy), che si distingue in quattro valori:
  - 0: indifferente;
  - 1: spindle destro, sottomandrino destro;
  - 2: spindle sinistro, sottomandrino sinistro;
  - 3: spindle sinistro, sottomandrino destro;
  - 4: spindle destro, sottomandrino sinistro;

- *LUNG*: è la lunghezza dell'utensile in rotazione;
- *DIAM*: è il diametro dell'utensile in rotazione;
- *rpmSetPoint*: è la velocità di rotazione comandata a regime per l'utensile;
- *spindleRpm*: è la lettura della velocità di rotazione istantanea (assoluta) dello spindle;
- *bearingsCelsius*: è la lettura della temperatura dei cuscinetti;
- *statorCelsius*: è la lettura della temperatura dello statore.

Nella Figura 3.4 vengono riportate le informazioni relative a *statoRot=1*.

```
...
PLC : 388331209535.495 ## LogMonitorInverter(1): idChiave=436587 statoRot=1 [3763153] ##
PLC : 388331209535.498 ## LogMonitorInverter(1): UTE=FRESA300 (SROT=2) LUNG=97.849 DIAM=300.500 [mm] [3763153] ##
PLC : 388331209535.499 ## LogMonitorInverter(1): rpmSetPoint=6000.000 spindleRpm=5972.000 [rpm] [3763153] ##
PLC : 388331209535.501 ## LogMonitorInverter(1): rpmSetPoint=6000.000 inverterRpm=5958.000 [rpm] [3763153] ##
PLC : 388331209535.502 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=24.000 [°C] [3763153] ##
PLC : 388331209535.503 ## LogMonitorInverter(1): CurrentPeak=32.623 [A] [3763153] ##
PLC : 388331209535.505 ## LogMonitorInverter(1): VibromPeak=0.500 [mm/sec] [3763153] ##
PLC : 388331209535.506 ## LogMonitorInverter(1): Time:  samples=482 in timeSamples=4.810 [sec] [3763153] ##
...
PLC : 388331380690.227 ## LogMonitorInverter(1): idChiave=436590 statoRot=1 [3780268] ##
PLC : 388331380690.230 ## LogMonitorInverter(1): UTE=FRE18DX (SROT=1) LUNG=166.000 DIAM=18.400 [mm] [3780268] ##
PLC : 388331380690.232 ## LogMonitorInverter(1): rpmSetPoint=19000.000 spindleRpm=19011.000 [rpm] [3780268] ##
PLC : 388331380690.233 ## LogMonitorInverter(1): rpmSetPoint=19000.000 inverterRpm=18852.000 [rpm] [3780268] ##
PLC : 388331380690.235 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=24.000 [°C] [3780268] ##
PLC : 388331380690.236 ## LogMonitorInverter(1): CurrentPeak=20.281 [A] [3780268] ##
PLC : 388331380690.237 ## LogMonitorInverter(1): VibromPeak=2.700 [mm/sec] [3780268] ##
PLC : 388331380690.238 ## LogMonitorInverter(1): Time:  samples=512 in timeSamples=5.110 [sec] [3780268] ##
...
```

Figura 3.4: Esempio di file di log per statoRot=1

Al regime dell'inverter (statoRot 1->2), come mostrato in Figura 3.5, vengono visualizzate alcune letture istantanee e i valori di picco registrati nel periodo di accelerazione precedente per raggiungere il setpoint (informazioni associate univocamente alla idChiave considerata). In particolare:

- *CurrentPeak*: rappresenta corrente di picco assorbita dall'elettromandrino;
- *VibromPeak*: indica la velocità di vibrazione efficace di picco.

```
...
PLC : 388331215725.644 ## LogMonitorInverter(1): idChiave=436587 statoRot=2 [3763772] ##
PLC : 388331215725.648 ## LogMonitorInverter(1): UTE=FRESA300 (SROT=2) LUNG=97.849 DIAM=300.500 [mm] [3763772] ##
PLC : 388331215725.649 ## LogMonitorInverter(1): rpmSetPoint=6000.000 spindleRpm=6002.000 [rpm] [3763772] ##
PLC : 388331215725.651 ## LogMonitorInverter(1): rpmSetPoint=6000.000 inverterRpm=6060.000 [rpm] [3763772] ##
PLC : 388331215725.653 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=25.000 [°C] [3763772] ##
PLC : 388331215725.654 ## LogMonitorInverter(1): CurrentPeak=7.225 CurrentMean=5.337 CurrentSigma=1.335 [A] [3763772] ##
PLC : 388331215725.656 ## LogMonitorInverter(1): VibromPeak=2.600 VibromMean=0.891 VibromSigma=0.626 [mm/sec] [3763772] ##
PLC : 388331215725.657 ## LogMonitorInverter(1): Time:  samples=380 in timeSamples=3.790 [sec] [3763772] ##
...
PLC : 388331392220.438 ## LogMonitorInverter(1): idChiave=436590 statoRot=2 [3781421] ##
PLC : 388331392220.441 ## LogMonitorInverter(1): UTE=FRE18DX (SROT=1) LUNG=166.000 DIAM=18.400 [mm] [3781421] ##
PLC : 388331392220.443 ## LogMonitorInverter(1): rpmSetPoint=19000.000 spindleRpm=19026.000 [rpm] [3781421] ##
PLC : 388331392220.445 ## LogMonitorInverter(1): rpmSetPoint=19000.000 inverterRpm=18984.000 [rpm] [3781421] ##
PLC : 388331392220.446 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=25.000 [°C] [3781421] ##
PLC : 388331392220.448 ## LogMonitorInverter(1): CurrentPeak=10.302 CurrentMean=4.455 CurrentSigma=1.440 [A] [3781421] ##
PLC : 388331392220.450 ## LogMonitorInverter(1): VibromPeak=1.800 VibromMean=1.622 VibromSigma=0.156 [mm/sec] [3781421] ##
PLC : 388331392220.451 ## LogMonitorInverter(1): Time:  samples=1000 in timeSamples=9.990 [sec] [3781421] ##
...
```

Figura 3.5: Esempio di file di log per statoRot=2

Al cambio di velocità (statoRot 2->1) o alla decelerazione per lo spegnimento (statoRot 2->3) vengono visualizzati, inoltre, i valori statistici registrati nel periodo di regime precedente durante la lavorazione sul pezzo. In particolare:

- *CurrentMean*: è la media campionaria delle misure di corrente;
- *CurrentSigma*: è la deviazione standard campionaria di corrente;
- *VibromMean*: è la media campionaria delle misure di vibrazione;
- *VibromSigma*: è la deviazione standard campionaria di vibrazione.

Anche allo spegnimento dell'inverter (statoRot 3->0) vengono visualizzati le medesime letture istantanee nonché i valori di picco registrati nel periodo precedente di decelerazione per l'arresto.

A fine lavorazione, infine, come mostrato in Figura 3.6, sarà visualizzabile un timestamp relativo alla durata dell'intera lavorazione idChiave (da ON ad OFF: statoRot 0->...->0).

```
...
PLC : 388331233976.205 ## LogMonitorInverter(1): idChiave=436587 totalTime=29.250 [sec] [3765597] ##
...
PLC : 388331417741.145 ## LogMonitorInverter(1): idChiave=436590 totalTime=42.160 [sec] [3783973] ##
...
```

Figura 3.6: Esempio di file di log fine lavorazione

3.1.3 Dati di interesse

Per questo lavoro di tesi Biesse ha proposto, come primo tema di ricerca, l'integrazione di una caratterizzazione dell'elettromandrino durante la rampa di accelerazione di quest'ultimo. Lo stato macchina che si riferisce a questa fase è *statoRot=1*.

I parametri ritenuti interessanti per questa analisi sono i seguenti: *VibromPeak*, *rpmSetPoint*, *spindleRpm*, *UTE*, *DIAM*, *LUNG*, *CurrentPeak*, *bearingsCelsius*, *statorCelsius*, *numTP*. I dati richiesti, tranne *numTP*, sono indicati nelle righe successive alla definizione dello *statoRot=1*, come si può vedere in Figura 3.7.

```
...
PLC : 388331209535.495 ## LogMonitorInverter(1): idChiave=436587 statoRot=1 [3763153] ##
PLC : 388331209535.498 ## LogMonitorInverter(1): UTE=FRESA300 (SROT=2) LUNG=97.849 DIAM=300.500 [mm] [3763153] ##
PLC : 388331209535.499 ## LogMonitorInverter(1): rpmSetPoint=6000.000 spindleRpm=5972.000 [rpm] [3763153] ##
PLC : 388331209535.501 ## LogMonitorInverter(1): rpmSetPoint=6000.000 inverterRpm=5958.000 [rpm] [3763153] ##
PLC : 388331209535.502 ## LogMonitorInverter(1): bearingsCelsius=25.000 statorCelsius=24.000 [°C] [3763153] ##
PLC : 388331209535.503 ## LogMonitorInverter(1): CurrentPeak=32.623 [A] [3763153] ##
PLC : 388331209535.505 ## LogMonitorInverter(1): VibromPeak=0.500 [mm/sec] [3763153] ##
PLC : 388331209535.506 ## LogMonitorInverter(1): Time:  samples=482 in timeSamples=4.810 [sec] [3763153] ##
...
```

Figura 3.7: I dati richiesti per l'analisi

Il numero dell'elettromandrino, invece, è indicato nella riga successiva alla definizione dello *statoRot=0* immediatamente precedente allo *statoRot=1* in analisi, come si può notare nella Figura 3.8. In particolare:

- nella riga con *statoRot=1* viene riportata la medesima chiave univoca di lavorazione a parità di elettromandrino (accensione, lavorazione e spegnimento elettromandrino);
- nella riga successiva con *statoRot=1* non viene più riportato il nome dell'elettromandrino; infatti; le informazioni sono legate dalla idChiave.

```
...
PLC : 388331204715.341 ## LogMonitorInverter(1): idChiave=436587 statoRot=0 [3762671] ##
PLC : 388331204715.343 ## LogMonitorInverter(1): TP11 numVibrom=2 [3762671] ##
...
```

Figura 3.8: Indicazione del numero dell'elettromandrino nei file di log

3.2 Attività di ETL tramite Logstash

In questo lavoro di tesi per effettuare le attività di ETL, è stato utilizzato Logstash, che permette di acquisire i dati, trasformarli e inviarli alla destinazione desiderata. È stato scelto Logstash poiché permette di importare e gestire in maniera efficiente dati non strutturati, come i file di log messi a disposizione da Biesse. Come già detto in precedenza, per poter acquisire e trasformare i dati, è necessario creare un file di configurazione che ha al suo interno tre importanti blocchi: "input", "filter" e "output". Lo scopo di questa attività è quello di estrarre i dati di interesse e di ottenere in output dei dati strutturati, in modo tale da avere, per ogni lavorazione, i relativi parametri nella fase di rampa di accelerazione dell'elettromandrino.

3.2.1 La sezione input del file .conf

La sezione di input del file di configurazione è costituita da diversi plugin che consentono l'importazione di diversi tipi di file. In questo caso è stato utile l'utilizzo del plugin "file", al fine di importare i dati dei file di log messi a disposizione da Biesse. Nel Listato 3.1 viene mostrato il codice realizzato per questo lavoro di tesi relativo alla sezione "input".

```
1 input {
2     file {
3         path => "inserire path"
4         start_position => "beginning"
5         sincedb_path => "NUL"
6         max_open_files => 100000
7     }
8 }
```

Listato 3.1: Plugin di input file

Plugin di input *file*

Il plugin di input "file" consente di importare un file di testo e di trasformarlo in un evento. Più precisamente, per impostazione predefinita, ogni riga del file da analizzare viene trattato come un evento separato. Grazie ai plugin all'interno della sezione "filter", è possibile unire più righe in un unico evento.

Tramite l'utilizzo di *path*, è possibile specificare il percorso dei file da importare, mentre *start_position* permette di scegliere da dove parte Logstash nell'elaborare i file, se all'inizio o alla fine. Come si può intuire, *max_open_files* consente di specificare il numero massimo di file che possono essere analizzati; poiché il valore di default è 4095 è stato necessario impostare un numero che consentisse di importare tutti i file di log fornitoci.

