

Syllabus Certified Tester AI Testing

Versione 2.0

International Software Testing Qualifications Board



Realizzato da
Alliance for Qualification, Artificial Intelligence United, Chinese Software Testing
Qualifications Board e Korean Software Testing Qualifications Board



Avviso di Copyright

Avviso di Copyright © International Software Testing Qualifications Board (di seguito denominato ISTQB®)

ISTQB® è un marchio registrato dell'International Software Testing Qualifications Board.

Copyright © 2026, gli autori Klaudia Dussa-Zieger (chair), Stuart Reid, Vipul Kocher, Qin Liu, Jarosław Hryszko, Kyle Alexander Siemens e Werner Henschelchen.

Copyright © 2021, gli autori Klaudia Dussa-Zieger (chair), Vipul Kocher, Qin Liu, Stuart Reid, Kyle Alexander Siemens, Werner Henschelchen e Adam Leon Smith.

Tutti i diritti riservati. Con la presente gli autori trasferiscono il copyright a ISTQB®. Gli autori (in qualità di attuali titolari del copyright) e ISTQB® (in qualità di futuro titolare del copyright) hanno concordato le seguenti condizioni di utilizzo:

- È possibile copiare estratti di questo documento, per uso non commerciale, a condizione che ne venga riconosciuta la fonte. Qualsiasi Training Provider accreditato può utilizzare questo syllabus come base per un corso di formazione, a condizione che gli autori e ISTQB® siano riconosciuti come fonte e titolari del copyright del syllabus, e a condizione che qualsiasi pubblicità di tale corso di formazione possa menzionare il syllabus solo dopo che i materiali di formazione abbiano ricevuto l'accREDITAMENTO ufficiale da parte di un Member Board riconosciuto da ISTQB®.
- Qualsiasi individuo o gruppo di individui può utilizzare questo syllabus come base per articoli e libri, a condizione che gli autori e ISTQB® siano riconosciuti come fonte e titolari del copyright del syllabus.
- È proibito qualsiasi altro utilizzo di questo syllabus senza aver prima ottenuto l'approvazione scritta di ISTQB®.
- Qualsiasi Member Board riconosciuto da ISTQB® può tradurre questo syllabus, a condizione che riproduca il suddetto Avviso di Copyright nella versione tradotta del syllabus.

Cronologia delle Revisioni

Versione	Data	Osservazioni
1.0	01/10/2021	Rilascio per GA
2.0 Alpha	15/08/2025	Review Alpha
2.0 Beta	07/01/2026	Review Beta
2.0	17/04/2026	Rilascio per GA

Indice dei Contenuti

Avviso di Copyright	2
Cronologia delle Revisioni.....	3
Indice dei Contenuti	4
Ringraziamenti	8
0 Introduzione	9
0.1 Scopo di questo Syllabus	9
0.2 Il Certified Tester AI Testing	9
0.3 Percorso di Carriera per i Tester	9
0.4 Business Outcome	10
0.5 Obiettivi di Apprendimento, Obiettivi Pratici e Livello Cognitivo di Conoscenza	10
0.6 L'Esame per il Certificato Certified Tester AI Testing.....	11
0.7 Accreditamento	11
0.8 Gestione degli Standard	11
0.9 Livello di Dettaglio.....	11
0.10 Com'è Organizzato questo Syllabus	12
1 Introduzione all'Intelligenza Artificiale – 120 minuti.....	14
1.1 Introduzione all'AI	15
1.1.1 Sistemi Basati sull'AI e Sistemi Convenzionali	15
1.1.2 AI Ristretta, AI Generale e Super AI	15
1.1.3 Diversi Tipi di Tecnologie AI.....	16
1.1.4 AI Generativa.....	17
1.1.5 Hardware per i Sistemi di Machine Learning	18
1.1.6 Sviluppo e Hosting dei Modelli AI.....	19
1.1.7 Framework di Sviluppo del Machine Learning	19
1.1.8 Regolamenti e Standard per l'AI	20
2 Caratteristiche di Qualità per i Sistemi Basati sull'AI – 45 minuti.....	22
2.1 Caratteristiche di Qualità per i Sistemi Basati sull'AI	23
2.1.1 Caratteristiche di Qualità Specifiche dell'AI	23
2.1.2 AI e Sicurezza Fisica	24
2.2 Criteri di Accettazione per i Sistemi Basati sull'AI	25
2.2.1 Criteri di Accettazione per i Sistemi Basati sull'AI	25

