Potenziare l'AI usando l'AI

Superare le sfide della digitalizzazione e della qualità dei dati attraverso l'Intelligenza Artificiale e la Generative AI

Roma, febbraio, 2025

L'innovazione, da sempre un imprescindibile ingrediente per competere su ogni mercato, oggi può fare leva su innumerevoli tecnologie, diversamente pervasive ed efficaci, che ci inducono a costruire soluzioni dirompenti a supporto di "noi umani".

È un percorso affascinante, mirato ad "aumentare" le nostre capacità intellettive aiutandoci a maturare decisioni consapevoli, sempre eticamente corrette e sostenibili: è un percorso che conduce a un'intelligenza aumentata, più genuina e concreta di una semplice intelligenza artificiale, è un'intelligenza che cresce grazie al nostro passato ed alla capacità di inferire nuova conoscenza, indirizzando virtuosamente i nostri comportamenti futuri.

Mario Ettorre

Chief Information Security Officer – Invitalia S.p.A.

Massimo Chiriatti

Chief Technology and Innovation Officer - ISG - Lenovo Italy

Sommario

Introduzione

Sebbene in numerosi campi applicativi la **Data Science (DS)**, **l'Artificial Intelligence (AI)** e la **Generative AI (GenAI)** abbiano compiuto notevoli progressi, facilitando sempre più l'estrazione di conoscenza dai dati grezzi, in altri contesti tali benefici si sono apprezzati sicuramente molto meno: si tratta prevalentemente di quegli ambiti in cui i processi di ***digital transf*ormation** sono stati eseguiti solo parzialmente, nella migliore delle ipotesi, dando origine ad una digitalizzazione "**monca**" dei processi target. Ciò, di fatto, ha inibito l'attuazione di sane pratiche di Data Science e GenAI che proprio nei dati ritrovano le loro fondamenta.

A complicare questo scenario di base contribuiscono **l'eterogeneità** e la **scarsa qualità** dei dati che, spesso, si riscontra anche in quei contesti più fortunati che hanno vissuto un'esperienza di trasformazione digitale, totale o parziale.

Osservando questo scenario sorge spontaneo chiedersi se le tecnologie di DS e (Gen)AI, piuttosto che essere immediatamente utilizzate per analizzare i dati (come prevalentemente fatto fino ad oggi), non possano essere preliminarmente sfruttate per colmare quel gap in termini di carenza e scarsa qualità di dati. In tal modo, si creerebbero le condizioni più favorevoli e il terreno più adatto per consentire a queste "nuove" tecniche di scaricare a terra tutto il loro potenziale!

In altri termini, ci chiediamo: è possibile chiedere all'AI di essere di supporto all'AI? Ovvero: "è possibile potenziare l'AI usando l'AI?"

In altri termini ci chiediamo: è possibile chiedere supporto all'AI? È possibile "potenziare" l'AI usando l'AI?

In questo breve articolo, esploreremo la problematica dei "**dati mancati"** e della "**qualità dei dati**" fornendo **esempi concreti** di utilizzo delle tecniche di Data Science e (Gen)AI per superare tali sfide.

Nota: nel prosieguo faremo una distinzione netta tra "**dati mancati**" e "**dati mancanti**":

- i primi sono riconducibili a tutte quelle mancate occasioni di digitalizzazione dei processi costringendoci, oggi, a constatare l'assoluta mancanza dei dati correlati ai processi di interesse;
- i "dati mancanti", invece, sono riconducibili a mancante imputazioni di dati, o altre cause a questa assimilabili.

La Generative AI: un boost per la digital transformation

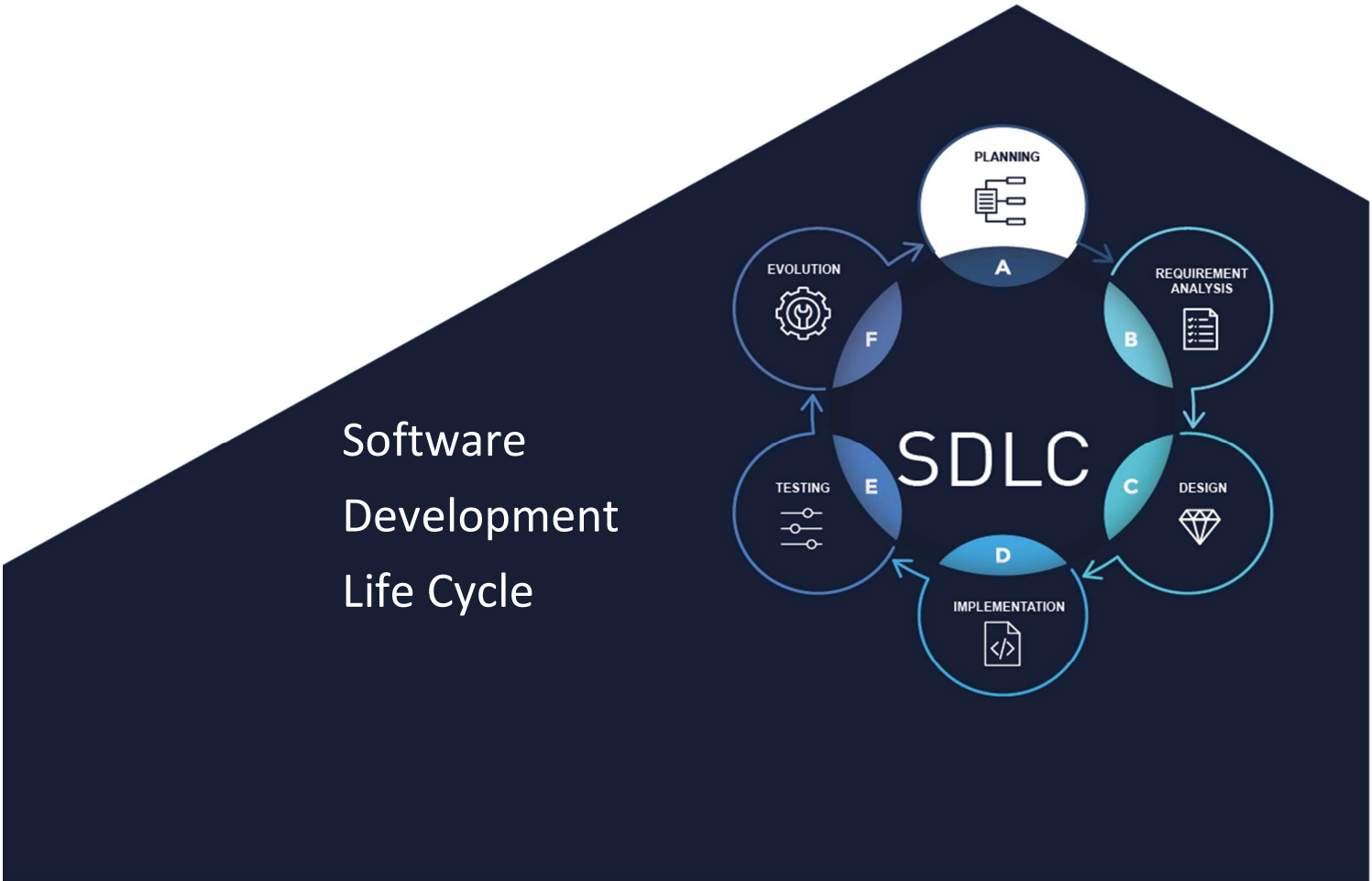
Un aspetto dirompente dell'Intelligenza Artificiale, in particolare della (Gen)AI, è sicuramente la capacità di supportare la generazione di applicazioni software, offrendo nuove opportunità per la rapida digitalizzazione di processi non ancora informatizzati.