3.2.2 La sezione filter del file .conf

La sezione "filter" del file di configurazione consente di creare dei filtri che si posizionano al centro della pipeline tra input e output. Logstash permette di utilizzare vari plugin di filtro per gestire efficacemente gli eventi. Di seguito sono descritti i plugin utilizzati in questo lavoro di tesi, con i relativi codici scritti nel file Logstash.

Plugin di filtro *grok*

Il plugin di filtro "grok" consente di analizzare un testo arbitrario e di strutturarlo. "Grok" è, infatti, un ottimo modo per analizzare i dati di log non strutturati e convertirli in qualcosa di strutturato e interrogabile. Questo filtro utilizza le espressioni regolari, che consentono di mappare parti specifiche di ogni riga del file da analizzare in dei campi. Se non si riesce a trovare il modello di cui si ha bisogno, è possibile scrivere, come fatto in questo lavoro di tesi, un modello personalizzato. Nel Listato 3.2 viene mostrato il codice corrispondente.

```
filter {
    grok {
        match => {"message" => ["PLC : %{BASE16FLOAT:timestamp:float} ## LogMonitorInverter\([0-9]+\): idChiave=%
        {NUMBER:idChiave} statoRot=%{NUMBER:statoRot} \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): TP%{NUMBER:numTP} \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): TP%{NUMBER:numTP} numVibrom=[0-9]+ \[%
        {BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): UTE=%{GREEDYDATA:UTE} \(SROT=[0-9]+\) LUNG=%
        {BASE16FLOAT:LUNG:float} DIAM=%{BASE16FLOAT:DIAM:float} \[mm\] \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): AGGRE-numStMand-UTE=%{GREEDYDATA:UTE} \(SROT=[0-9]+\)
        LUNG=%{BASE16FLOAT:LUNG:float} DIAM=%{BASE16FLOAT:DIAM:float} \[mm\] \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): rpmSetPoint=%{BASE16FLOAT:rpmSetPoint:float}
        spindleRpm=%{BASE16FLOAT:spindleRpm:float} \[rpm\] \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): bearingsCelsius=%{BASE16FLOAT:bearingsCelsius:float}
        statorCelsius=%{BASE16FLOAT:statorCelsius:float} %{NOTSPACE} \[%{BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): CurrentPeak=%{BASE16FLOAT:CurrentPeak:float} \[A\] \[%
        {BASE10NUM:ist_acc:int}\] ##",
        "PLC : [0-9]+.[0-9]+ ## LogMonitorInverter\([0-9]+\): VibromPeak=%{BASE16FLOAT:VibromPeak:float} \[mm/sec\] \
        [%{BASE10NUM:ist_acc:int}\] ##"]}
    }

    if "_grokparsefailure" in [tags] {
        drop {}
    }
```

Listato 3.2: Plugin di filtro grok

Questa sezione del file Logstash permette di selezionare per ogni riga gli attributi di interesse per l'analisi, ovvero *VibromPeak*, *rpmSetPoint*, *spindleRpm*, *UTE*, *DIAM*, *LUNG*, *CurrentPeak*, *bearingsCelsius*, *statorCelsius* e *numTP*. Oltre a questi, è necessario estrarre anche il valore compreso tra parentesi quadre (*ist_acc*) per poter eseguire successivamente l'aggregazione di tutti i valori che si riferiscono alla stessa lavorazione in un'unica riga. Tale processo viene descritto nel prossimo plugin di filtro utilizzato.

Plugin di filtro *aggregate*

Lo scopo del filtro "aggregate" è quello di aggregare le informazioni disponibili tra diversi eventi (in genere righe di file di log) appartenenti a una stessa attività e, infine, di inserire le informazioni aggregate nell'evento dell'attività finale. Nel Listato 3.3 viene mostrato il codice corrispondente.

```
    aggregate {
       task_id => "%{ist_acc}"
       code => "
       map['timestamp'] ||= event.get('timestamp')
       map['ist_acc'] ||= event.get('ist_acc')
       map['idChiave'] ||= event.get('idChiave')
       map['statoRot'] ||= event.get('statoRot')
       map['numTP'] ||= event.get('numTP')
       map['UTE'] ||= event.get('UTE')
       map['LUNG'] ||= event.get('LUNG')
       map['DIAM'] ||= event.get('DIAM')
       map['rpmSetPoint'] ||= event.get('rpmSetPoint')
       map['spindleRpm'] ||= event.get('spindleRpm')
       map['bearingsCelsius'] ||= event.get('bearingsCelsius')
       map['statorCelsius'] ||= event.get('statorCelsius')
       map['CurrentPeak'] ||= event.get('CurrentPeak')
       map['VibromPeak'] ||= event.get('VibromPeak')"

       push_map_as_event_on_timeout => true
       timeout_tags => ['_aggregatetimeout']
       timeout => 10
    }

    if "_aggregatetimeout" not in [tags] {
        drop {}
    }

    if [statoRot] not in [0,1] {
        drop {}
    }
```

Listato 3.3: Plugin di filtro aggregate

La parte finale di questa sezione del file Logstash permette di eliminare tutte le righe relative agli stati dell'elettromandrino che non sono di interesse per l'analisi. Lo stato 0 è utile poiché ad esso è associata l'informazione sul nome dell'utensile installato sull'elettromandrino.

Plugin di filtro *mutate*

Il plugin di filtro "mutate" consente di eseguire delle modifiche sui dati. Ad esempio, tramite esso, è possibile effettuare la sostituzione, la creazione e la modifica dei campi. Nel Listato 3.4 viene mostrato il codice corrispondente.

```
    if [statoRot] == "0" {
        mutate {
        remove_field => ["timestamp"]
        remove_field => ["ist_acc"]
        remove_field => ["UTE"]
        remove_field => ["LUNG"]
        remove_field => ["DIAM"]
        remove_field => ["rpmSetPoint"]
        remove_field => ["spindleRpm"]
        remove_field => ["bearingsCelsius"]
        remove_field => ["statorCelsius"]
        remove_field => ["CurrentPeak"]
        remove_field => ["VibromPeak"]
        remove_field => ["statoRot"]
        }
    }

    aggregate {
    task_id => "%{idChiave}"
    code => "
    map['idChiave'] ||= event.get('idChiave')
    map['numTP'] ||= event.get('numTP')
    map['UTE'] ||= event.get('UTE')
    map['LUNG'] ||= event.get('LUNG')
    map['DIAM'] ||= event.get('DIAM')
    map['statoRot'] ||= event.get('statoRot')
    map['timestamp'] ||= []
    map['timestamp'] << {'timestamp' => event.get('timestamp')}
    map['ist_acc'] ||= []
    map['ist_acc'] << {'ist_acc' => event.get('ist_acc')}
    map['rpmSetPoint'] ||= []
    map['rpmSetPoint'] << {'rpmSetPoint' => event.get('rpmSetPoint')}
    map['spindleRpm'] ||= []
    map['spindleRpm'] << {'spindleRpm' => event.get('spindleRpm')}
    map['bearingsCelsius'] ||= []
    map['bearingsCelsius'] << {'bearingsCelsius' => event.get('bearingsCelsius')}
    map['statorCelsius'] ||= []
    map['statorCelsius'] << {'statorCelsius' => event.get('statorCelsius')}
    map['CurrentPeak'] ||= []
    map['CurrentPeak'] << {'CurrentPeak' => event.get('CurrentPeak')}map['VibromPeak'] ||= []
    map['VibromPeak'] << {'VibromPeak' => event.get('VibromPeak')}
    "

    push_map_as_event_on_timeout => true
    timeout_tags => ['_aggregatetimeout1']
    timeout => 10
    }

    if "_aggregatetimeout1" not in [tags] {
        drop {}
    }

    mutate {
        remove_field => [ "tags" ]
    }

    mutate {
        remove_field => "[timestamp][0]"
        remove_field => "[ist_acc][0]"
        remove_field => "[rpmSetPoint][0]"
        remove_field => "[spindleRpm][0]"
        remove_field => "[bearingsCelsius][0]"
        remove_field => "[statorCelsius][0]"
        remove_field => "[CurrentPeak][0]"
        remove_field => "[VibromPeak][0]"
    }
```

Listato 3.4: Plugin di filtro mutate

In questa sezione del file Logstash, il filtro "mutate" è stato utilizzato per eliminare tutte le informazioni relative allo stato 0, mantenendo soltanto l'informazione sul nome dell'utensile. Successivamente viene utilizzato nuovamente il filtro "aggregate" per aggregare questa informazione con tutti gli altri parametri di interesse per l'analisi. Ciò è possibile grazie ad *idChiave*, che identifica la lavorazione dell'elettromandrino.

Plugin di filtro *ruby*

Il plugin di filtro "ruby" viene utilizzato per eseguire degli script *Ruby* all'interno di Logstash. Ruby è un linguaggio di programmazione open source dinamico e orientato agli

oggetti. Il linguaggio che ha maggiormente ispirato il suo autore è *Smalltalk*, da cui Ruby ha tratto la maggior parte delle sue caratteristiche. Grazie alla sua semplicità e alla sua flessibilità esso è diventato uno dei linguaggi più popolari e utilizzati, soprattutto in ambito di sviluppo web.

Per ogni lavorazione dell'elettromandrino (identificata da *idChiave*) possono essere presenti più di uno `statoRot=1`. Per poter separare le informazioni di questi stati in due o più righe diverse, è stato necessario realizzare uno script in Ruby, che viene mostrato nel Listato 3.5.

```
    ruby {
     code => '
     f1 = event.get("ist_acc")
     f2 = event.get("rpmSetPoint")
     f3 = event.get("spindleRpm")
     f4 = event.get("bearingsCelsius")
     f5 = event.get("statorCelsius")
     f6 = event.get("CurrentPeak")
     f7 = event.get("VibromPeak")
     f8 = event.get("timestamp")

     if f1.is_a? Array and f2.is_a? Array and f3.is_a? Array and f4.is_a? Array and f5.is_a? Array and f6.is_a?
     Array and f7.is_a? Array and f8.is_a? Array and f1.length == f2.length and f1.length == f3.length and
     f1.length == f4.length and f1.length == f5.length and f1.length == f6.length and f1.length == f7.length and
     f1.length == f8.length
         a = []
         f1.each_index { |x|
             a << {
                 "ist_acc" => f1[x], "rpmSetPoint" => f2[x], "spindleRpm" => f3[x],
                 "bearingsCelsius" => f4[x], "statorCelsius" => f5[x], "CurrentPeak" => f6[x],
                 "VibromPeak" => f7[x], "timestamp" => f8[x]
             }
         }
         event.set("[@metadata][stuff]", a)
     end
     '
 }
 if [@metadata][stuff] {
     split { field => "[@metadata][stuff]" }
     mutate {
         replace => {
             "ist_acc" => "%{[@metadata][stuff][ist_acc][ist_acc]}"
             "rpmSetPoint" => "%{[@metadata][stuff][rpmSetPoint][rpmSetPoint]}"
             "spindleRpm" => "%{[@metadata][stuff][spindleRpm][spindleRpm]}"
             "bearingsCelsius" => "%{[@metadata][stuff][bearingsCelsius][bearingsCelsius]}"
             "statorCelsius" => "%{[@metadata][stuff][statorCelsius][statorCelsius]}"
             "CurrentPeak" => "%{[@metadata][stuff][CurrentPeak][CurrentPeak]}"
             "VibromPeak" => "%{[@metadata][stuff][VibromPeak][VibromPeak]}"
             "timestamp" => "%{[@metadata][stuff][timestamp][timestamp]}"
         }
     }
 }
```

Listato 3.5: Plugin di filtro ruby

Codice rimanente sezione *filter*

Nel Listato 3.6 viene mostrato il codice rimanente nella sezione *filter* del file Logstash. Le sue righe hanno lo scopo di convertire i valori degli attributi di interesse in `float` o `integer` nonché di eseguire delle operazioni per visualizzare l'informazione sul timestamp nel modo corretto.

```
 mutate {
     remove_field => [ "@version" ]
     remove_field => [ "@timestamp" ]
 }

 mutate {
     convert => ["timestamp","float"]
     convert => ["idChiave","integer"]
     convert => ["statoRot","integer"]
     convert => ["numTP","integer"]
     convert => ["ist_acc","integer"]
     convert => ["numTp","integer"]
     convert => ["rpmSetPoint","float"]
     convert => ["spindleRpm","float"]
     convert => ["bearingsCelsius","float"]
     convert => ["statorCelsius","float"]
     convert => ["CurrentPeak","float"]
     convert => ["VibromPeak","float"]
 }

 ruby {
    code => "event.set('timestamp', event.get('timestamp') + 1262304000000)"
```

```
    }

    ruby {
        code => "event.set('timestamp', event.get('timestamp') / 1000)"
    }

    date {
        match => [ "timestamp","UNIX" ]
    }

    mutate {
        remove_field => ["timestamp"]
    }
}
```

Listato 3.6: Codice rimanente sezione filter

3.2.3 La sezione output del file .conf

La sezione "output" del file di configurazione Logsatsh contiene diversi plugin che hanno il compito di inviare i dati degli eventi a una particolare destinazione.