3	Machine Learning – 375 minuti	27
3.1	Introduzione al Machine Learning	28
3.1.1	Diverse Forme di Machine Learning	28
3.1.2	Workflow del Machine Learning	29
3.1.3	Esercizio Pratico: Creare un Modello di Machine Learning	31
3.1.4	Modelli Preaddestrati, Fine-Tuning e Retrieval-Augmented Generation	31
3.2	Dati per il Machine Learning	32
3.2.1	Attività di Preparazione dei Dati	32
3.2.2	Esercizio Pratico: Preparazione dei Dati a Supporto della Creazione di un Modello di Machine Learning	33
3.2.3	Dataset di Addestramento, di Validazione e di Test	34
3.3	Metriche di Efficienza delle Prestazioni Funzionali di ML per la Classificazione	35
3.3.1	Calcolo delle Metriche di Efficienza delle Prestazioni Funzionali di Machine Learning	35
3.3.2	Esercizio Pratico: Valutare un Modello di Machine Learning utilizzando Metriche Selezionate di Efficienza delle Prestazioni Funzionali di ML	36
3.3.3	Esercizio Pratico: Mostrare l'Impatto di Diverse Combinazioni di Modelli di Machine Learning e Dataset	36
3.4	Reti Neurali	36
3.4.1	Struttura e Funzionamento di una Rete Neurale Profonda	37
3.4.2	Esercizio Pratico: Sperimentare l'Implementazione di un Perceptron	38
3.4.3	Misure di Copertura per le Reti Neurali	38
4	Testing dei Sistemi Basati sull'AI – 195 minuti	40
4.1	Introduzione al Testing dei Sistemi Basati sull'AI	41
4.1.1	Sistemi Basati sull'AI Bloccati e Adattivi	41
4.1.2	Motivazioni per un Approccio Statistico al Testing dei Sistemi Basati sull'AI	42
4.1.3	Oracoli del Test per i Sistemi Basati sull'AI	43
4.2	Testing dell'AI Generativa e dei Large Language Model	44
4.2.1	Testing dell'AI Generativa	44
4.2.2	Red Teaming	45
4.2.3	Esercizio Pratico: Testing Esplorativo di un Large Language Model	46
4.3	Livelli di Test e Sistemi di Machine Learning	46
4.3.1	Livelli di Test per i Sistemi di Machine Learning	46
4.3.2	Testing Basato sul Rischio dei Sistemi di Machine Learning	47
5	Testing dei Dati di Input per i Sistemi di Machine Learning – 180 minuti	48

5.1	Testing dei Dati di Input per i Sistemi di Machine Learning	49
5.1.1	Rischi Relativi ai Dati di Input e Relative Mitigazioni.....	49
5.1.2	Testing per Bias.....	50
5.1.3	Testing della Pipeline di Dati	51
5.1.4	Test di Rappresentatività dei Dati	52
5.1.5	Testing dei Vincoli di Dataset	53
5.1.6	Testing di Correttezza delle Etichette.....	54
5.1.7	Esercizio Pratico: Testing dei Dati di Input.....	55
6	Testing del Modello per i Sistemi di Machine Learning – 225 minuti	56
6.1	Testing del Modello per i Sistemi di Machine Learning	57
6.1.1	Rischi Relativi ai Modelli di Machine Learning e Relative Mitigazioni.....	57
6.1.2	Documentazione e Review del Modello di Machine Learning	58
6.1.3	Testing dell'Efficienza delle Prestazioni Funzionali di ML dei Sistemi Probabilistici di Machine Learning	59
6.1.4	Adversarial Testing dei Sistemi di Machine Learning	61
6.1.5	Testing Metamorfico	61
6.1.6	Esercizio Pratico: Applicare il Testing Metamorfico	62
6.1.7	Drift Testing	62
6.1.8	Testing per Overfitting e Underfitting	63
6.1.9	Testing A/B	63
6.1.10	Testing Back-to-Back	64
7	Testing dello Sviluppo del Machine Learning – 30 minuti	66
7.1	Testing dello Sviluppo del Machine Learning	67
7.1.1	Rischi Relativi allo Sviluppo del Machine Learning e Relative Mitigazioni.....	67
7.1.2	Testing del Deployment dei Sistemi di Machine Learning	68
8	Elenco delle Abbreviazioni	70
9	Termini Specifici dell'AI	72
10	Riferimenti	81
10.1	Standard	81
10.2	Documenti ISTQB®	81
10.3	Riferimenti al Glossario.....	81
10.4	Libri, Articoli e Pagine Web.....	82
11	Marchi Commerciali.....	83