In tale accezione, la (Gen)AI si presenta come uno strumento per il recupero efficiente del gap, in termini di “dati mancati”, che abbiamo ereditato dal passato e derivanti da processi di digitalizzazione parziali o, in alcuni casi, del tutto assenti. Per fare fronte a simili difficoltà, questi nuovi strumenti tecnologici possono essere utilizzati per supportare la fase di analisi e progettazione di nuove applicazioni, la generazione di codice finalizzato all'interfacciamento con database, l’implementazione di catene di ETL, la realizzazione di logica applicativa, spingendosi fino alla creazione di interfacce utente, alle elaborazioni statistiche descrittive-predittive-prescrittive e a molto altro ancora. [1]

In altri termini, le attività finalizzate all'informatizzazione di processi a vario grado di complessità, possono subire una forte accelerazione grazie alla reale trasformazione dell'intero ciclo di vita del processo di sviluppo software, fino a ieri prevalentemente manuale.

Sulla base di quanto sopra esposto, proprio in riferimento al *Software Development Lyfecycle* – **SDLC**, si riportano nel seguito alcune specifiche applicazioni e casi d'uso di GenAI volendo evidenziare come questa possa accelerare le varie fasi del suddetto *lifecycle*.

Quest'ultimo, di cui si riporta una schematizzazione nella figura seguente, è un approccio metodologico di riferimento nell'ambito dell'ingegneria del software: essenzialmente, esso delinea una serie ben strutturata di fasi del processo di sviluppo software, dal suo principio sino alla fine, con obiettivi principali e attività di riferimento. Di seguito, illustreremo una sua articolazione e i possibili utilizzi di Data Science e (Gen)AI finalizzati all'accelerazione e automazione delle singole parti o fasi identificate.



Planning

La fase iniziale del SDLC, comunemente denominata "Planning", riveste un'importanza cruciale nel determinare il successo complessivo del progetto. Durante questa fase, il team di sviluppo si impegna nell'identificazione e nella raccolta dei macro-requisiti del software. Ciò implica un'analisi delle esigenze del cliente, la valutazione dei vincoli tecnici e finanziari, nonché la delineazione di una strategia di gestione dei rischi. Il risultato di questa fase è rappresentato da documenti chiave che contengono la sintesi dei requisiti di massima ed un primo piano di progetto. Di seguito si riportano le principali attività che si svolgono durante la fase di planning.

- **Individuazione degli obiettivi**: si definiscono i principali obiettivi che si vogliono perseguire grazie al supporto del software e quali problemi si vogliono risolvere.
- **Identificazione dei macro-requisiti**: si elencano le macro-funzionalità e le principiali caratteristiche che il software dovrà garantire.
- **Stima del budget e del tempo di sviluppo**: si elaborano le stime dei costi di realizzazione e i relativi tempi di produzione.
- **Creazione di un piano di progetto**: si definiscono le attività che dovranno essere svolte e le tempistiche stimate per ciascuna di esse.

L'intelligenza artificiale e la Generative AI possono supportare la fase di planning in diversi modi.

- In merito all'individuazione degli obiettivi e alle relative priorità, è possibile usare tecniche quali:
    - **Natural Language Processing** (NLP) per estrarre informazioni da documenti di specifica, backlog di prodotto e storico dei ticket;
    - **Topic Modeling** per identificare i temi e le priorità che emergono dalle fonti documentali di progetto;

- **Sentiment Analysis e Opinion Monitoring** per valutare la soddisfazione degli utenti e le criticità eventualmente emerse dall'utilizzo di precedenti versioni del prodotto.

- In riferimento alla generazione di idee e possibili scenari, la Generative AI può essere utilizzata per generare suggerimenti quali nuove funzionalità o nuovi modi di utilizzo, grazie al supporto offerto dalle seguenti tecniche:
  - **Generative Adversarial Networks**[1] (GAN) per generare diverse alternative di design per l'interfaccia utente o l'architettura del software;
  - **Reinforcement Learning** per ottimizzare la scelta delle funzionalità in base a un set di obiettivi predefiniti;
  - **Genetic Algorithms** per esplorare lo spazio di progettazione e identificare ulteriori soluzioni innovative.

- Per quanto riguarda la stima del budget e del tempo di sviluppo, l'AI può essere utilizzata per produrre stime di maggiore precisione sfruttando, proficuamente, le seguenti tecniche:
  - **Machine Learning** per prevedere tempi e costi necessari per implementare le diverse funzionalità richieste;
  - **Regressione lineare** per stimare il budget complessivo del progetto;
  - **Simulazione Monte Carlo** per valutare l'impatto di fattori di rischio sul cronoprogramma e sul budget.

- La creazione di piani di progetto può essere assistita dalla AI per creare piani di progetto più efficienti e realistici tramite l'impiego delle seguenti tecniche:
  - **Scheduling algorithms** per ottimizzare la sequenza delle attività di sviluppo;

[1] Le Generative Adversarial Networks (GANs) sono un tipo di rete neurale artificiale composta da due modelli: un generatore e un discriminatore. Il generatore è progettato per creare nuovi dati che imitano la distribuzione di un dataset di riferimento, mentre il discriminatore ha il compito di distinguere tra i dati reali e quelli generati [2].

- **Resource allocation algorithms** per assegnare le risorse umane e finanziarie alle diverse attività;
- **Risk management algorithms** per identificare e mitigare i rischi del progetto.

Requirement Analysis

La fase successiva, nota come "Analysis", si concentra sulla comprensione dettagliata dei requisiti identificati nella fase precedente. Qui, il team di sviluppo analizza in modo critico i requisiti funzionali e non funzionali del software, al fine di definire con precisione le specifiche di input alla progettazione. Inoltre, durante questa fase, vengono esaminati i requisiti di sistema e si svolge un'analisi di fattibilità per valutare la coerenza tra le richieste del cliente e le risorse disponibili. Il risultato di questa fase è rappresentato da documenti quali l'analisi dei requisiti di dettaglio e il progetto concettuale.