Plugin di output *elasticsearch*

I risultati del lavoro di analisi dei log devono essere, infine, inviati a Elasticsearch utilizzando l'omonimo plugin. Gli eventi vengono, quindi, spostati in Elasticsearch, indicizzati e trattati come documenti. Si noti che l'opzione principale presente in questo plugin è *index*, che consente di fissare un nome per l'indice di Elasticsearch in cui saranno contenuti i documenti. Nel Listato 3.7 viene riportato il codice corrispondente.

```
output {
    elasticsearch {
        hosts => ["localhost:9200"]
        index => "logbiesse"
    }
}
```

Listato 3.7: Plugin di output elasticsearch

Nella Figura 3.9 si può vedere il risultato del plugin "elasticsearch" su Kibana.

Time	Document
Jun 7, 2022 @ 09:50:24	@timestamp: Jun 7, 2022 @ 09:50:24.573 bearingsCelsius: 24 CurrentPeak: 20.23 DIAM: 5 idChiave: 9,334 ist_acc: 254,782 LUNG: 100.041 numTP: 1 rpmSetPoint: 18,000 spindleRpm: 18,006 statorCelsius: 27 statoRot: 1 UTE: D5P1T VibromPeak: 2.9 _id: r2bsU4EBDTCIYr0X6E3K _index: logbiesse _score: - _type: _doc
Jun 7, 2022 @ 09:49:48.442	@timestamp: Jun 7, 2022 @ 09:49:48.442 bearingsCelsius: 24 CurrentPeak: 20.298 DIAM: 10 idChiave: 9,333 ist_acc: 251,169 LUNG: 122.454 numTP: 1 rpmSetPoint: 15,000 spindleRpm: 14,950 statorCelsius: 25 statoRot: 1 UTE: D10P1T VibromPeak: 2.1 _id: rmbsU4EBDTCIYr0X6E3K _index: logbiesse _score: - _type: _doc
Jun 7, 2022 @ 09:41:20.116	@timestamp: Jun 7, 2022 @ 09:41:20.116 bearingsCelsius: 25 CurrentPeak: 20.212 DIAM: 5 idChiave: 9,332 ist_acc: 200,337 LUNG: 100.041 numTP: 1 rpmSetPoint: 18,000 spindleRpm: 17,951 statorCelsius: 27 statoRot: 1 UTE: D5P1T VibromPeak: 2.6 _id: rWbsU4EBDTCIYr0X6E3K _index: logbiesse _score: - _type: _doc

Figura 3.9: Esempio di estrazione dei parametri su Kibana

CAPITOLO 4

Progettazione del framework

In questo capitolo viene descritta l'architettura del framework realizzato per l'analisi dei dati di interesse relativi agli elettromandrini installati sui macchinari di Biesse, che sono stati individuati nel capitolo precedente. Successivamente vengono descritti gli obiettivi di ciascun elemento del framework e come tali elementi interagiscono tra di loro.

4.1 Architettura del framework

Nel precedente capitolo si è discusso del primo tema di ricerca per questo lavoro di tesi, ossia l'integrazione di una caratterizzazione dell'elettromandrino durante la rampa di accelerazione di quest'ultimo. Dopo aver concluso questa attività è stato necessario proseguire con il lavoro.

Il secondo obiettivo è stato quello di eseguire analisi sui dati relativi agli elettromandrini installati sui macchinari di Biesse, al fine di consentire successivi sviluppi futuri che si dovranno concentrare sull'estrazione di una firma di guasto, non predittiva, di un componente macchina. L'obiettivo finale dell'azienda è, infatti, quello di individuare un'impostazione corretta di soglie sui parametri di interesse al fine di evitare guasti sulle macchine di sua produzione. Per arrivare a tutto ciò, in questo lavoro di tesi è stato realizzato un framework (Figura 4.1) che consente di effettuare un'analisi descrittiva preliminare. I componenti di tale framework sono tre, ovvero:

- *componente di estrazione e ingestion dei dati*: permette di estrarre i dati di interesse dai log dei macchinari forniti da Biesse e di eseguire l'ingestion in un altro sistema;
- *componente di visualizzazione dei dati*: permette di eseguire una prima visualizzazione generale rispetto ai parametri degli elettromandrini da analizzare;
- *componente di reportistica*: si interfaccia con il componente per la visualizzazione dei dati e permette di ottenere una reportistica che consente di eseguire analisi più approfondite.

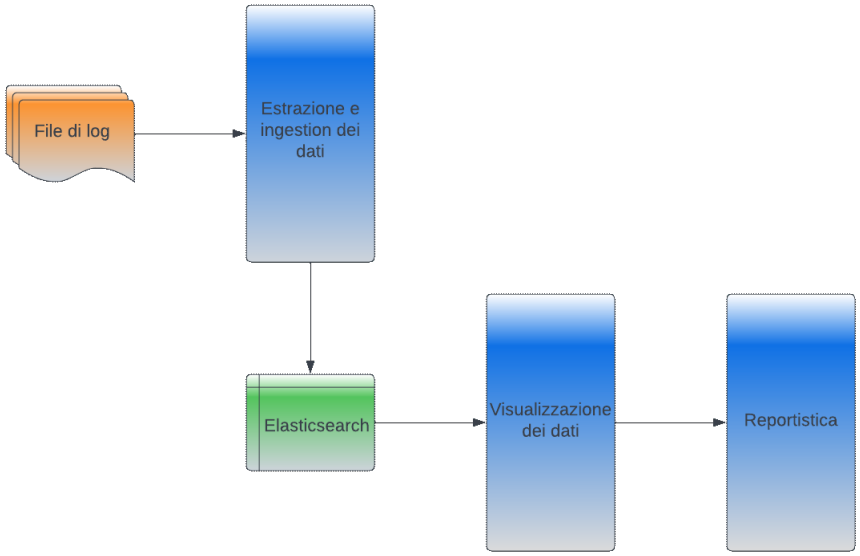


Figura 4.1: Architettura del framework

4.2 Componente estrazione e ingestion dei dati

L'ingestion dei dati è quel processo che consente lo spostamento e il trasferimento di diversi tipi di dati dalle loro fonti ad un altro sistema per l'elaborazione.

Il numero di dispositivi smart e IOT sta crescendo rapidamente; quindi il volume e il formato dei dati generati stanno, a loro volta, aumentando. Ciò è considerato un aspetto importante per l'importazione di big data, in quanto le aziende hanno bisogno di acquisire ed elaborare un ampio volume di dati a una velocità accettabile. Quando i dati provengono da fonti diverse quasi sicuramente hanno un formato e una struttura eterogenei; pertanto, c'è bisogno di una trasformazione in un formato comune che richiede, spesso, elaborazioni complicate e dispendiose in termini di tempo.

Per eseguire il processo di ingestion dei dati in modo efficace, si dovrebbe utilizzare lo strumento di acquisizione corretto e compatibile con l'applicazione di interesse. Dei parametri importanti per questa scelta sono sicuramente la dimensione dei dati e il loro formato, che può essere strutturato (ad esempio, tabulare), non strutturato (ad esempio, immagini) o semi-strutturato (ad esempio, file xml). Un altro aspetto da considerare è la frequenza dei dati; infatti, è possibile distinguere l'acquisizione in tempo reale da quella batch, nella quale i dati vengono archiviati in batch e trasferiti in un intervallo di tempo specifico. Infine, bisogna considerare anche la velocità del flusso di dati; l'obiettivo deve essere quello di rendere compatibile la pipeline di ingestion con il traffico dati aziendale.

In questo lavoro di tesi, per poter eseguire le analisi per il gruppo Biesse, è stato necessario l'utilizzo di un componente per l'estrazione e l'ingestion dei dati. Come già spiegato in precedenza, i dati di riferimento sono stati estrapolati da file di log forniti dall'azienda stessa. Pertanto, tramite Logstash, è stato possibile estrarre i dati e, successivamente, eseguire l'ingestion in un altro sistema. Questo sistema è Elasticsearch, il componente dello stack ELK dove vengono immagazzinati i dati e che si interfaccia facilmente con Logstash.

4.3 Componente visualizzazione dei dati

La visualizzazione dei dati è la rappresentazione grafica di informazioni e dati. Gli strumenti di visualizzazione dei dati forniscono soluzioni semplici per osservare e comprendere le tendenze, i valori anomali e le ricorrenze presenti nei dati utilizzando elementi visivi come grafici, diagrammi e mappe.

Con l'ingresso nell'era dei big data, la visualizzazione sta diventando uno strumento sempre più necessario per dare un senso alle miliardi di righe di dati generate ogni giorno. La visualizzazione dei dati aiuta a spiegare il loro contenuto organizzandoli in modo più chiaro, eliminando il superfluo e mettendo in primo piano le informazioni utili. Tuttavia, non è così semplice e immediato utilizzare un grafico in modo che risulti attrattivo, oppure presentare le informazioni in un'infografica. Una visualizzazione efficace dei dati richiede un sottile equilibrio tra forma e funzionalità. Un grafico molto semplice potrebbe essere poco piacevole per attirare l'attenzione o per rivelare efficacemente i punti chiave. D'altra parte, una visualizzazione troppo ricca potrebbe non riuscire a trasmettere efficacemente il messaggio oppure potrebbe essere inconcludente. Quindi i dati e le immagini devono essere combinati nella maniera ottimale.

È difficile immaginare una disciplina professionale che non possa trarre beneficio da una migliore comprensione dei dati. Tutte le aree della scienza, dell'ingegneria, della tecnologia e della matematica traggono vantaggio da una migliore comprensione dei dati. Lo stesso vale per il settore pubblico, la finanza, il marketing, la storia, i beni di consumo, i servizi, l'istruzione, lo sport, etc. Quindi esistono indubbiamente importanti applicazioni pratiche nel mondo reale, e quanto meglio gli argomenti vengono esposti in modo visivo, ad esempio attraverso delle dashboard, tanto più efficacemente si possono sfruttare le informazioni.

In questo lavoro di tesi, per poter raggiungere gli obiettivi appena descritti, è stato necessario l'utilizzo di un componente per la visualizzazione dei dati. Esso ha lo scopo di connettersi al sistema su Elasticsearch, ovvero dove sono memorizzati i dati di interesse, e restituire elementi visivi per una prima analisi descrittiva.

4.4 Componente reportistica

Un report è un documento concepito per trasmettere una serie di informazioni che sono state raccolte e analizzate preventivamente secondo determinati criteri. La reportistica (o reporting) dei dati costituisce la base di qualsiasi processo di analisi dei dati. Attraverso il reporting si evidenziano i fattori critici di successo e si identificano gli indicatori chiave, noti come *KPI* (Key Performance Indicator). Il loro obiettivo è quello di evidenziare le performance delle risorse, delle attività operative, nonché quelle dell'azienda nel suo complesso.