12	Appendice A – Obiettivi di Apprendimento/Livello Cognitivo di Conoscenza	84
	Livello 1: Ricordare (K1)	84
	Livello 2: Comprendere (K2).....	84
	Livello 3: Applicare (K3).....	85
	Livello 4: Analizzare (K4).....	85
13	Appendice B – Matrice di Tracciabilità dei Business Outcome rispetto agli Obiettivi di Apprendimento	87
14	Appendice C – Release Note	97
15	Indice Analitico	98

Ringraziamenti

L'Assemblea Generale di ISTQB® ha pubblicato formalmente questo documento il 17 aprile 2026.

Task force AI di ISTQB (v2.0): Klaudia Dussa-Zieger (chair), Stuart Reid, Vipul Kocher, Qin Liu, Jaroslaw Hryszko, Kyle Alexander Siemens e Werner Henschelchen

Le seguenti persone hanno partecipato alla review del syllabus e hanno fornito commenti: Marina Abratis, Tom Adams, Laura Albert, Abhishek Alladi, Menno van den Berg, Earl Burba, Simeone Chiumarulo, Marco Ciarlito, Alessandro Collino, Jean-Baptiste Crouigneau, Yara Dalgamoni, Taz Daughtrey, Wim Decoutere, Dmitrii Degtiarenko, Iuliia Emelianova, Lozana Enbah, Tamás Gergely, David Hendrickx, David Janota, Sagar Joshi, Norbert Juhász, Willem Keesman, John Kurowski, Ine Lutterman, Niranjana Maharajh, Rik Marselis, Judy McKay, Gary Mogyorodi, Markus Niehammer, Tauhida Parveen, Arnd Pehl, Lukas Piska, Daniel Polan, Andrew Pollner, Nishan Portoyan, Meile Posthuma, Miroslav Renda, Randall Rice, Piet de Roo, Nicola de Rosa, Mark Rutz, Salvatore Sarno, Klaus Skafte, Giancarlo Tomasig, Yaron Tsubery, Rahul Verma, André Verschelling, Linda Vreeswijk, Mario Winter

Task force AI di ISTQB (v1.0): Klaudia Dussa-Zieger (chair), Vipul Kocher, Qin Liu, Stuart Reid, Adam Leon Smith, Kyle Alexander Siemens e Werner Henschelchen

Il team ringrazia gli autori dei tre syllabus che hanno contribuito:

- A4Q: Rex Black, Bruno Legeard, Jeremias Rößler, Adam Leon Smith, Stephan Goericke, Werner Henschelchen
- AiU: autori principali Vipul Kocher, Saurabh Bansal, Srinivas Padmanabhuni e Sonika Bengani e coautori Rik Marselis, José M. Diaz Delgado
- CSTQB/KSTQB: Qin Liu, Stuart Reid

Il team ringrazia gli Exam, Glossary e Marketing Working Group per il supporto fornito durante tutto lo sviluppo del syllabus, e i Member Board per i suggerimenti e i contributi.

Traduzione italiana a cura di ITA-STQB (www.ita-stqb.org) – info@ita-stqb.org

0 Introduzione

0.1 Scopo di questo Syllabus

Questo syllabus costituisce la base per la certificazione ISTQB® Certified Tester AI Testing. ISTQB® fornisce questo syllabus come segue:

- Ai Member Board, per la traduzione nella loro lingua locale e per l'accreditamento dei Training Provider. I Member Board possono adattare il syllabus alle loro specifiche esigenze linguistiche e modificare i riferimenti per allinearli alle proprie pubblicazioni locali.
- Agli organismi di certificazione, per ricavare domande d'esame nella loro lingua locale, adatte agli obiettivi di apprendimento di questo syllabus.
- Ai Training Provider, per produrre materiale didattico e determinare i metodi di insegnamento appropriati.
- Ai candidati alla certificazione, per prepararsi all'esame di certificazione (sia come parte di un corso di formazione, sia in modo indipendente).
- Alla comunità internazionale del software e systems engineering, per promuovere la professione del testing del software e dei sistemi e come base per libri e articoli.