L'AI può supportare la fase di "Requirement Analysis" per:

- analizzare grandi volumi di dati di mercato e di clienti volendo identificare le esigenze e le opportunità di business; ciò è particolarmente utile per suggerire eventuali *feature* che, sebbene non emerse dalle prime interlocuzioni con il cliente, possono risultare utili sulla base delle evidenze che emergono dalla "*competitive market analysis";*
- generare automaticamente casi d'uso e *storyboard* per migliorare la comprensione dei requisiti;
- sviluppare prototipi interattivi per facilitare la validazione dei requisiti con il supporto degli *stakeholder*;
- generare automaticamente documentazione di specifica dei requisiti in linguaggio naturale; in particolare, a partire dalle registrazioni dei meeting, è possibile

acquisire, tradurre e sintetizzare in modo automatico, i verbali dai quali la Generative AI può estrarre i *topic* rilevanti ed un possibile *feature catalogue*;

- creare modelli di dati realistici per testare la fattibilità dei requisiti.

Design

La fase di "Progettazione" rappresenta il punto in cui i requisiti identificati vengono tradotti in un design tecnico concreto. In questo contesto, il team di sviluppo si impegna nella progettazione dell'architettura del software, stabilendo la struttura generale del sistema e specificando i dettagli di implementazione dei singoli componenti. Questa fase comprende anche l'attività di prototipazione, attraverso la quale vengono creati modelli iniziali del software per valutarne la funzionalità e l'usabilità. Il risultato di questa fase è rappresentato da documenti quali la progettazione architetturale e la progettazione di dettaglio.

L'AI e la GenAI possono essere utilizzate per:

- progettare, in modalità semi-automatica, modelli di architetture software ottimali;
- identificare potenziali problemi di progettazione e suggerire soluzioni alternative;
- stimare lo sforzo di sviluppo, la tempistica del progetto e la distribuzione ottimale delle risorse sulle attività tramite tecniche di ottimizzazione e reasoning [3];
- generare automaticamente diagrammi UML e altri artefatti di progettazione;
- creare prototipi funzionali per testare l'architettura del software.

Implementation

Nella fase di Implementazione, il design elaborato nella fase precedente viene effettivamente implementato sotto forma di codice sorgente. Questa fase coinvolge attivamente gli sviluppatori nel processo di scrittura, testing e integrazione del codice, al fine di creare un sistema funzionante e coerente con i requisiti raccolti. È importante sottolineare che il controllo di versione e la gestione dei cambiamenti giocano un ruolo chiave in questa fase, assicurando la coerenza e l'integrità del codice sorgente prodotto. Il risultato di questa fase è rappresentato dal software codificato e integrato.

L'AI può essere utilizzata per:

- sviluppare codice sorgente automaticamente o suggerire il completamento del codice in base al contesto di riferimento; un esempio concreto di questo approccio è **Codex**, un modello di linguaggio AI che può generare codice sorgente in risposta a descrizioni di problemi di programmazione fornite in linguaggio naturale [4];
- tradurre codice sorgente preesistente verso un altro linguaggio (ad es. da C++ a Scala o da Fortran a Java, ecc.), accelerando efficacemente i processi di *refactoring*, *reengineering* e *reverse engineering* di "vecchio codice" di difficile manutenzione per carenza di conoscenze e mancanza di documentazione;
- identificare bug e vulnerabilità di sicurezza nel codice sorgente;
- generare in modo automatico la documentazione relativa al codice prodotto;
- eseguire automaticamente gli unit test per garantire la qualità e l'affidabilità del software.

Testing

La fase di "Test" è dedicata alla verifica e alla validazione del software sviluppato per garantire il rispetto dei requisiti raccolti ed il corretto funzionamento. Durante questa fase viene eseguito un insieme di test funzionali, prestazionali e di sicurezza, al fine di individuare eventuali difetti o problemi insiti nel sistema. Il risultato di questa fase è rappresentato da report di test dettagliati e da una versione del software pronta per la distribuzione.

Anche in questa fase la Generative AI può essere utilizzata in modo particolarmente proficuo per:

- generare automaticamente il test plan ed una corrispondente quantità di test case (funzionali ed integrati) per aumentare considerevolmente la copertura del codice sviluppato;
- creare dati sintetici di test[2] per sollecitare e collaudare scenari estremi, difficili da replicare utilizzando esclusivamente dati reali; inoltre, grazie alla produzione massiva di dati sintetici, è possibile verificare le prestazioni del codice prodotto anche su larga scala riducendo i costi da sostenere nel caso in cui si volessero utilizzare esclusivamente dati reali.

[2] I dati sintetici sono dati generati artificialmente da algoritmi o modelli, piuttosto che essere raccolti o osservati nella realtà

Distribution

La fase di "Distribuzione" rappresenta il momento in cui il software viene effettivamente rilasciato al cliente o agli utenti finali. Durante questa fase, il software viene installato nell'ambiente di produzione, gli utenti vengono formati sull'uso del nuovo sistema e viene fornito supporto post-implementazione per risolvere eventuali problemi o domande. È importante garantire una transizione senza intoppi verso il nuovo sistema, minimizzando al contempo l'impatto sugli utenti esistenti. Il risultato di questa fase è rappresentato da una versione completamente operativa del software, accompagnata dall'adeguata formazione e supporto forniti agli utenti (*adoption*).

Oggi, al centro della trasformazione digitale delle organizzazioni e in riferimento alla fase di *Distribution*, ritroviamo il paradigma *dell'Infrastructure as Code* (IaC), un approccio moderno alla gestione e alla distribuzione delle infrastrutture digitali. L'IaC, in stretta coniugazione con le risorse offerte dal *cloud computing*, consente agli sviluppatori di definire e gestire l'intera configurazione dell'infrastruttura attraverso apposito codice, trasformando le operazioni di *provisioning*, configurazione e gestione delle risorse in un processo automatizzato e definito tramite linguaggi di programmazione standard. Questo approccio ha prodotto una serie di vantaggi significativi che hanno rivoluzionato il modo in cui le infrastrutture digitali sono create, gestite e distribuite.