Se si ha a disposizione una quantità di dati elevata che proviene da diverse fonti, la tentazione potrebbe essere quella di creare un "collage" di informazioni che, seppur completo, potrebbe essere di difficile comprensione per gli utenti. Ciò che è fondamentale è evidenziare gli elementi più importanti omettendo quelli che lo sono meno o che non sono interessanti per l'analisi desiderata. Non c'è dubbio che i dati siano una risorsa preziosa per le aziende. Tuttavia, ciò dipende dalla facilità d'uso e dall'usabilità dei report.

Tra le varie tipologie di report che possono essere realizzati nelle aziende si possono distinguere principalmente i *report direzionali* e i *report operativi*. La differenza dipende essenzialmente dai destinatari del report. Per quanto riguarda i report direzionali, in essi sono importanti la completezza, la chiarezza e la sintesi, in quanto sono prodotti per i vertici del management. I report operatvi, invece, sono pensati per i manager intermedi: hanno, quindi, la necessità di essere più dettagliati, per poter gestire in modo più accurato i processi aziendali.

Importante è sottolineare la differenza tra report e dashboard, in quanto spesso viene fatta confusione tra i due termini. Il report è un documento statico, che contiene un livello di dettaglio maggiore e che presenta una precisa fotografia dei dati, spesso organizzati in forma testuale mediante tabelle. La dashboard è, invece, una visualizzazione dei dati di sintesi e di indicatori di facile lettura e comprensione.

In questo lavoro di tesi, oltre ai componenti di estrazione e ingestion dei dati e di visualizzazione, è stato necessario realizzare un componente per la reportistica. Esso si interfaccia con il componente per la visualizzazione e ha lo scopo di restituire dei report per poter eseguire analisi dei dati più dettagliate.

CAPITOLO 5

Implementazione del framework

In questo capitolo viene descritta l'implementazione del framework che è, come già detto in precedenza, suddiviso in tre componenti. L'implementazione del componente per l'estrazione e l'ingestion dei dati è stata già trattata nel Capitolo 3, in quanto è stato necessario presentare inizialmente come i dati dovessero essere preparati l'analisi. Nel primo paragrafo di questo capitolo viene descritta l'implementazione del componente per la visualizzazione dei dati, mentre nel secondo viene discussa l'implementazione del componente per la reportistica.

5.1 Implementazione della componente di visualizzazione dei dati

In questa sezione viene presentata l'implementazione della componente per la visualizzazione dei dati, interamente eseguita in Python. Prima di descrivere come può essere creata una dashboard in Python c'è, però, bisogno di effettuare un collegamento con Elasticsearch.

5.1.1 Collegamento con Elasticsearch

Per poter effettuare le analisi in Pyhton è necessaria l'estrazione dei dati da Elasticsearch. Essi saranno memorizzati all'interno della struttura dati *dataframe*, grazie all'utilizzo della libreria *pandas*. Nel Listato 5.1 è possibile vedere lo script realizzato in Python per estrarre i dati dall'indice denominato "logbiesse" presente su Elasticsearch. Dato che quest'ultimo limita il risultato della ricerca a 10.000 messaggi, è stato necessario, grazie a `max_result_window`, impostare un valore pari a 200.000 per rendere possibile l'estrazione di tutti i dati. Nelle righe successive del codice sono presenti delle istruzioni per poter prelevare effettivamente i dati e per poterli memorizzare in un dataframe.

```
1  from elasticsearch import Elasticsearch
2  import pandas as pd
3  import numpy as np
4
5  es.indices.put_settings(index = "logbiesse",
6                          body= {"index" : {
7                                  "max_result_window" : 200000}})
8
9  total_docs = 200000
10 response = es.search(
11     index = e_index,
12     size = total_docs
13 )
14
15 print(es.info())
16
17 elastic_docs = response["hits"]["hits"]
18 fields = {}
19
20 for num, doc in enumerate(elastic_docs):
21     source_data = doc["_source"]
```

```
22      for key, val in source_data.items():
23          try:
24              fields[key] = np.append(fields[key], val)
25          except KeyError:
26              fields[key] = np.array([val])
27
28  biesse_data = pd.DataFrame(fields)
29  print(biesse_data.info())
```

Listato 5.1: Estrazione dei dati da Elasticsearch

5.1.2 Dashboard per la visualizzazione dei dati

In questo lavoro di tesi, per la creazione di dashboard in Python, è stato utilizzato *Dash*. Questo è un framework open source per la creazione di interfacce di visualizzazione dei dati. Rilasciato nel 2017 come libreria Python, è cresciuto fino a includere implementazioni per *R* e *Julia*. Dash aiuta i data scientist e i data analyst a creare applicazioni web per l'analisi dei dati, senza richiedere conoscenze avanzate di sviluppo web. Tale framework è costruito basandosi su tre tecnologie, che sono:

- *Flash*: fornisce la funzionalità del server web;
- *React.js*: esegue il rendering dell'interfaccia utente della pagina Web;
- *Plotly.js*: genera i grafici utilizzati nell'applicazione.

Non è importante saper far funzionare tutte queste tecnologie insieme, in quanto è lo stesso framework a gestirle.

Dash è stato creato da *Plotly*, un'azienda canadese conosciuta soprattutto grazie alla popolare libreria che porta il suo nome. Plotly ha rilasciato Dash sotto licensa *MIT*; quindi è possibile utilizzarlo liberamente senza alcun costo. Al fine di realizzare una dashboard mediante l'utilizzo di questo framework, bisogna suddividere il processo in due passaggi:

- definire il layout della dashboard;
- utilizzare i *callback* per determinare quali parti dell'app sono interattive e a cosa reagiscono.

Nel Listato 5.2 è possibile vedere le prime righe dello script Pyhton da noi implementato per la realizzazione di dashboard.

```
1   import dash
2   import dash_core_components as dcc
3   import dash_html_components as html
4   from dash.dependencies import Output, Input
5
6   data = pd.read_csv ('biesse_data.csv')
7   data.rename(columns = {'@timestamp':'timestamp'}, inplace = True)
8   data.sort_values(by = ['timestamp'], inplace = True)
9   data["timestamp"] = data['timestamp'].astype("datetime64[ns,CET]")
10  data = data.dropna()
11
12  data1 = data[data["numTP"] == 1]
13  data11 = data[data["numTP"] == 11]
14
15  external_stylesheets = [
16      {
17          "href": "https://fonts.googleapis.com/css2?"
18          "family=Lato:wght@400;700&display=swap",
19          "rel": "stylesheet",
20      },
21  ]
22  app = dash.Dash(__name__, external_stylesheets=external_stylesheets)
23  app.title = "TP_Dashboards"
```

Listato 5.2: Importazione delle librerie e preparazione dei dati

Nelle righe da 1 a 3, viene effettuata l'importazione delle librerie richieste, che sono le seguenti:

- *dash*: consente di inizializzare l'applicazione;
- *dash_core_components*: consente di creare componenti interattivi, come grafici, menù a discesa o intervalli di date;
- *dash_html_components*: consente di accedere ai tag HTML.

Nella riga 4, invece, vengono importati gli oggetti `Input` e `Output`, che servono per poter gestire correttamente gli elementi grafici. Successivamente, nelle righe da 6 a 13, i dati vengono letti e preelaborati per poter essere utilizzati nella dashboard. Si può notare come, a partire da `data`, vengano creati due nuovi dataframe, ciascuno relativo ad uno specifico elettromandrino. Infatti, nei log che sono stati forniti da Biesse, sono presenti soltanto gli elettromandrini *TP1* e *TP11*. Nelle righe da 15 a 23 viene specificato un file CSS esterno, viene creata un'istanza della classe dashboard e ad essa viene assegnato un titolo.

A questo punto è necessario definire il layout della dashboard; nel Listato 5.3 è possibile vedere il codice relativo a questo passaggio.

```
app.layout = html.Div(
    children=[
        html.Div(
            children=[
                html.H1(
                    children="Dashboard_TP1-TP11", className="header-title"
                ),
                html.P(
                    children="",
                    className="header-description",
                ),
            ],
            className="header",
        ),
        html.Div(
            children=[
                html.Div(
                    children=[
                        html.Div(
                            children="Date_Range", className="menu-title"
                        ),
                        dcc.DatePickerRange(
                            id="date-range",
                            min_date_allowed=data.timestamp.min().date(),
                            max_date_allowed=data.timestamp.max().date(),
                            start_date=data.timestamp.min().date(),
                            end_date=data.timestamp.max().date(),
                        ),
                    ]
                ),
            ],
            className="menu",
        ),
        html.Div(
            children=[
                html.Div(
                    children=dcc.Graph(
                        id="VibromPeak", config={"displayModeBar": False},
                    ),
                    className="card",
                ),
                html.Div(
                    children=dcc.Graph(
                        id="CurrentPeak", config={"displayModeBar": False},
                    ),
                    className="card",
                ),
                html.Div(
                    children=dcc.Graph(
                        id="spindleRpm", config={"displayModeBar": False},
                    ),
                    className="card",
                ),
                ...
                ...
                ...
                ],
            className="wrapper",
        ),
    ]
)
```

Listato 5.3: Definizione del layout della dashboard

Questo codice determina il layout dell'applicazione utilizzando una struttura ad albero costituita da componenti Dash. Nelle righe da 1 a 13 è possibile vedere i componenti HTML

di Dash relativi alla sezione dell'header della pagina web. Si inizia definendo il componente padre (`html.Div`), per poi passare alla definizione dei suoi figli, ovvero un titolo (`html.H1`) e un paragrafo (`html.P`). Nelle righe successive vengono, invece, definiti altri elementi della dashboard, ovvero una sezione dove è possibile selezionare il range di date che si vuole considerare e una serie di *figure* che contengono grafici relativi ai diversi parametri degli elettromandrini. Essi sono dei grafici a linee che misurano nel tempo l'andamento dei parametri degli elettromandrini.

Definito tutto ciò, ora è necessario fare in modo che l'applicazione reagisca alle interazioni con l'utente. A tal fine, è necessario utilizzare le *funzioni di callback*. In Dash, quando un input cambia, viene attivata una funzione di callback. La funzione esegue alcune operazioni predeterminate, come il filtraggio di un set di dati, e restituisce un output all'applicazione. In sostanza, i callback collegano input e output nell'app. Nel Listato 5.4 è possibile vedere la funzione di callback utilizzata per aggiornare i grafici.

```
1  @app.callback(
2      [
3          Output("VibromPeak", "figure"),
4          Output("CurrentPeak", "figure"),
5          Output("spindleRpm", "figure"),
6          Output("rpmSetPoint", "figure"),
7          Output("statorCelsius", "figure"),
8          Output("bearingsCelsius", "figure"),
9          Output("UTE", "figure"),
10         Output("LUNG", "figure"),
11         Output("DIAM", "figure"),
12     ],
13     [
14         Input("date-range", "start_date"),
15         Input("date-range", "end_date"),
16     ],
17 )
18 def update_charts(start_date, end_date):
19     mask = (
20         (data.timestamp >= start_date)
21         & (data.timestamp <= end_date)
22     )
23     filtered_data = data1.loc[mask, :]
24     filtered_data11 = data11.loc[mask, :]
25     CurrentPeak_figure = {
26         "data": [
27             {
28                 "x": filtered_data["timestamp"],
29                 "y": filtered_data["CurrentPeak"],
30                 "type": "lines",
31                 "name":"TP1"
32             },
33             {
34                 "x": filtered_data11["timestamp"],
35                 "y": filtered_data11["CurrentPeak"],
36                 "type": "lines",
37                 "name": "TP11"
38             }
39         ],
40         "layout": {
41             "title": {
42                 "text": "CurrentPeak",
43                 "xanchor": "left",
44             },
45             "xaxis": {"fixedrange": True},
46             "yaxis": {"fixedrange": True},
47             "colorway": ["#17B897", "#fabd05"],
48         },
49     }
50     ...
51     ...
52     ...
53     return VibromPeak_figure, CurrentPeak_figure, spindleRpm_figure, rpmSetPoint_figure, statorCelsius_figure, \
54         bearingsCelsius_figure, LUNG_figure, UTE_figure, DIAM_figure
```

Listato 5.4: Utilizzo di callback per l'interazione con l'utente

Nelle righe da 1 a 17 vengono definiti gli input e gli output all'interno del decoratore `app.callback`. Innanzitutto, si definiscono gli output utilizzando gli oggetti `Output`. Essi accettano due argomenti, che sono:

- identificatore dell'elemento che verrà modificato durante l'esecuzione della funzione;
- proprietà dell'elemento da modificare.