0.2 Il Certified Tester AI Testing

La certificazione Certified Tester AI Testing (CT-AI) è destinata a chi è coinvolto nel testing dei sistemi basati sull'AI. Vi rientrano persone con ruoli diversi, quali tester, test analyst, data analyst, test engineer, test consultant, test manager, user acceptance tester e sviluppatori software. Questa certificazione è adatta anche a chi desidera acquisire una comprensione di base del testing dei sistemi basati sull'AI, come project manager, quality manager, responsabili dello sviluppo software, business analyst, membri dei team di operations, direttori IT e consulenti di management.

0.3 Percorso di Carriera per i Tester

Lo schema ISTQB® supporta i professionisti del testing in tutte le fasi della loro carriera. Chi consegue la certificazione ISTQB® Certified Tester Foundation Level può essere interessato anche ai Core Advanced Level (Test Analyst, Technical Test Analyst e Test Manager) e, successivamente, all'Expert Level (Test Management o Improving the Test Process). Il percorso Specialist consente di approfondire specifici approcci e attività di test, ad esempio Agile Testing, Test Automation, AI Testing, Testing with Generative AI o Mobile App Testing, oppure approfondire conoscenze di testing relative a specifici settori industriali, ad esempio Automotive o Gaming. Per le informazioni più aggiornate sull'ISTQB® Certified Tester Scheme, visitare www.istqb.org.

0.4 Business Outcome

Questa sezione elenca i Business Outcome attesi da un candidato che ha conseguito la certificazione CT-AI.

Un Certified Tester AI Testing è in grado di:

BO1	Comprendere lo stato attuale dell'AI, inclusa l'AI generativa.
BO2	Sperimentare l'implementazione e il testing di modelli di machine learning.
BO3	Comprendere il funzionamento e il testing di reti neurali semplici.
BO4	Comprendere le caratteristiche di qualità specifiche dell'AI definite dalla norma ISO/IEC 25059.
BO5	Calcolare e interpretare le metriche di efficienza delle prestazioni funzionali di ML per i modelli di machine learning.
BO6	Riconoscere l'ambito e l'importanza dei due livelli di test specifici del testing dei sistemi di machine learning.
BO7	Contribuire allo sviluppo di una strategia di test efficace per un sistema di machine learning.
BO8	Progettare ed eseguire test case per i sistemi di machine learning.

0.5 Obiettivi di Apprendimento, Obiettivi Pratici e Livello Cognitivo di Conoscenza

Gli Obiettivi di Apprendimento (LO, Learning Objective) supportano i Business Outcome e vengono utilizzati per creare gli esami CT-AI. I livelli degli specifici obiettivi di apprendimento sono indicati all'inizio di ogni capitolo e classificati come segue:

- K1: Ricordare
- K2: Comprendere
- K3: Applicare
- K4: Analizzare

Ulteriori dettagli ed esempi di obiettivi di apprendimento sono riportati nell'Appendice A.

Per tutti i termini elencati come parole chiave subito sotto i titoli dei capitoli, è necessario ricordare (K1) la denominazione corretta e la definizione riportate nel Glossario ISTQB® o nel Capitolo 9, anche se non sono menzionati esplicitamente nell'obiettivo di apprendimento.

Gli Obiettivi Pratici (HO, Hands-on Objective) si focalizzano sull'applicazione pratica degli Obiettivi di Apprendimento e sono indicati all'inizio di ogni capitolo. Il livello di un HO è classificato come segue:

- H0: può comprendere una dimostrazione dal vivo di un esercizio o un video registrato. Poiché il partecipante non la esegue personalmente, non si tratta propriamente di un esercizio.
- H1: esercizio guidato. I partecipanti seguono una sequenza di passi eseguiti dal docente.
- H2: esercizio con suggerimenti. Al partecipante viene assegnato un esercizio corredato di suggerimenti pertinenti, che consentono di risolverlo entro il tempo previsto.