I vantaggi principali derivanti da un virtuoso utilizzo dell'IaC sono molteplici:

- in primo luogo, l'IaC ha introdotto una maggiore **automazione e riproducibilità** nell'implementazione delle infrastrutture, riducendo drasticamente le possibilità di errori riconducibili alle configurazioni manuali. Questo approccio automatizzato facilita la consistenza e la coerenza tra le infrastrutture degli ambienti di sviluppo, test e produzione semplificando il processo di distribuzione;
- in secondo luogo, l'IaC favorisce la **scalabilità e la flessibilità** delle infrastrutture, consentendo agli sviluppatori di aggiungere o rimuovere risorse in modo dinamico in risposta alle esigenze derivanti dal carico di lavoro in continuo cambiamento. Questo approccio basato sul codice rende più agevole l'adattamento

dell'infrastruttura alle richieste dei servizi e delle applicazioni, migliorando l'efficienza operativa complessiva;

- in terzo luogo, l'IaC promuove la **standardizzazione** e la **gestione centralizzata** delle configurazioni, permettendo agli sviluppatori di definire modelli di infrastruttura riproducibili e facilmente gestibili. Ciò semplifica la collaborazione tra team, riduce la complessità della gestione delle risorse e facilita la conformità alle *best practice* e agli standard di sicurezza;
- infine, l'IaC favorisce la **cultura DevOps e l'integrazione continua**, consentendo agli sviluppatori di automatizzare completamente il ciclo di vita dello sviluppo e della distribuzione del software. L'infrastruttura è trattata come parte integrante del codice dell'applicazione, consentendo agli sviluppatori di testare e distribuire le modifiche all'infrastruttura insieme al codice dell'applicazione stessa, garantendo una distribuzione rapida e affidabile dei servizi.

Nella fase di Distribution, la (Gen)AI può essere utilizzata per:

- generare automaticamente codice IaC basato su specifiche esigenze e requisiti. Ciò può accelerare il processo di *deploy* dell'infrastruttura, riducendo la necessità di scrivere manualmente il codice IaC *from scratch*;
- ottimizzare il codice IaC esistente, identificando e suggerendo miglioramenti nelle pratiche di codifica, nell'efficienza delle risorse, nella gestione delle dipendenze e nella scalabilità dell'infrastruttura;
- validare il codice IaC generato, utilizzando modelli di apprendimento automatico e rilevando potenziali errori, incoerenze o violazioni delle best practice. Questo aiuta a garantire la correttezza e la sicurezza dell'infrastruttura gestita come codice;
- la Generative AI può essere utilizzata per creare modelli predittivi dell'infrastruttura in base ai dati storici e ai pattern di utilizzo. Questi modelli possono essere utilizzati per pianificare e ottimizzare la distribuzione delle risorse, migliorando la scalabilità, la disponibilità e le prestazioni complessive dell'infrastruttura;
- integrando la Generative AI con sistemi di automazione e orchestrazione, è possibile automatizzare le operazioni di gestione e manutenzione dell'infrastruttura,

consentendo una risposta rapida agli eventi, l'ottimizzazione delle risorse, la gestione dei carichi di lavoro e il *provisioning* in modo dinamico, nonché l'ottimizzazione dei costi.

Evolution

Questa fase si focalizza sul supporto continuo e sulla gestione del software successiva alla fase di rilascio: essa comprende attività come la correzione degli errori, l'implementazione di miglioramenti e l'applicazione di aggiornamenti di sicurezza. L'obiettivo principale di questa fase è garantire che il software funzioni correttamente nel tempo mantenendo, al contempo, la sua affidabilità e la sua efficacia. Il risultato di questa fase si sostanzia in versioni aggiornate del prodotto software, sempre accompagnate da documentazione dettagliata relativa alle modifiche apportate.

In questa fase la (Gen)AI può essere utilizzata per:

- identificare e correggere in modo (semi)automatico i bug rilevati nel software generando le opportune *patch*;
- prevedere guasti e anomalie;
- ottimizzare le prestazioni;
- generare automaticamente la documentazione di accompagnamento alle varie *release*;
- analizzare automaticamente i *feedback* provenienti dagli utenti per intercettare le possibili *change request* tramite impiego virtuoso delle tecniche di NLP e (Gen)AI;
- automatizzare la gestione dei ticket producendo, qualora possibile, risposte immediate alle problematiche sollevate, oppure predisponendo appositi PBI per la produzione delle necessarie evoluzioni.

Da quanto esposto nelle precedenti sezioni, appaiono evidenti i numerosi benefici che scaturiscono dall'applicazione della Data Science e della (Gen)AI al ciclo di vita del software elevandolo ad una dimensione moderna, più efficiente e supportata dalle nuove tecnologie: ci apprestiamo a vivere l'era del "**SDLC+**".

Tra i principali benefici individuati si ricordano, in modo non esaustivo, i seguenti:

- decisa accelerazione per i processi di digitalizzazione non ancora avviati o ancora incompleti;
- maggiore efficienza e produttività, poiché è possibile automatizzare numerose attività ripetitive e dispendiose in termini di tempo, liberando gli sviluppatori per concentrarsi su compiti più creativi e strategici;
- migliore qualità del software in quanto è possibile identificare e correggere i bug in modo più efficace;
- riduzione dei costi derivante, proporzionalmente, dall'automazione di diverse fasi del SDLC;
- supporto all'innovazione poiché l'AI e la Generative AI possono facilitare l'esplorazione di nuove idee e la sperimentazione di nuovi approcci allo sviluppo del software.

Software/System Development Life Cycle

- Maggiore efficienza e produttività, poiché è possibile automatizzare numerose attività ripetitive e dispendiose in termini di tempo.
- Migliore qualità del software in quanto è possibile identificare e correggere i bug in modo più efficace.
- Riduzione dei costi derivante, proporzionalmente, dall'automazione di diverse fasi del SDLC.
- Supporto all'innovazione poiché l'AI e la Generative AI possono facilitare l'esplorazione di nuove idee e la sperimentazione di nuovi approcci allo sviluppo del software.

La Generative AI per migliorare la qualità dei dati

La qualità dei dati è un aspetto cruciale alla base dei processi di *decision-making* il cui successo è direttamente dipendente dalla qualità dei dati stessi. Errori, inconsistenze e incompletezze nei dati possono compromettere la validità delle inferenze: alcuni autori sostengono che la qualità dei dati può contribuire fino all'80% del successo di un progetto [5] conducendo a conclusioni imprecise o fuorvianti e limitando, di fatto, la capacità dell'AI di distillare conoscenza utile (*actionable insights*) [6].

I problemi di scarsa qualità dei dati possono essere riconducibili ad un'ampia varietà di cause tra cui, in modo esemplificativo e non esaustivo, si riportano le seguenti.

- **Errori umani**: tali errori sono la causa più frequente di una scarsa qualità dei dati e possono essere commessi durante la raccolta, l'inserimento o la modifica dei dati nei sistemi attraverso apposite applicazioni.
- **Problemi di duplicazione**: i dati duplicati alterano la realtà dei fatti e le risultanze statistiche.
- **Bug:** tipicamente contenuti nelle procedure software per la produzione, acquisizione ed elaborazione dei dati stessi.
- **Dati non disponibili** sulla base di alcuni fattori quali, ad esempio, la privacy o la sicurezza.
- **Dati mancanti o incompleti:** le cause sono variamente riconducibili alle ragioni esposte nei punti precedenti.