Successivamente, si definiscono gli input utilizzando gli oggetti `Input`. Anch'essi accettano due argomenti, che sono:

- identificatore dell'elemento utilizzato per la modifica;
- proprietà dell'elemento osservato da fornire in input quando si verifica una modifica.

Nella riga 18 viene definita la funzione che viene applicata quando il range di date cambia. Si può notare che gli argomenti della funzione corrispondono all'ordine degli oggetti forniti alla callback. Infine, dalla riga 19 in poi, viene definito il corpo della funzione. In particolare, vengono definiti i grafici a linee per misurare l'andamento dei parametri degli elettromandrini. È possibile vedere come, nelle righe che vanno da 25 a 39, siano state definite due linee per il grafico, una per ogni elettromandrino. Invece, nella sezione *layout* (righe 40-47), è possibile selezionare l'asse (x o y), il colore delle linee e il formato del titolo del grafico. Tutte queste azioni sono state replicate nei grafici degli altri parametri che sono presenti nelle righe successive che, però, sono state omesse nel Listato 5.4 in quanto il codice è ripetitivo.

5.2 Implementazione della componente di reportistica

In questa sezione viene presentata l'implementazione della componente per la reportistica, anch'essa realizzata interamente in Python. In particolare, questa componente produce due report, uno intitolato *report analisi per giorno* e l'altro *report analisi TP11*, entrambi in Excel.

5.2.1 Report analisi per giorno

Come già indicato in precedenza, all'interno dei file di log messi a disposizione per questo lavoro di tesi sono presenti due elettromandrini; lo scopo del *report analisi per giorno* è quello di mostrare l'andamento dei valori di *massimo, minimo, media* e *mediana* dei parametri di interesse aggregati per giorno. Nei grafici, inoltre, è stata inserita anche un'informazione relativa ad un esempio di guasto. L'azienda Biesse, infatti, ha fornito un'indicazione di un esempio di guasto, ovvero di un problema di rottura di cono HSK sull'elettromandino TP11 con utensile FRESA_200_R260 (VibromPeak=190.600 [mm/sec]), avvenuto il 31/03/2022.

Per poter produrre i report in Excel è stato utilizzato *XlsxWriter*. Questo è modulo Python concesso in licenza BSD; il suo codice sorgente è interamente disponibile su Github. Esso può essere utilizzato per scrivere testo, numeri e formule in più fogli di lavoro e supporta funzionalità come formattazione, inserimento di immagini e di grafici, impostazione della pagina, filtri automatici, formattazione condizionale, etc. XlsxWriter presenta vari vantaggi rispetto ai moduli Python alternativi per la scrittura di file Excel; in particolare:

- supporta più funzionalità di Excel rispetto a qualsiasi modulo alternativo;
- nella maggior parte dei casi i file prodotti sono equivalenti al 100% ai file prodotti da Excel;
- ha un'ampia documentazione, file di esempio e test;
- è veloce e può essere configurato per utilizzare pochissima memoria anche per file di output molto grandi.

Lo svantaggio è che, però, non è in grado di leggere o modificare i file XLSX excel esistenti.

Il Listato 5.5 riporta il codice Python realizzato per "plottare" i valori di massimo, minimo, media e mediana aggregati per giorno per un solo parametro di interesse.

```
df_tp1 = biesse_data[biesse_data["numTP"] == 1]
df_tp11 = biesse_data[biesse_data["numTP"] == 11]

tp1_max = df_tp1.resample('D', on='@timestamp').statorCelsius.max()
tp11_max = df_tp11.resample('D', on='@timestamp').statorCelsius.max()
tp1_min = df_tp1.resample('D', on='@timestamp').statorCelsius.min()
tp11_min = df_tp11.resample('D', on='@timestamp').statorCelsius.min()
tp1_median = df_tp1.resample('D', on='@timestamp').statorCelsius.median()
tp11_median = df_tp11.resample('D', on='@timestamp').statorCelsius.median()
tp1_avg = df_tp1.resample('D', on='@timestamp').statorCelsius.mean()
tp11_avg = df_tp11.resample('D', on='@timestamp').statorCelsius.mean()

# plotting the points
plt.figure(figsize=(30,7))
plt.subplot(1, 4, 1)
plt.plot(tp1_max, label = "TP1", marker="o")
plt.plot(tp11_max, label = "TP11", marker="o")
plt.annotate('31 marzo',
             (tp11_max.index[91], tp11_max[91]),
             xytext=(10, 10),
             textcoords='offset_points',
             arrowprops=dict(arrowstyle='->'))
plt.legend()
plt.title("statorCelsius_max")

plt.subplot(1, 4, 2)
plt.plot(tp1_min, label = "TP1", marker="o")
plt.plot(tp11_min, label = "TP11", marker="o")
plt.annotate('31 marzo',
             (tp11_min.index[91], tp11_min[91]),
             xytext=(10, 10),
             textcoords='offset_points',
             arrowprops=dict(arrowstyle='->'))
plt.legend()
plt.title("statorCelsius_min")

plt.subplot(1, 4, 3)
plt.plot(tp1_avg, label = "TP1", marker="o")
plt.plot(tp11_avg, label = "TP11", marker="o")
plt.annotate('31 marzo',
             (tp11_avg.index[91], tp11_avg[91]),
             xytext=(10, 10),
             textcoords='offset_points',
             arrowprops=dict(arrowstyle='->'))
plt.legend()
plt.title("statorCelsius_avg")

plt.subplot(1, 4, 4)
plt.plot(tp1_median, label = "TP1", marker="o")
plt.plot(tp11_median, label = "TP11", marker="o")
plt.annotate('31 marzo',
             (tp11_median.index[91], tp11_median[91]),
             xytext=(10, 10),
             textcoords='offset points',
             arrowprops=dict(arrowstyle='->'))
plt.legend()
plt.title("statorCelsius_median")
```

Listato 5.5: Plot dei valori di massimo, minimo, media e mediana

Nella parte iniziale del codice si può notare come venga effettuato il ricampionamento dei dataframe relativi ai due elettromandrini sulla colonna *statorCelsius*. Lo scopo, infatti, è quello di aggregare i dati per giorno utilizzando `.max()`, `.min()`, `.median()` e `.mean()`.

Nelle righe successive vi è il codice realizzato, mediante la libreria *matplotlib*, per la visualizzazione dei grafici. Da notare come, attraverso `plt.annotate()`, è possibile evidenziare sui grafici il giorno in cui è stata registrata la rottura di un componente, ovvero il 31/03/2022 (91° giorno).

5.2.2 Report analisi TP11

Lo scopo del *report analisi TP11* è quello concentrare l'analisi soltanto su *TP11*, ovvero sull'elettromandrino in cui vi è un esempio di guasto. In questo report vengono considerati quattro periodi separati: la settimana prima del guasto, il giorno del guasto, la settimana dopo e il periodo totale disponibile. Tutto ciò viene fatto per cercare di comprendere la variazione dei parametri dell'elettromandrino nel periodo vicino al guasto e per poter effettuare un confronto con i valori che si riferiscono al periodo totale di logging.

Per ogni elettromandrino viene calcolato, in ogni periodo, il valore massimo, minimo, la media e la mediana e viene riportato un *boxplot* per visualizzare la distribuzione dei dati. Un boxplot (esempio in Figura 5.1) è una rappresentazione grafica utilizzata per descrivere la

distribuzione di un campione tramite semplici indici di dispersione e di posizione. I suoi elementi fondamentali sono:

- *Mediana dei dati*: è indicata dalla linea centrale della scatola. La metà dei dati si trova sopra questo valore, l'altra metà sotto.
- *Quartili*: sono indicati dalla parte inferiore e superiore della scatola. Ciascuno di essi esclude un quarto (25%) dei dati, uno verso l'alto e uno verso il basso. La lunghezza della scatola è la loro differenza ed è chiamata *range interquartile (IGR)*.
- *Baffi*: sono indicati dalle linee che si estendono dalla scatola. Essi si estendono per 1,5 volte dall'IQR nella parte superiore e inferiore della scatola.
- *Outlier*: sono i dati presenti sopra o sotto la fine dei baffi.
- *Massimo e minimo*: sono indicati dalle linee orizzontali alla fine dei baffi. Essi rappresentano il valore massimo e minimo presente nei dati senza considerare gli outlier.

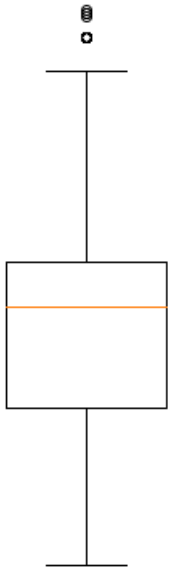


Figura 5.1: Esempio di boxplot

Nel Listato 5.6 è possibile vedere il codice relativo al calcolo degli outlier e di *upper_bound* e *lower_bound*, ovvero dei valori oltre ai quali i dati vengono considerati, appunto, degli outlier. All'interno del report tutti questi elementi vengono riportati per ogni parametro e in ogni periodo di riferimento.

```
arr1 = df_tp11[p]

# finding the 1st quartile
q1 = np.quantile(arr1, 0.25)

# finding the 3rd quartile
q3 = np.quantile(arr1, 0.75)
med = np.median(arr1)

# finding the iqr region
iqr = q3-q1

# finding upper and lower whiskers
upper_bound = q3+(1.5*iqr)
lower_bound = q1-(1.5*iqr)

# outliers

outliers = arr1[(arr1 <= lower_bound) | (arr1 >= upper_bound)]
```

Listato 5.6: Calcolo degli outlier

Per poter ragionare in modo completo, i boxplot e i valori di massimo, minimo, media e varianza vengono riportati nel report anche dopo aver rimosso gli outlier. Nel Listato 5.7 si può vedere il codice Python relativo alla creazione del boxplot eliminando i valori di outlier.

```
1 plt.figure(figsize=(12, 7))
2 arr2 = arr1[(arr1 >= lower_bound) & (arr1 <= upper_bound)]
3 plt.boxplot(arr2)
```

Listato 5.7: Boxplot senza outlier

CAPITOLO 6

Utilizzo del framework e knowledge pattern estratti

In questo capitolo vengono descritti l'utilizzo del framework e l'estrazione di knowledge pattern. In particolare, viene illustrato il modo con cui utilizzare le tre componenti del framework e di come è possibile, tramite la dashboard e i report prodotti, estrarre conoscenza utile dai risultati.

6.1 Utilizzo del framework

Per l'utilizzo del framework è necessario che i file di log siano memorizzati all'interno della macchina sulla quale viene eseguito il codice `.conf` di Logstash. Lo script ha lo scopo di estrarre soltanto i dati di interesse dalle migliaia di righe di log a disposizione, e di memorizzarli e indicizzarli in Elasticsearch.

Una volta indicizzati i dati in Elasticsearch è possibile utilizzare l'ultimo dei componenti dello stack ELK, ovvero Kibana. Esso viene utilizzato per visualizzare e accedere ai dati presenti su Elasticsearch. È possibile accedervi, tramite qualsiasi browser, immettendo come indirizzo `localhost:6501`.

Kibana è composto da diverse sezioni; quella più utilizzata è *Discover*. Essa è molto importante perché permette l'esplorazione interattiva dei dati. Tramite essa, è possibile inviare query, eseguire dei filtraggi e visualizzare i dati dai documenti. Se, all'interno dei dati del nostro indice, è presente un attributo legato al tempo, la distribuzione dei documenti (nel tempo) viene visualizzata tramite un istogramma nella parte superiore della pagina. Nella Figura 6.1 è possibile vedere la distribuzione dei documenti nel tempo estratti dai log dei macchinari di Biesse.