0.6 L'Esame per il Certificato Certified Tester AI Testing

L'esame CT-AI sarà basato sugli Obiettivi di Apprendimento descritti in questo syllabus. Tutte le sezioni del syllabus sono oggetto d'esame, ad eccezione dell'Introduzione, degli Obiettivi Pratici, dei Riferimenti e delle Appendici. Per rispondere alle domande d'esame può essere necessario utilizzare materiale tratto da più di una sezione di questo syllabus. Gli standard e i libri sono inclusi come riferimenti, ma il loro contenuto non è oggetto d'esame oltre quanto sintetizzato nel syllabus stesso.

Per ulteriori dettagli, fare riferimento al documento Exam Structures and Rules relativo a CT-AI v2.0.

Il principale criterio di ammissione per chiunque desideri sostenere l'esame CT-AI è il possesso del certificato ISTQB® Certified Tester Foundation Level [CTFL].

Ai candidati si raccomanda inoltre vivamente di:

- Avere almeno una conoscenza di base dello sviluppo software o del testing del software, ad esempio sei mesi di esperienza come tester di sistema o user acceptance tester, data scientist o sviluppatore software.
- Frequentare un corso accreditato secondo gli standard ISTQB® (presso uno dei Member Board riconosciuti da ISTQB).

0.7 Accreditamento

Un Member Board ISTQB® può accreditare i Training Provider il cui materiale didattico segue questo syllabus. I Training Provider dovrebbero ottenere le linee guida per l'accreditamento dal Member Board o dall'organismo che effettua l'accreditamento. Un corso accreditato è riconosciuto come conforme a questo syllabus e consente di includere nel corso un esame ISTQB®.

Le linee guida per l'accreditamento di questo syllabus seguono le Accreditation Guidelines generali pubblicate dal Processes Management and Compliance Working Group.

0.8 Gestione degli Standard

Organizzazioni internazionali di standardizzazione quali IEEE e ISO hanno pubblicato standard relativi alle caratteristiche di qualità e al testing del software. Lo scopo di questi riferimenti è fornire un quadro di riferimento (come nei rimandi alle norme ISO/IEC 25059 e ISO/IEC 25010 relative a un modello di qualità per i sistemi basati sull'AI) oppure, se il lettore lo desidera, una fonte di informazioni aggiuntive. Si noti che i syllabus utilizzano i documenti di standard come riferimento. I documenti di standard non sono oggetto d'esame.

0.9 Livello di Dettaglio

Il livello di dettaglio di questo syllabus consente di realizzare corsi ed esami coerenti a livello internazionale. Per raggiungere questo obiettivo, il syllabus comprende:

- Obiettivi formativi generali che descrivono l'intento del syllabus CT-AI
- Un elenco di parole chiave che gli studenti devono essere in grado di ricordare
- Obiettivi di apprendimento per ciascuna area di conoscenza, che descrivono il risultato cognitivo di apprendimento da conseguire

- Una descrizione dei concetti chiave, con riferimenti a fonti quali letteratura riconosciuta o standard

Il contenuto del syllabus non descrive l'intera area di conoscenza del testing del software: riflette il livello di dettaglio da trattare nei corsi di formazione CT-AI. Si focalizza sull'introduzione ai concetti fondamentali dell'AI e, in particolare, del machine learning (ML), nonché sul modo in cui è possibile testare i sistemi basati su queste tecnologie.

Il syllabus utilizza la terminologia (ossia la denominazione e il significato) dei termini impiegati nel testing del software e nella Quality Assurance secondo il Glossario ISTQB®.

0.10 Com'è Organizzato questo Syllabus

Sono presenti sette capitoli con contenuti oggetto d'esame. L'intestazione di primo livello di ciascun capitolo ne specifica la durata; non sono indicate tempistiche al di sotto del livello di capitolo. *Per i corsi di formazione accreditati, il syllabus richiede un minimo di 19,5 ore di istruzione, distribuite nei sette capitoli come segue:*