In questa sezione, focalizzando specifiche problematiche di *Data Quality*, ci chiediamo se e come l'AI possa concretamente intervenire per affrontare e risolvere efficacemente le problematiche individuate: **in altri termini ci chiediamo, ancora una volta, se l'AI possa essere di aiuto all'AI!**

Incompletezza

Il problema dei dati mancanti o incompleti rappresenta una sfida intrinseca alla gestione e all'analisi dei dati in molteplici ambiti scientifici e applicativi [7]. Questa problematica è pervasiva e può derivare da una serie di fattori che influenzano il processo di raccolta, l'archiviazione e la gestione dei dati. Le cause sottostanti alla mancanza di dati possono essere raggruppate in diverse categorie che comprendono sia fattori tecnici che comportamentali, presentando interazioni complesse tra di esse.

Fattori tecnici

- **Errori di raccolta**: la mancanza di dati può essere dovuta a errori o inefficienze nel processo di raccolta dei dati stessi. Questi errori possono derivare da dispositivi di

misurazione difettosi, procedure di campionamento non rappresentative o difetti nel sistema di acquisizione dati.

- **Problemi di archiviazione e trasmissione**: durante il processo di archiviazione e trasmissione dei dati, possono verificarsi errori o interruzioni che portano alla perdita o alla corruzione dei dati stessi. Incompatibilità tra diversi sistemi di archiviazione o di trasmissione dei dati possono contribuire ad accentuare ulteriormente questo problema.
- **Mancanza di standardizzazione**: l'assenza di standardizzazione nei formati dei dati o nelle procedure di registrazione può rendere difficile l'integrazione e la comparazione dei dati provenienti da diverse fonti, aumentando il rischio di dati mancanti o incompleti.

Fattori comportamentali

- **Non conformità degli utenti**: gli utenti coinvolti nel processo di raccolta e gestione dei dati potrebbero agire in modo non pienamente conforme alle procedure stabilite o potrebbero omettere di registrare alcune informazioni rilevanti, contribuendo così alla mancanza di dati.
- **Errori umani**: esempio come la dimenticanza o la negligenza, possono portare alla mancanza di dati durante il processo di registrazione o di inserimento dei dati nei sistemi informativi.
- **Resistenza al cambiamento**: l'introduzione di nuovi sistemi o procedure per la gestione dei dati potrebbe incontrare resistenza da parte degli utenti, influenzando negativamente la qualità e la completezza dei dati raccolti.

Volendo affrontare le problematiche sopra illustrate, attingendo al mondo delle tecniche più sofisticate di Generative AI, a titolo esemplificativo si propone l'utilizzo delle GAN (*Generative Adversarial Network*).

Le GAN, come precedentemente accennato, sono essenzialmente composte da due reti neurali antagoniste, il generatore e il discriminatore, che sono addestrate simultaneamente in modo competitivo. Il generatore cerca di generare dati artificiali che siano indistinguibili dai dati reali, mentre il discriminatore cerca di distinguere tra dati reali e dati artificiali

prodotti dal generatore. Questa competizione porta alla continua raffinazione delle capacità del generatore nel creare dati sempre più realistici.

Le GAN, dunque, possono essere utilizzate per generare **dati sintetici** che imitano la distribuzione dei dati originali, permettendo di "riempire" i dati mancanti in un set di dati [8]. Questo approccio può essere particolarmente utile in diversi domini applicativi come, ad esempio, quello della sanità nel quale i dati mancanti possono rappresentare una solida barriera per l'esecuzione dei protocolli medici previsti [9].

Un esempio di utilizzo della Generative AI per la gestione di dati mancanti è fornito da uno studio condotto da ricercatori dell'Università di Cambridge [9]. In questo studio, le *Generative Adversarial Networks* (GANs) sono state impiegate per generare dati sintetici volti a colmare le lacune informative presenti in un set di dati sulla malattia di Alzheimer.

Tale approccio ha consentito ai ricercatori di:

- **aumentare la completezza del dataset**, la generazione di dati sintetici ha permesso di integrare le informazioni mancanti, rendendo il dataset più completo e rappresentativo della popolazione di riferimento;
- **migliorare l'accuratezza delle analisi**, l'utilizzo di dati completi e realistici ha prodotto un aumento dell'accuratezza e dell'affidabilità delle analisi condotte sull'Alzheimer;
- **ottenere nuove informazioni**, l'analisi del dataset completo ha permesso ai ricercatori di identificare nuove informazioni e pattern che non erano evidenti nel dataset originale con dati mancanti.

Nel contesto della gestione dei dati mancanti, trovano ampio spazio di applicazione ulteriori tecniche di *machine learning* come alberi decisionali, regressione e *k-Nearest Neighbors* (KNN). Ad esempio, la tecnica di imputazione dei dati MICE (*Multiple Imputation by Chained Equations*) utilizza modelli di regressione multipla per predire i valori mancanti basandosi sui valori assunti dalle altre variabili. Casi di successo includono l'applicazione di MICE nell'analisi dei dati del settore sanitario per migliorare la precisione della previsione relativa alle prestazioni cliniche [11].

Così anche la tecnica KNN ha trovato efficace implementazione per stimare ed inserire l'età mancante di pazienti in un dataset medico, migliorando l'accuratezza di modelli predittivi del rischio di malattie [12].

Inconsistenza e incoerenza

L'inconsistenza e la contraddittorietà dei dati rappresentano due sfide significative per la *data quality*, avendo un impatto significativo sull'affidabilità e sull'utilità delle informazioni:

- **l'inconsistenza** si verifica quando lo stesso valore o concetto è rappresentato in modo diverso in differenti dataset o all'interno dello stesso dataset. Ad esempio, la data di nascita di un contribuente potrebbe essere registrata in formati differenti (gg/mm/aaaa vs mm/gg/aaaa) o con errori di battitura;
- **la contraddittorietà** si verifica quando due o più dati all'interno dello stesso ecosistema informativo si contraddicono a vicenda. Ad esempio, un cliente potrebbe essere registrato come "attivo" in un sistema e, allo stesso tempo, come "inattivo" in un altro. O, ancora, in corrispondenza dello stesso cliente, potremmo rilevare, allo stesso tempo, indirizzi di residenza diversi su sistemi diversi. Tali incongruenze ostacolano la corretta profilazione del cliente e la fornitura di servizi personalizzati.

Le problematiche sopra descritte possono derivare da diverse cause, tra cui: errori umani, processi di integrazione inefficienti a causa di mancata standardizzazione dei formati e delle terminologie, duplicazione delle informazioni, fonti dati non affidabili poiché incomplete, errate o non aggiornate.