Successivamente, dopo aver memorizzato correttamente i dati ed eseguito una prima esplorazione, si deve utilizzare la componente per la visualizzazione dei dati. Tramite essa, ci si può collegare con l'indice creato in Elasticserch e avere a disposizione i dati in Python per poter creare una dashboard interattiva. Infine, la componente per la reportistica si collega con l'ultima appena descritta e produce due report in Excel.

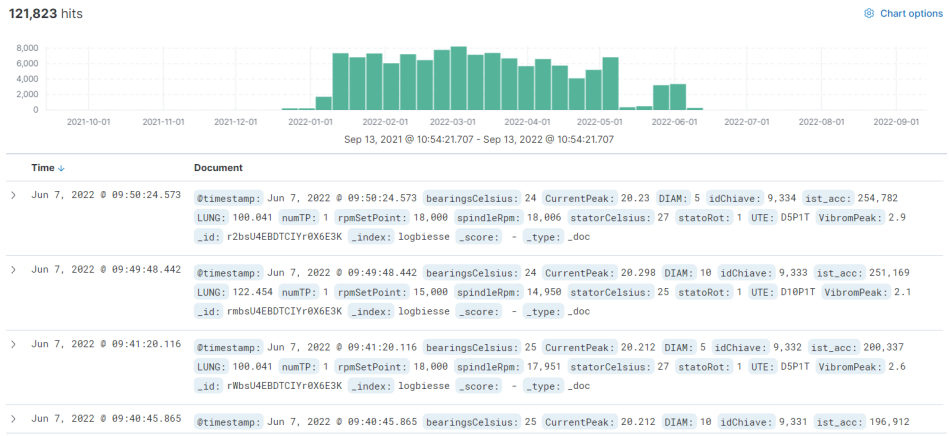


Figura 6.1: Sezione Discover di Kibana

6.2 Estrazione dei knowledge pattern

In questa sezione illustriamo alcuni degli strumenti da noi utilizzati per l'estrazione di conoscenza dai dati di Biesse.

6.2.1 Dashboard per la visualizzazione dei dati

La componente per la visualizzazione dei dati produce una dashboard che riporta informazioni riguardanti l'attributo temporale. Nello specifico, viene mostrato l'andamento nel tempo dei parametri di interesse, ovvero *VibromPeak*, *rpmSetPoint*, *spindleRpm*, *UTE*, *DIAM*, *LUNG*, *CurrentPeak*, *bearingsCelsius*, *statorCelsius*, per i due elettromandrini presenti nei log.

Nella dashboard è presente, come mostrato nella Figura 6.2, un filtro che permette di selezionare un giorno di inizio e di fine per poter esaminare l'andamento dei parametri in un periodo specifico.

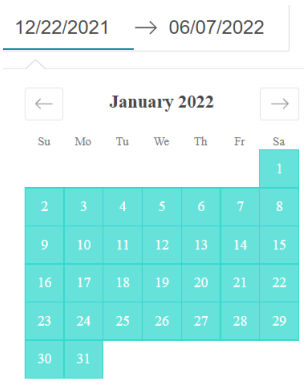


Figura 6.2: Filtro per selezionare il periodo di interesse

Nel grafico mostrato in Figura 6.3 è presente l'andamento di *VibromPeak*, che indica la velocità di vibrazione efficace di picco dell'elettromandrino. È possibile distinguere l'andamento degli elettromandrini dal diverso colore delle linee. Inoltre, se si passa il cursore sopra i punti del grafico, si ottiene l'informazione sul giorno e sul valore di VibromPeak.

Si può notare come ci siano dei valori molto più alti rispetto agli altri come, ad esempio, il valore di 190 mm/sec nel 31 marzo, ovvero nel giorno in cui si ha la rottura di un componente dell'elettromandrino TP11. Questa è, sicuramente, un'informazione importante che può essere esaminata dagli esperti di dominio. Un altro aspetto da considerare è che, nei giorni

precedenti al guasto, ci sono dei valori abbastanza alti nell'elettromandrino in cui si ha una rottura; questo potrebbe essere un segnale da studiare. Inoltre, nel grafico si nota un andamento molto simile nei due elettromandrini, con dei periodi in cui non sono presenti i valori, ossia nei weekend, quando si hanno dei fermi macchina.

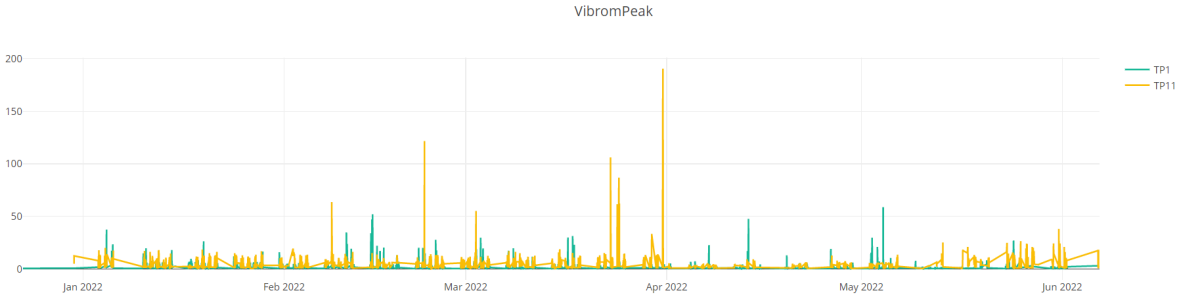


Figura 6.3: Andamento nel tempo di *VibromPeak*

Anche nei grafici relativi agli altri parametri si può osservare un andamento affine tra i due elettromandrini. In particolare, esaminiamo il grafico in Figura 6.4 che si riferisce a *spindleRpm*, che indica la velocità di rotazione istantanea.

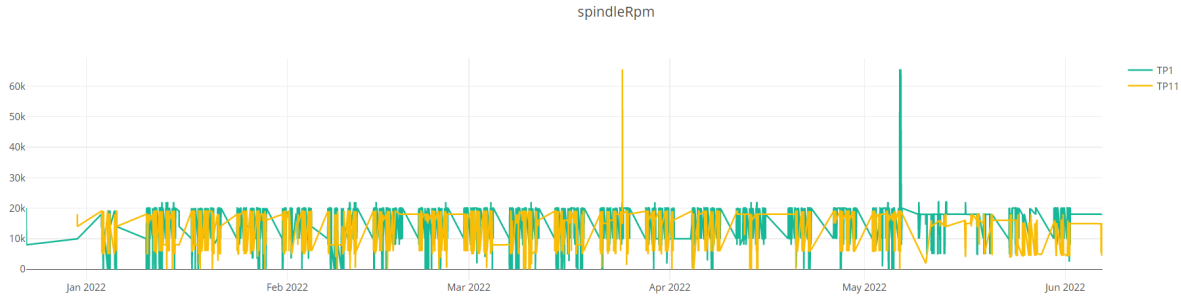


Figura 6.4: Andamento nel tempo di *spindleRpm*

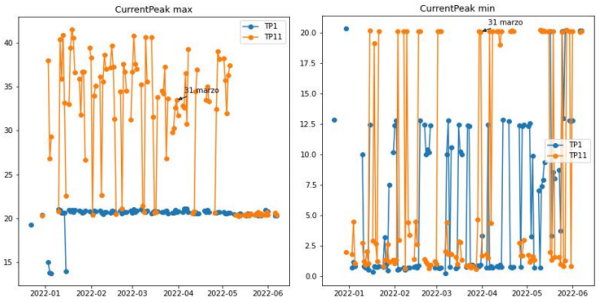
Anche in questo caso si nota un valore molto alto per TP11 nel periodo precedente al guasto, precisamente nel giorno 24 marzo. Inoltre, relativamente all'altro elettromandrino (TP1), sono presenti due valori anch'essi elevati rispetto al valore normale nel giorno 6 maggio.

6.2.2 Report analisi per giorno

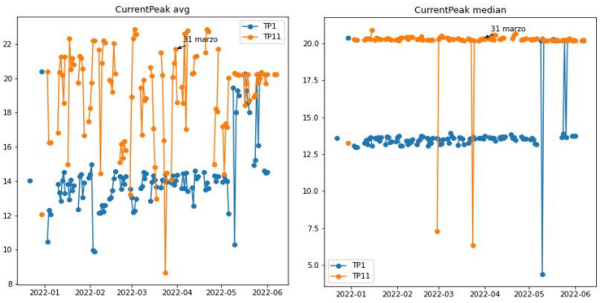
La componente per la reportistica produce due report; uno di questi è il report che esegue il plot dei valori di massimo, minimo, media e mediana dei parametri di interesse aggregati per giorno.

Un parametro sul quale è interessante concentrare l'analisi è *CurrentPeak*, che rappresenta la corrente di picco assorbita dall'elettromandrino (Figura 6.5).

Nel grafico *CurrentPeak max* si può notare come il valore massimo di *CurrentPeak* di TP1 varia pochissimo nel tempo, a differenza di TP11 in cui il range e l'andamento sono molto diversi, tranne che per l'ultimo mese. Interessante è anche il grafico sulla mediana, in cui i due andamenti sono molto simili a meno di alcuni minimi che andrebbero studiati con gli esperti di dominio. Per un periodo verso la fine delle osservazioni a disposizione, invece, i valori di TP1 crescono rispetto a quanto avveniva in precedenza; infatti, si può osservare che i valori minimi nel grafico *CurrentPeak min* aumentano rispetto al normale.



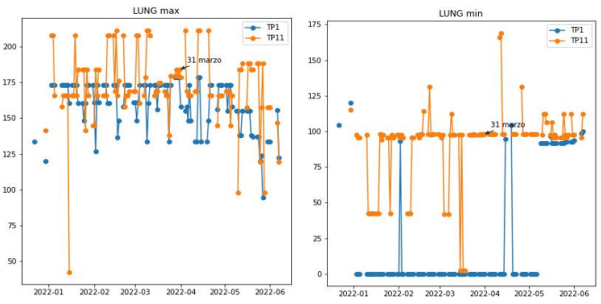
(a) *CurrentPeak max* **(b)** *CurrentPeak min*



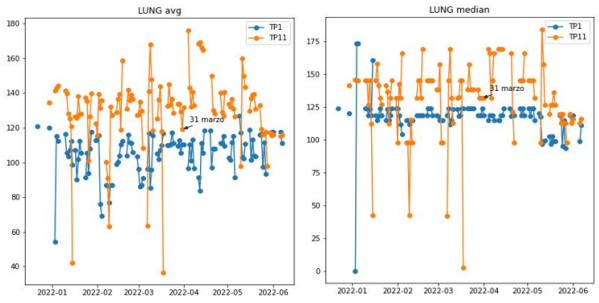
(c) *CurrentPeak avg* **(d)** *CurrentPeak median*

Figura 6.5: Andamento di *CurrentPeak*

Nelle Figure 6.6 e 6.7 vengono mostrati i grafici relativi a *LUNG* e *DIAM*, che indicano, rispettivamente, la lunghezza e il diametro degli utensili installati sugli elettromandrini. Dai pochi dati a disposizione non sembra esserci una correlazione tra il guasto e la dimensione dell'utensile. Questo però, utilizzando un numero maggiore di osservazioni relative ai guasti, potrebbe essere oggetto di studio in futuro, in quanto l'impatto che potrebbe avere la conformazione dell'utensile sui guasti potrebbe essere interessante da analizzare.