- Capitolo 1: 120 minuti - Introduzione all'Intelligenza Artificiale (AI)
 - Comprendere le differenze fondamentali tra i sistemi basati sull'AI e i sistemi convenzionali ed esplorare lo spettro delle capacità dell'AI, dall'AI ristretta alla super AI.
 - Acquisire una conoscenza di base delle tecnologie AI, inclusa l'AI generativa (GenAI), e dell'hardware, dell'hosting e dei framework di sviluppo utilizzati per implementare i sistemi di machine learning (MLS).
 - Apprendere in che modo regolamenti e standard influenzano lo sviluppo e il testing delle soluzioni basate sull'AI.
- Capitolo 2: 45 minuti - Caratteristiche di Qualità per i Sistemi Basati sull'AI
 - Apprendere le caratteristiche di qualità specifiche dei sistemi basati sull'AI, comprese quelle definite nella norma ISO/IEC 25059, e comprendere le considerazioni relative alla sicurezza fisica nell'impiego dell'AI in sistemi critici.
 - Esaminare come definire criteri di accettazione appropriati, adattati al comportamento e alle prestazioni peculiari delle soluzioni basate sull'AI.
- Capitolo 3: 375 minuti - Machine Learning
 - Comprendere i tipi di ML, i passi principali del workflow di sviluppo del ML e il modo in cui i modelli preaddestrati, il fine-tuning e la Retrieval-Augmented Generation contribuiscono ai moderni sistemi basati sull'AI.
 - Apprendere la preparazione dei dati, i ruoli dei dataset di addestramento, di validazione e di test e il modo in cui questi influenzano lo sviluppo e le prestazioni dei modelli ML.
 - Esaminare le reti neurali, incluse la loro struttura e le relative misure di copertura, e acquisire esperienza pratica con le metriche delle prestazioni e con l'utilizzo di una matrice di confusione.
- Capitolo 4: 195 minuti - Testing dei Sistemi Basati sull'AI

- Comprendere le sfide specifiche del testing dei sistemi basati sull'AI, incluse le differenze di testabilità tra sistemi bloccati e adattivi, la necessità di un testing statistico e le difficoltà nel definire gli oracoli del test.
- Apprendere come testare la GenAI e i Large Language Model (LLM), utilizzando tecniche quali il red teaming e il testing esplorativo per l'AI che svolge attività di test.
- Esaminare le strategie di test per gli MLS, considerando i diversi livelli di test e l'applicazione del testing basato sul rischio.
- Capitolo 5: 180 minuti - Testing dei Dati di Input per i Sistemi di Machine Learning
 - Apprendere come testare e validare i dati di input per gli MLS, incluse le tecniche per rilevare i bias, verificare la correttezza delle etichette, valutare la rappresentatività dei dati ed eseguire il testing della pipeline di dati.
- Capitolo 6: 225 minuti - Testing del Modello per i Sistemi di Machine Learning
 - Scoprire gli approcci di test per mitigare i rischi nei modelli ML, incluse le review della documentazione, il testing dell'efficienza delle prestazioni funzionali di ML e il rilevamento di overfitting, underfitting e deriva.
 - Apprendere approcci di test avanzati, inclusi l'adversarial testing, il testing A/B, il testing back-to-back e l'utilizzo di attacchi per individuare i punti deboli del modello.
 - Acquisire una comprensione pratica del testing metamorfico, incluso il modo in cui derivare e applicare test case quando gli oracoli del test tradizionali sono insufficienti.
- Capitolo 7: 30 minuti - Testing dello Sviluppo del Machine Learning
 - Apprendere approcci e strategie di test per mitigare i rischi durante lo sviluppo e il deployment degli MLS, al fine di supportare un comportamento robusto del sistema negli ambienti di produzione.

1 Introduzione all'Intelligenza Artificiale – 120 minuti

Parole Chiave

Nessuna

Parole Chiave Specifiche dell'AI

sistema basato sull'AI, intelligenza artificiale, AI generale, machine learning, framework di sviluppo ML, AI ristretta, super AI

Obiettivi di Apprendimento per il Capitolo 1:

1.1 Introduzione all'AI

AI-1.1.1 (K2) Differenziare i sistemi basati sull'AI dai sistemi convenzionali

AI-1.1.2 (K2) Distinguere tra AI ristretta, AI generale e super AI

AI-1.1.3 (K2) Spiegare i diversi tipi di tecnologie AI

AI-1.1.4 (K2) Spiegare l'AI generativa

AI-1.1.5 (K2) Confrontare le opzioni hardware disponibili per implementare sistemi di machine learning