Per contrastare queste problematiche, è fondamentale implementare *best practice* per la *data quality*, come:

- **standardizzazione dei formati e delle terminologie** per garantire un'uniformità nella rappresentazione dei dati;
- **validazione e verifica dei dati** per identificare e correggere errori e incongruenze;
- **integrazione dati efficiente** per combinare dati da diverse fonti in modo coerente e affidabile;
- **utilizzo di strumenti di *data quality*** per monitorare e migliorare la qualità dei dati nel tempo.

L'adozione di queste best practice permette di migliorare la consistenza e la coerenza dei dati, garantendo una migliore affidabilità e un maggiore valore per le informazioni aziendali incrementando, di conseguenza, la fiducia nei dati stessi.

Volendo perseguire tali obiettivi, possiamo ricorrere ancora agli algoritmi di AI quali il data  ***matching* e il *record linkage*.**

- ***Data matching***: si concentra sull'identificazione di record duplicati all'interno dello stesso dataset. Algoritmi come *Jaccard similarity* o *Levenshtein distance* calcolano la distanza tra due record, permettendo di identificarne la duplicazione con un elevato grado di accuratezza. In questo contesto il termine "distanza" indica il numero minimo di modifiche elementari necessarie per trasformare una stringa in un'altra. Le modifiche elementari ammesse sono: la sostituzione, l'inserimento e la cancellazione di un carattere in una stringa.
- ***Record linkage***: estende il *data matching* all'identificazione di record simili tra dataset differenti. La tecnica si avvale di algoritmi di apprendimento automatico per estrarre caratteristiche salienti dai dati e comparare record di differenti fonti, anche in presenza di formati e terminologie non coerenti.

Numerosi sono gli esempi provenienti dal mondo industriale che testimoniano i benefici tangibili derivanti dall'applicazione delle tecniche sopra indicate, ad esempio, l'implementazione del *record linkage* per identificare frodi nel settore delle telecomunicazioni [13].

Duplicazione

La duplicazione dei dati rappresenta un fenomeno significativo nell'ambito dell'informatica e del *data management*, con implicazioni rilevanti in termini di risorse computazionali, spazio di archiviazione e sicurezza delle informazioni. Questo fenomeno si verifica quando gli stessi dati vengono replicati in più posizioni all'interno di un sistema o tra sistemi differenti. Le cause della duplicazione dei dati possono derivare da processi anomali di sincronizzazione, errori di programmazione, requisiti di *backup* o meccanismi di *caching*.

L'AI può essere utilizzata per migliorare la qualità dei dati attraverso tecniche di *data-cleaning*, come l'eliminazione di duplicati, la correzione di errori e l'armonizzazione dei formati dei dati [14].

Ad esempio, Google utilizza l'AI per pulire e organizzare i dati di Google Maps, migliorando la precisione e l'utilità del servizio.

Le tecniche di deduplicazione basate sull'apprendimento automatico possono essere utilizzate per identificare record duplicati o simili all'interno di un set di dati. Queste tecniche possono includere l'uso di algoritmi di *clustering* o di confronto di stringhe basati su feature, ciò al fine di calcolare la similarità tra i record e identificare quelli che potrebbero essere duplicati. L'algoritmo di confronto di stringhe basato su feature TF-IDF (*Term Frequency-Inverse Document Frequency*) è ampiamente utilizzato per l'individuazione di record duplicati.

Rumore

Nel contesto dell'analisi dei dati, il rumore si riferisce a qualsiasi componente indesiderata che pregiudica l'integrità e la veridicità di un set di dati. In parole più semplici, si tratta di informazioni errate, inconsistenti o incomplete che si mescolano ai dati reali, rendendone difficoltosa l'analisi e l'interpretazione.

Esistono diverse tipologie di rumore nei dati, classificabili in base alle loro caratteristiche:

- **rumore casuale**, valori casuali e imprevedibili che si discostano dalla distribuzione naturale dei dati;
- **rumore sistematico**, distorce i dati in modo costante come, ad esempio, accade nel caso di errori di misurazione;
- **rumore a picco**, valori anomali che si discostano significativamente dalla media del dataset.

Nel caso dei dati rumorosi, gli algoritmi di *machine learning* possono essere utilizzati per filtrare i dati e rilevare anomalie *(outlier detection)*.

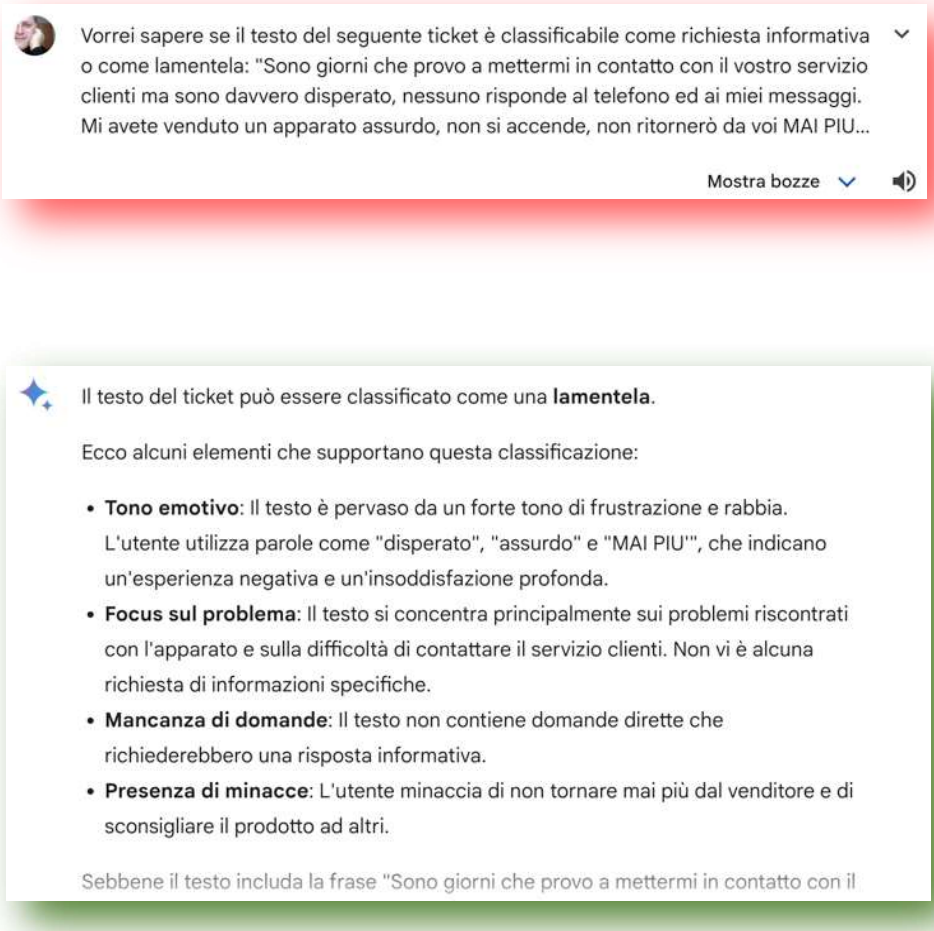
Algoritmi come LOF (*Local Outlier Factor*), *Isolation Forest* [15][16], *Support Vector Machines* (SVM) e tecniche di clustering possono identificare i dati che deviano significativamente dai modelli di comportamento normale e rimuoverli o trattarli in modo appropriato.