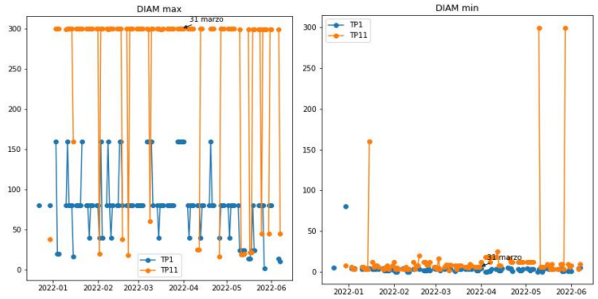


(a) *LUNG max* **(b)** *LUNG min*



(c) *LUNG avg* **(d)** *LUNG median*

Figura 6.6: Andamento della lunghezza dell'utensile



(a) *DIAM max* **(b)** *DIAM min*

(c) *DIAM avg* **(d)** *DIAM median*

Figura 6.7: Andamento del diametro dell'utensile

6.2.3 Report analisi TP11

Il secondo report prodotto dalla componente per la reportistica è quello relativo alla descrizione dei parametri dell'elettromandrino TP11. Nello specifico, esso riporta le informazioni relative a massimo, minimo, media e mediana in quattro periodi diversi, che sono la settimana prima del guasto, il giorno del guasto, la settimana dopo e il periodo totale disponibile. Come già detto in precedenza, questo lavoro viene eseguito per studiare la variazione dei parametri dell'elettromandrino nel periodo vicino al guasto e per poter effettuare un confronto con i valori che si riferiscono al periodo totale di logging. In aggiunta, vengono riportati anche dei boxplot per poter comprendere la distribuzione dei dati.

Nella Figura 6.8 viene mostrata una parte del report, ossia quella relativa al parametro *VibromPeak*. Nella Figura 6.9, invece, vengono mostrati i boxplot senza l'eliminazione degli outlier, relativi sempre al parametro *VibromPeak*.

VibromPeak

settimana	*prima*	*31-mar*	*dopo*	*all*
VibromPeak outliers	[10.5 10.7	[3.0 3.2 3.	[5.4]	[9.4 11.1 12.4 9.7 9.5 9.3 12.7 11.3 12.8 10. 9.8 10.2 10.5 17.799 14.3 13.9
upper_bound	10,05	2,75	5,35	8,9
lower_bound	-3,15	-0,05	-1,45	-3,1

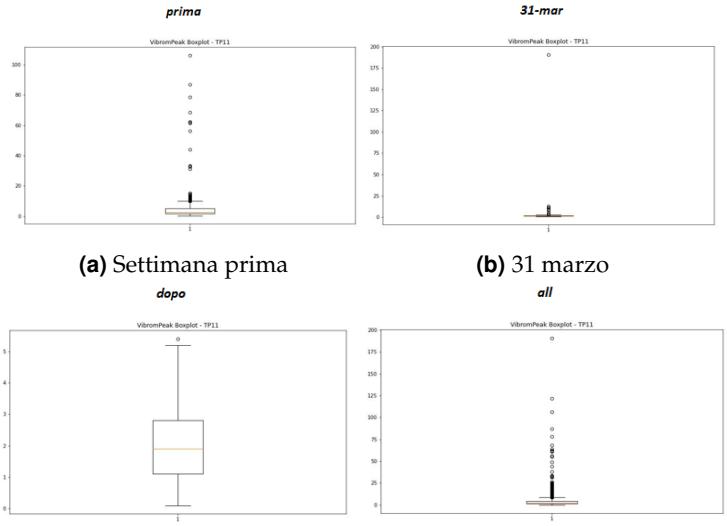
statistiche con outliers:

settimana		*prima*	*31-mar*	*dopo*	*all*
VibromPe	max	106,099	190,6	5,4	190,6
	min	0,3	0,8	0,1	0,1
	avg	4,14192	3,13139	2,00501	3,3227
	median	2,6	1,2	1,9	2,4

statistiche senza outliers:

settimana		*prima*	*31-mar*	*dopo*	*all*
VibromPe	max	10	2,6	5,2	8,9
	min	0,3	0,8	0,1	0,1
	avg	3,53025	1,33333	2,00186	2,87904
	median	2,55	1,1	1,9	2,3

Figura 6.8: Analisi sul parametro *VibromPeak*



(a) Settimana prima **(b)** 31 marzo

(c) Settimana dopo **(d)** Periodo totale

Figura 6.9: Boxplot relativi a *VibromPeak*

Osservando le informazioni ricavate è possibile notare come ci sia una variazione considerevole tra la settimana prima e la settimana dopo del guasto. In quella prima sono presenti numerosi valori di *VibromPeak* molto più alti rispetto al normale, cosa che non avviene, invece, per i giorni che seguono il guasto. Utilizzando i valori sopra la parte terminale dei baffi dei boxplot, vengono ricavati gli outlier. Effettuando questo calcolo, i valori molto grandi del parametro sono considerati proprio degli outlier; quindi, essi potrebbero essere oggetto di studio dagli esperti di dominio.

Nella Figura 6.10 viene mostrata la parte del report relativa al parametro *bearingsCelsius*, in cui vengono mostrate le sue statistiche.

bearingsCelsius

settimana		*prima*	*31-mar*	*dopo*	*all*
bearingsCelsius outl		[]	[29. 30. 31	[25. 30.]	[39. 25. 26. 27. 22. 30. 32. 33. 35. 36. 28. 38. 34. 31. 23. 29. 24. 37. 20.]
upper_bound		44,5	28,5	29,5	27
lower_bound		16,5	24,5	25,5	27

statistiche con outliers:

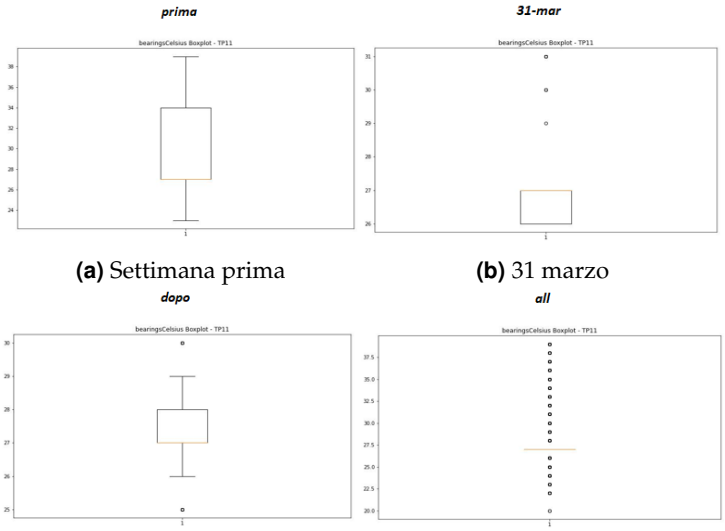
settimana		*prima*	*31-mar*	*dopo*	*all*
bearingsC	max	39	31	30	39
	min	23	26	25	20
	avg	30,2192	26,8175	27,1764	27,5307
	median	27	27	27	27

statistiche senza outliers:

settimana		*prima*	*31-mar*	*dopo*	*all*
bearingsC	max	39	27	29	27
	min	23	26	26	27
	avg	30,2192	26,5969	27,1748	27
	median	27	27	27	27

Figura 6.10: Analisi sul parametro *bearingsCelsius*

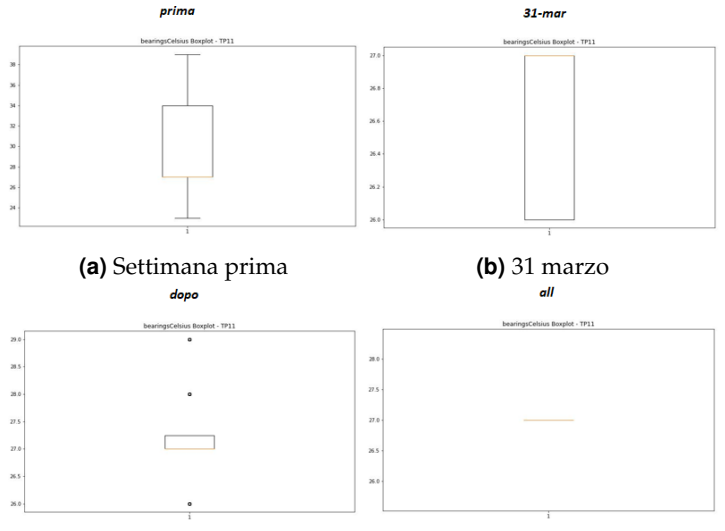
Nelle Figure 6.11 e 6.12 vengono mostrati, invece, i boxplot con e senza gli outlier relativi sempre al parametro *bearingsCelsius*.



(a) Settimana prima **(b)** 31 marzo

(c) Settimana dopo **(d)** Periodo totale

Figura 6.11: Boxplot con outlier relativi a *bearingsCelsius*



(a) Settimana prima **(b)** 31 marzo

(c) Settimana dopo **(d)** Periodo totale

Figura 6.12: Boxplot senza outlier relativi a *bearingsCelsius*

È interessante porre l'attenzione sui boxplot che considerano tutto il periodo di logging. Si osserva che, al loro interno vengono, considerati outlier tutti i punti di valore superiore e inferiore alla mediana. Questo avviene perché c'è un numero considerevole di punti di valore 27 relativi al parametro *bearingsCelsius*, che è proprio il valore della mediana.

CAPITOLO 7

Discussione

In questo capitolo vengono discussi i punti di forza e di debolezza del progetto e le lezioni apprese. Queste ultime sono di fondamentale importanza in quanto rappresenteranno informazioni preziose in fase di pianificazione delle future iniziative da avviare.

7.1 Punti di forza

Il progetto *Intelligence 5.0*, come già detto nei capitoli precedenti, punta a creare una rete di sensori collaborativi applicati ai componenti delle macchine a controllo numerico per attività di diagnostica predittiva. Il sistema, attraverso l'utilizzo di algoritmi di Machine Learning, fornirà analisi predittive su possibili guasti o malfunzionamenti dei macchinari, fornendo agli operatori un'interfaccia utente intelligente per un utilizzo ottimale dei macchinari.

Il suo obiettivo fondamentale è quello di sviluppare un framework in grado di utilizzare le conoscenze disponibili in modo integrato, consentendo la gestione dei dati e delle informazioni relative all'intero ciclo di vita di una macchina, dalla sua progettazione e realizzazione, passando per la fase di utilizzo, per arrivare alla fine del ciclo di vita.

La realizzazione di questo progetto consentirà di avere significativi miglioramenti tecnologici sia nei prodotti che nei processi. In termini di prodotti, le macchine attuali saranno migliorate e sarà introdotta una nuova generazione di macchine "intelligenti". Infatti, queste ultime passeranno da essere semplici strumenti di produzione a macchine facili da usare, capaci di comunicare con gli utenti. Inoltre, potranno essere utilizzate per migliorare i processi che le coinvolgono, fornendo utili indicatori agli addetti ai lavori. In particolare, l'implementazione di una fitta rete di sensori garantirà il continuo monitoraggio degli asset più critici della macchina in un'ottica di manutenzione predittiva, implementata grazie all'uso combinato dei nuovi algoritmi di Intelligenza Artificiale, Machine Learning e Digital Twin, che costituiranno lo stato dell'arte nell'ambito delle tecniche per i sistemi Industry 4.0.

Un altro punto di forza del progetto è che le macchine saranno in grado di trasferire grandi volumi di dati al cloud. Ciò consentirà di raccogliere ed elaborare grandi quantità di dati, aumentare la quantità di statistiche di feedback ricevute dalle diverse parti della macchina, generare output più affidabili sullo stato dei vari elementi e, quindi, fornire previsioni più accurate. Grazie a questo sistema la macchina permetterà lo svolgimento da remoto di numerose funzionalità; i tecnici per l'assistenza potranno avere una panoramica dettagliata delle condizioni della macchina ed essere capaci di intervenire in modo efficiente

e tempestivo senza la necessità di trasferte. Ciò porterà sicuramente dei grandi benefici per il cliente finale in termini di tempo di risoluzione del problema e costi di trasferta.

Quelli che finora sono stati descritti come i cambiamenti tecnologici apportati dalle innovazioni introdotte dal progetto porteranno a un elemento innovativo molto importante, ossia a una maggiore sostenibilità. Infatti, l'efficienza dei processi ridurrà il carico mentale degli operatori e migliorerà le condizioni di lavoro, oltre a ridurre i downtime. In questo modo, unitamente alla manutenzione programmata e predittiva, si potranno avere macchine molto più efficienti a parità di ciclo di vita rispetto a quelle odierne. Componenti e sistemi più affidabili avranno meno guasti non riparabili, con conseguente riduzione delle sostituzioni e del consumo dei materiali. L'efficienza dei processi operativi delle macchine porterà a una migliore pianificazione dei magazzini ottenendo una riduzione dei costi di gestione e logistica.