A tal proposito, si cita l'applicazione di *Isolation Forest* per la rilevazione di frodi nelle transazioni finanziarie, facendo registrare ottime prestazioni [17].

Anche i dati erroneamente classificati possono essere considerati una particolare manifestazione di "rumore" all'interno delle nostre sorgenti informative. Si supponga, ad esempio, di voler analizzare la distribuzione percentuale dei ticket registrati nel contesto di un Contact Center rispetto alle due principali categorie (Richiesta Informazioni, Reclamo). È evidente che un erroneo processo di classificazione comporterà risultanze numeriche del tutto fuorvianti che potrebbero innescare azioni correttive inappropriate. Anche in siffatti casi, l'AI può intervenire in modo piuttosto efficace riclassificando automaticamente i ticket

in base al testo in esso contenuti e prodotto tramite trascrizione automatica delle conversazioni intercorse tra utente e Contact Center.

Nella figura sottostante si riporta un semplice esempio di ri-classificazione eseguita tramite apposito prompt, è stato chiesto al motore di classificare il ticket nel riquadro rosso, inizialmente classificato come “Richiesta Informativa” all’interno del CRM. Nel riguardo verde è possibile apprezzare il risultato della corretta classificazione.



Bias

Il *bias* nei dati, o pregiudizio, è una problematica critica nell'ambito dell'Intelligenza Artificiale (AI): esso può introdursi in vari stadi del ciclo di vita dei dati, dalla raccolta all'elaborazione e all'analisi, producendo risultati distorti, decisioni non eque e potenziali discriminazioni. Ad esempio, un algoritmo di *Machine Learning* addestrato su un *dataset* con *bias* di genere o razziale può perpetuare o peggiorare tali pregiudizi nelle sue previsioni. Ancora, se un dataset utilizzato per addestrare un modello di *screening* per il cancro contiene principalmente casi di pazienti di sesso maschile, il modello potrebbe essere meno accurato nel diagnosticare il cancro nelle pazienti di sesso femminile. Altri tipi di *bias* includono il *bias* temporale, in cui i dati più recenti sono sovra rappresentati rispetto ai dati più vecchi, e il *bias* di attribuzione, in cui vengono assegnate erroneamente determinate caratteristiche ai dati.

Il *bias* può essere introdotto durante la raccolta dei dati, ma può anche derivare da pregiudizi impliciti negli strumenti di raccolta dei dati o nelle decisioni prese durante il *cleaning* e l'elaborazione dei dati. Infine, il *bias* può essere introdotto durante l'analisi dei dati, ad esempio, attraverso la scelta di modelli o metodi di analisi inappropriati.

Per mitigare il problema del bias nei dati, è possibile adottare diverse contromisure specifiche offerte dalle tecniche di Data Analytics e AI, tra cui si riportano le seguenti a titolo esemplificativo:

- **raccolta dei dati**, è necessario verificare che il campione raccolto sia rappresentativo della popolazione di interesse; questo può essere ottenuto attraverso tecniche di campionamento stratificato o ponderato, che mirano a garantire che tutti i gruppi all'interno della popolazione siano adeguatamente rappresentati;
- **elaborazione dei dati**, durante questa fase è importante identificare e correggere eventuali errori o pregiudizi; tale obiettivo si può perseguire attraverso l'uso di tecniche di normalizzazione o standardizzazione dei dati che mirano a ridurre il *bias* sistemico. Inoltre, i metodi di imputazione possono essere utilizzati per gestire i dati mancanti, riducendo il rischio di *bias* dovuto a dati incompleti;

- **tecniche di *debiasing/fairness-aware learning*** che garantiscono la robustezza dispetto ai pregiudizi. Ad esempio, gli algoritmi di apprendimento equo sono progettati per essere insensibili ai pregiudizi nei dati di addestramento. Questi algoritmi possono essere utilizzati per addestrare modelli di ML che producono risultati equi, indipendentemente dal *bias* nei dati di addestramento [18];
- **valutazione e monitoraggio dei modelli**, è fondamentale valutare e monitorare in modo continuo i modelli di AI per identificare e correggere eventuali pregiudizi. Questo può essere ottenuto attraverso l'uso di metriche di equità, come la parità demografica o la parità di opportunità, che misurano la performance del modello su diversi gruppi di popolazione.

Conclusioni

È quindi possibile potenziare l'AI usando l'AI?

Questo breve articolo nasce dalla voglia di volere investigare le possibili risposte a questa domanda, navigando lungo una dimensione che nasconde molti altri quesiti, tutti figli di un'innovazione che si presenta come un mare in tempesta dalla forza dirompente e incontenibile.

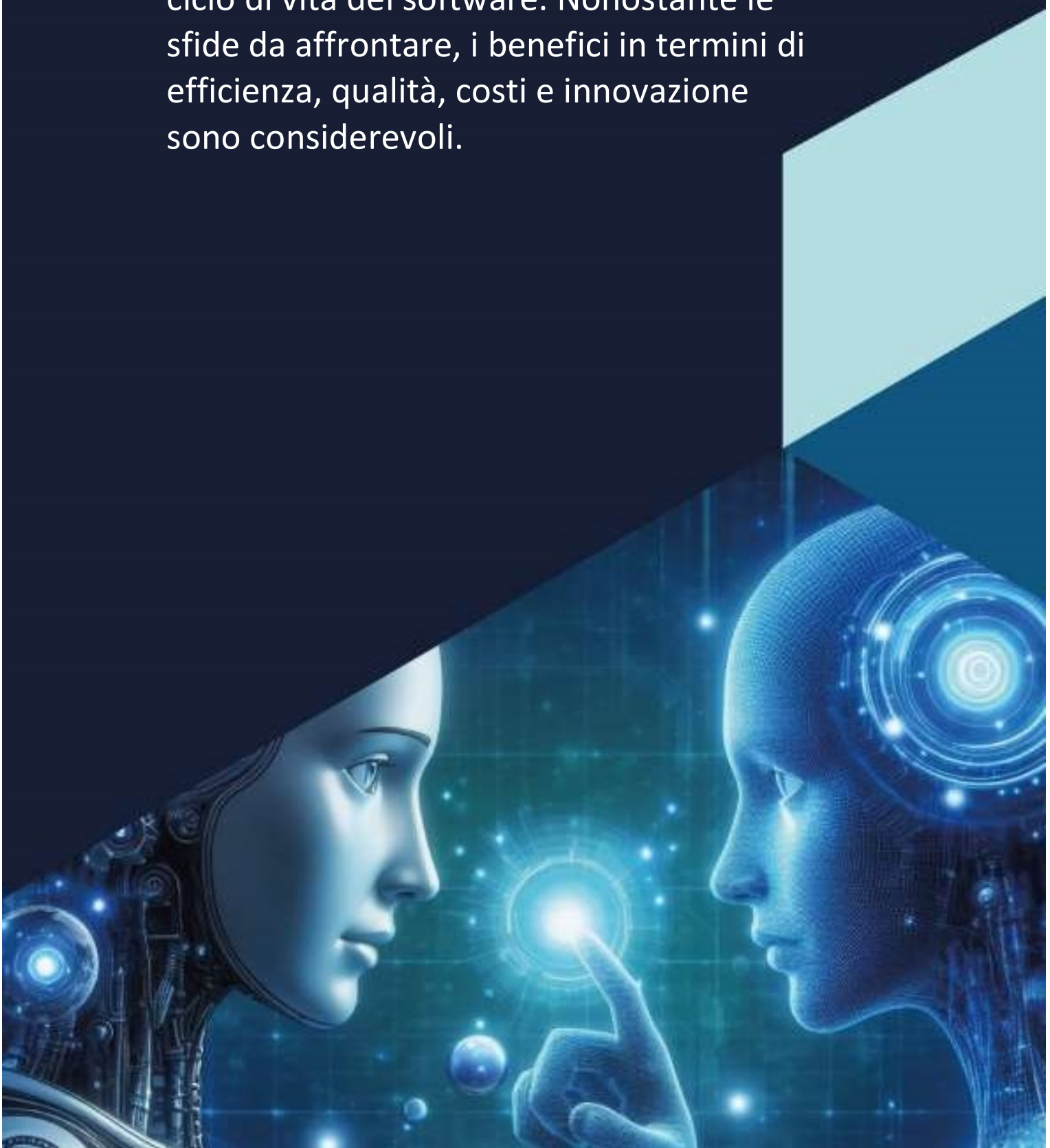
E in questo mare, tra le fazioni opposte rappresentate dagli irriducibili entusiasti dell'AI da un lato, e dai negazionisti/oppositori convinti dall'altro, c'è chi, come noi, preferisce la terza via: la via della conoscenza e dello studio continuo per poter controllare e dominare l'innovazione tecnologica, sempre a servizio dell'umano.

Siamo all'inizio di un percorso che, inesorabilmente, trasformerà i contorni delle nostre professioni, il nostro modo di "fare software", le strategie di "analisi dei dati". A tale proposito, questo breve articolo non ha l'ambizione di fornire risposte "scolpite" o definire nuove pietre miliari metodologiche ma, più semplicemente, vuole fornire uno spunto di riflessione sulle modalità con cui possiamo sollecitare l'AI al fine di ottenere dall'AI stessa il meglio di sé!

In particolare, si è voluto evidenziare il potenziale dell'AI e della GenAI quali "facilitatori" del processo di trasformazione digitale, e strumenti per il miglioramento continuo della qualità dei dati e del conseguente processo di estrazione di conoscenza.

Nondimeno, l'uso di queste tecnologie richiede l'adozione di un approccio consapevole, eticamente corretto, sicuro e solidamente costruito su competenze distintive, preservando il progresso e il benessere globale.

L’AI e la Generative AI offrono un enorme potenziale per trasformare il ciclo di vita del software. Nonostante le sfide da affrontare, i benefici in termini di efficienza, qualità, costi e innovazione sono considerevoli.



Bibliografia

1. Kandel, S., Paepcke, A., Hellerstein, J., & Heer, J. (2020). Wrangler: Interactive Visual Specification of Data Transformation Scripts
2. Goodfellow, I., et al. (2014). Generative Adversarial Nets. In Advances in Neural Information Processing Systems 27 (NIPS 2014).
3. Leone, Nicola & Pfeifer, Gerald & Faber, Wolfgang & Eiter, Thomas & Gottlob, Georg & Perri, Simona & Scarcello, Francesco. (2002). The DLV System for Knowledge Representation and Reasoning. ACM Transactions on Computational Logic (TOCL).
4. Kumar, Aman & Sharma, Priyanka. (2023). Open AI Codex: An Inevitable Future? International Journal for Research in Applied Science and Engineering Technology.
5. Redman, T. C. (1996). Data Quality for the Information Age. Artech House. ISBN:978-0-89006-883-0
6. Batini & Scannapieco, "Qualità dei Dati: Concetti, Metodi e Tecniche", Springer Milano, maggio 2018, ISBN-13/ 978-8847007338.
7. Roderick Little, Donald Rubin, Statistical Analysis with Missing Data, Third Edition | Wiley Series in Probability and Statistics, 2019, ISBN:9780470526798
8. Yoon, J., Jordon, J., & van der Schaar, M. (2018). GAIN: Missing Data Imputation using Generative Adversarial Nets. ICML.
9. Beaulieu-Jones, B. K., & Moore, J. H. (2017). Missing data imputation in the electronic health record
10. Bowles, C., et al. (2018). GAN Augmentation: Augmenting Training Data using Generative Adversarial Networks. ArXiv.
11. Li, F., & Enders, C. K. (2019). Using MICE for handling missing data in clinical trials with noninferiority
12. Batista, Gustavo & Monard, Maria-Carolina. (2002). A Study of K-Nearest Neighbour as an Imputation Method Hybrid Intelligent Systems, ser Front Artificial Intelligence Applications. 30. 251-260.
13. Thomas, Satish & Sluss, James. (2023). Fraud detection through data sharing using privacy-preserving record linkage, digital signature (EdDSA), and the MinHash

technique: Detect fraud using privacy preserving record links. The Journal of Engineering. 2023. 10.1049/tje2.12341.

14. Hellerstein, J. (2008). Quantitative data cleaning for large databases. United Nations Economic Commission for Europe.
15. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining (pp. 413-422). IEEE.
16. Zhang, J., Zou, H., Li, H., & Wang, X. (2019). A novel fraud detection method based on isolation forest with feature selection. Information Sciences, 505, 1-18.
17. Waspada, Indra & Bahtiar, Nurdin & Wirawan, Panji & Awan, Bagus. (2020). Performance Analysis of Isolation Forest Algorithm in Fraud Detection of Credit Card Transactions. Khazanah Informatika: Jurnal Ilmu Komputer dan Informatika. 6. 10.23917/khif.v6i2.10520.
18. Caton, Simon & Haas, Christian. (2023). Fairness in Machine Learning: A Survey. ACM Computing Surveys. 10.1145/3616865.