Infine, il gruppo Biesse prevede che le soluzioni sviluppate risulteranno di interesse anche per l'intera comunità scientifica; si può ragionevolmente prevedere che i risultati che verranno ottenuti grazie a questo progetto porteranno alla pubblicazione di vari articoli su riviste internazionali di prestigio. Il progetto, infatti, consentirà di apportare miglioramenti significativi allo stato dell'arte delle soluzioni di manutenzione predittiva.

7.2 Punti di debolezza

Come già detto in precedenza, questo lavoro di tesi è collocato all'interno di uno degli obiettivi realizzativi del progetto *Intelligence 5.0*. Tale obiettivo mira a definire soluzioni volte alla previsione del comportamento futuro degli asset di interesse, così da determinare la vita utile residua (RUL) e, di conseguenza, stabilire il momento più opportuno per eseguire la manutenzione delle apparecchiature o sostituirle.

Il primo tema di ricerca relativo all'obiettivo realizzativo appena menzionato ha lo scopo di trasferire le metodologie di scrittura dei log di Biesse e, contemporaneamente, estrarre una firma di guasto, non predittiva, di un componente macchina. Per portare a termine tali attività è necessaria sicuramente una quantità di dati maggiore rispetto a quella disponibile per questo lavoro di tesi. Non avendo a disposizione dati riportanti guasti o rotture di elettromandrini (ad eccezione di un solo punto temporale presente nel dataset in possesso), non è stato possibile soddisfare la richiesta di estrazione di firma di guasto.

Per poter valutare la qualità dei dati esistono delle metriche che aiutano a comprendere se essi possono essere sfruttati per condurre un'analisi affidabile basata sui dati stessi. Relativamente ai dati utilizzati in questo progetto i punti deboli sono la *completezza* e la *quantità*. La misura della completezza indica quanti dati mancano al set di dati per fornire una rappresentazione completa al 100% del contesto del mondo reale. Nel nostro caso mancavano le informazioni sui guasti dei macchinari, necessarie per poter eseguire un'analisi di maggior qualità. Invece, la misura di quantità indica quanto è appropriato il volume di dati posseduti in riferimento ad una determinata attività. Lavorare con più o meno dati del necessario può rivelarsi difficile da gestire. Nel nostro progetto avere una quantità di dati superiore avrebbe portato sicuramente a raggiungere risultati migliori.

Per quanto riguarda, invece, il punto sull'estrazione dei parametri di interesse è stato raggiunto un buon risultato in quanto è stato realizzato un sistema di data ingestion, già ampiamente discusso nei capitoli precedenti.

7.3 Lezioni apprese

In questa sezione vengono discusse le lezioni apprese grazie all'esecuzione del progetto; queste rappresentano le informazioni documentate che riflettono sia le esperienze positive che quelle negative di un progetto. Esse sono importanti soprattutto per il futuro, in quanto permettono di mettersi nelle condizioni di non ripetere eventuali errori commessi nel passato.

7.3.1 Qualità dei dati

Durante il periodo di lavoro per questo progetto è stata appresa una lezione fondamentale, ossia l'importanza di avere dati di alta qualità. La qualità dei dati è la misura di quanto questi ultimi siano adatti allo scopo per cui sono stati raccolti ed analizzati.

Grazie al rapido sviluppo dell'Information Technology, molti dei processi con cui abbiamo a che fare oggi sono stati digitalizzati. Da tanti anni ormai, si ci impegna a immagazzinare, trasferire, trasformare e, in generale, gestire i dati per supportare i sistemi IT affinché possano supportare il funzionamento delle organizzazioni. Bisogna, però, comprendere se si stia o meno sfruttando appieno il potenziale dei dati che si hanno in possesso.

I dati raccolti sono molto preziosi perché, se analizzati correttamente, possono aiutare a prendere decisioni importanti per le organizzazioni e per i clienti. Allo stesso tempo ci si sta sempre di più rendendo conto che per estrarre il massimo valore dai dati, è necessaria un'elevata qualità degli stessi. Dati incompleti, incoerenti, obsoleti o imprecisi possono fornire informazioni sbagliate ai decisori, portando a scelte poco oculate, che si traducono spesso in perdite economiche.

7.3.2 Scelta della tecnologia

Un'altra lezione appresa è l'importanza della scelta della tecnologia che deve essere utilizzata per il raggiungimento degli obiettivi prefissati. Quando si seleziona un tool, uno strumento o un software, quello che si vuole ottenere è rendere più efficienti e più semplici possibile le attività che devono essere svolte. In questo progetto sono stati utilizzati lo stack ELK e Python. Quest'ultimo, come sappiamo, è uno dei linguaggi più popolari in molti ambiti e, in particolare, nell'analisi dei dati. Per questo lavoro di tesi ha rappresentato, sicuramente un'ottima scelta, data la sua vasta gamma di soluzioni e di librerie adatte per i nostri obiettivi. Per la parte di estrazione dei parametri di interesse dai log di Biesse, anche Logstash si è dimostrato essere uno strumento eccellente. Infatti, tramite questo tool è stato possibile, con poche righe, eseguire le attività di nostro interesse in modo efficace. Le stesse attività, utilizzando un comune linguaggio di programmazione, avrebbero richiesto sicuramente una complessità di gran lunga superiore.

Conclusioni

Il progetto discusso nella presente tesi ha consentito di realizzare un framework per la gestione di guasti in un contesto Industry 4.0.

Inizialmente è stato descritto lo stack ELK, illustrando i suoi tre componenti, ovvero Elasticsearch, Logstash e Kibana. Si è proseguito con una breve introduzione a Python in cui sono stati esposti vari aspetti che lo caratterizzano, quali la sua storia, i motivi per cui utilizzarlo e i suoi principali campi di applicazione. Successivamente sono state presentati la realtà aziendale del gruppo Biesse e l'iniziativa "Intelligence 5.0", progetto innovativo che vuole migliorare vari aspetti dell'attuale mondo dell'Industria.

Dopo una prima parte introduttiva, il lavoro è stato quello di comprendere la struttura dei log forniti dall'azienda e, tramite l'utilizzo di Logstash, di estrapolare i dati di interesse memorizzandoli in un sistema documentale all'interno di Elasticsearch. Successivamente è stata descritta l'architettura del framework definito per analizzare i dati ottenuti. Dopo di ciò, si è visto come esso è stato implementato tramite Python. Il framework è composto da una componente che consente di estrarre i dati di interesse dai file di log e di eseguire l'ingestion, da una componente che permette di effettuare una visualizzazione generale rispetto ai parametri da analizzare e, infine, da un'ultima componente che effettua attività di reporting per consentire di eseguire analisi più approfondite. Lo scopo principale del lavoro svolto è quello di estrarre informazioni importanti relative ai macchinari di Biesse per permettere agli esperti di dominio di individuare eventuali malfunzionamenti.

L'ultima parte della tesi è dedicata alle lezioni apprese e ai punti di forza e di debolezza del lavoro svolto, in cui è emersa la fondamentale importanza dei dati per ottenere valore dall'analisi degli stessi.

Questo elaborato di tesi è utile per consentire successivi sviluppi futuri proposti dall'azienda. Essi si dovranno concentrare sull'estrazione di una firma di guasto, non predittiva, di un componente macchina. L'obiettivo finale del gruppo Biesse per questa attività è, infatti, quello di individuare un'impostazione corretta di soglie sui parametri di interesse al fine di evitare guasti sulle macchine di sua produzione.

Bibliografia

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

Jaber Alwidian, S Abdel Rahman, Maram Gnaim, and Fatima Al-Taharwah. Big data ingestion and preparation tools. *Modern Applied Science*, 14(9):12–27, 2020.

Manuela Aparicio and Carlos J Costa. Data visualization. *Communication design quarterly review*, 3(1):7–11, 2015.

Tarek Azzam, Stephanie Evergreen, Amy A Germuth, and Susan J Kistler. Data visualization and evaluation. *New Directions for Evaluation*, 2013(139):7–32, 2013.

Lisa Ehrlinger and Wolfram Wöß. A survey of data quality measurement and monitoring tools. *Frontiers in Big Data*, page 28, 2022.

Michael Frigge, David C Hoaglin, and Boris Iglewicz. Some implementations of the boxplot. *The American Statistician*, 43(1):50–54, 1989.

John D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

Maqbool Khan, Xiaotong Wu, Xiaolong Xu, and Wanchun Dou. Big data challenges and opportunities in the hype of industry 4.0. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & information systems engineering*, 6(4):239–242, 2014.

Andreea Mătăcuţă and Cătălina Popa. Big data analytics: Analysis of features and performance of big data ingestion tools. *Informatica Economica*, 22(2), 2018.

Wes Mckinney. pandas: a foundational python library for data analysis and statistics. *Python High Performance Science Computer*, 01 2011.

John McNamara. Creating excel files with python and xlsxwriter, 2021.

Fatemeh Nargesian, Erkang Zhu, Renée J Miller, Ken Q Pu, and Patricia C Arocena. Data lake management: challenges and opportunities. *Proceedings of the VLDB Endowment*, 12 (12):1986–1989, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12 (null):2825–2830, nov 2011. ISSN 1532-4435.

Sebastian Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *Journal of open source software*, 3(24):638, 2018.

Matthew Sadiku, Adebowale E Shadare, Sarhan M Musa, Cajetan M Akujuobi, and Roy Perry. Data visualization. *International Journal of Engineering Research And Advanced Technology (IJERAT)*, 2(12):11–16, 2016.

Sören Sonnenburg, Gunnar Rätsch, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio de Bona, Alexander Binder, Christian Gehl, and Vojtěch Franc. The shogun machine learning toolbox. *The Journal of Machine Learning Research*, 11:1799–1802, 2010.

I. Stančin and A. Jović. An overview and comparison of free python libraries for data mining and big data analysis. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 977–982, 2019. doi: 10.23919/MIPRO.2019.8757088.

Ikbal Taleb, Mohamed Adel Serhani, and Rachida Dssouli. Big data quality: A survey. In *2018 IEEE International Congress on Big Data (BigData Congress)*, pages 166–173. IEEE, 2018.

Li Da Xu, Eric L Xu, and Ling Li. Industry 4.0: state of the art and future trends. *International journal of production research*, 56(8):2941–2962, 2018.

Siti web consultati

- Amazon – `www.aws.amazon.com`
- ELK stack – `www.elastic.co`
- TIOBE – `www.tiobe.com`
- python-course.eu – `www.python-course.eu`
- html.it – `www.html.it`
- KDnuggets – `www.kdnuggets.com`
- Plotly Technologies Inc. – `www.plot.ly`
- seaborn – `www.seaborn.pydata.org`
- Dash – `www.plot.ly/products/dash`
- Keras – `www.keras.io`
- PyTorch – `www.pytorch.org`

- Biesse – `www.biessegroup.com`
- Ruby – `www.ruby-lang.org`
- Tableau – `www.tableau.com`
- Pragma4u – `www.pragma4u.it`
- Dialog – `www.dialog.it`
- realpython.com – `www.realpython.com`
- tecnlogia.libero.it – `www.tecnologia.libero.it`
- Wikipedia – `www.wikipedia.org`

Ringraziamenti

A conclusione di questo elaborato, desidero menzionare tutte le persone senza le quali questo lavoro di tesi non esisterebbe nemmeno.

In primis, ringrazio infinitamente la mia famiglia, che mi ha sempre sostenuto, appoggiando ogni mia decisione, fin dalla scelta del mio percorso di studi. Senza di voi non sarei mai potuto arrivare fin qui.

Ringraziamenti speciali vanno al Prof. Ursino e al Prof. Cauteruccio, per la loro immensa disponibilità e per i loro preziosi consigli.

Un grazie di cuore va anche ai miei amici di sempre per i bei momenti passati insieme in questi anni.

Grazie infinite a tutti voi.