AI-1.1.6 (K2) Confrontare le opzioni per lo sviluppo e l'hosting dei modelli AI

AI-1.1.7 (K2) Riassumere le funzionalità fornite dai framework di sviluppo ML

AI-1.1.8 (K2) Spiegare in che modo regolamenti e standard influenzano lo sviluppo e il testing dei sistemi basati sull'AI

1.1 Introduzione all'AI

Questo capitolo presenta le differenze fondamentali tra i sistemi convenzionali e i sistemi basati sull'AI, con particolare attenzione agli approcci di progettazione, all'adattabilità e alla spiegabilità. Descrive lo spettro delle capacità dell'AI, dall'AI ristretta all'AI generale e alla super AI, e illustra tecnologie fondamentali quali il machine learning, il deep learning e la GenAI. Il capitolo descrive inoltre l'hardware e gli ambienti di sviluppo comunemente utilizzati per i sistemi basati sull'AI, nonché i framework ML più diffusi. Infine, prende in esame i principali standard normativi e tecnici che guidano lo sviluppo e il deployment responsabili dell'AI nei diversi domini.

1.1.1 Sistemi Basati sull'AI e Sistemi Convenzionali

I sistemi informatici convenzionali sono generalmente programmati utilizzando linguaggi imperativi, nei quali gli sviluppatori definiscono esplicitamente istruzioni passo passo, incluse strutture quali le istruzioni if-then-else e i cicli. Questo approccio deterministico contribuisce a rendere il comportamento del sistema prevedibile e trasparente, facilitando la comprensione del modo in cui gli input producono output differenti. Al contrario, la maggior parte dei sistemi basati sull'AI, in particolare quelli che utilizzano il machine learning, non segue regole predefinite. Analizza invece i pattern presenti nei dati per determinare come rispondere a nuovi input. Ad esempio, un sistema di riconoscimento delle immagini basato sull'AI e addestrato a identificare i gatti non si basa su regole codificate esplicitamente. Apprende invece da un dataset di immagini di gatti, estraendo pattern che applica successivamente a immagini mai viste per classificarle correttamente.

Una differenza fondamentale tra i sistemi convenzionali e i sistemi basati sull'AI riguarda il modo in cui affrontano la risoluzione dei problemi. Molti sistemi basati sull'AI si avvalgono del ragionamento probabilistico, dell'inferenza statistica e del riconoscimento di pattern per generare risultati. Ciò consente ai modelli AI di gestire più efficacemente forme complesse di incertezza e ambiguità, producendo output che non sono sempre prevedibili.

Una sfida fondamentale dei sistemi basati sull'AI è la spiegabilità. Molti modelli AI, in particolare le architetture di deep learning, possono contenere miliardi di parametri, il che rende difficile per le persone interpretarne il funzionamento interno. Questa natura di "scatola nera" desta preoccupazione in domini critici quali la sanità, la finanza, la difesa e i trasporti, nei quali è essenziale comprendere perché un modello AI abbia preso una determinata decisione. Il conseguimento della trasparenza (vedi 2.1.1) e della spiegabilità nei processi decisionali guidati dall'AI è diventato un punto centrale della regolamentazione dell'AI.

Un'altra differenza significativa riguarda l'adattabilità. I sistemi convenzionali sono statici e richiedono generalmente aggiornamenti manuali per incorporare nuove conoscenze o rispondere ai cambiamenti dell'ambiente. I sistemi basati sull'AI, al contrario, possono essere autoapprendenti e migliorare continuamente le proprie prestazioni man mano che incontrano nuovi dati. Questa adattabilità rende l'AI particolarmente efficace negli ambienti dinamici. L'adattabilità richiede inoltre un monitoraggio continuo per mantenere nel tempo l'allineamento con i requisiti fondamentali.

1.1.2 AI Ristretta, AI Generale e Super AI

L'AI ristretta, nota anche come AI debole, è progettata per svolgere attività specifiche e comprende tutti i sistemi AI oggi in uso. I sistemi basati sull'AI ristretta operano in un dominio limitato e possono essere molto efficienti nel risolvere problemi specialistici, quali il riconoscimento delle immagini, l'elaborazione del parlato e la traduzione linguistica. Tuttavia, non sono in grado di generalizzare oltre le funzioni

apprese. Ad esempio, un sistema basato sull'AI che eccelle nel riconoscimento dei volti non è in grado di eseguire una traduzione linguistica, a meno che non venga esplicitamente riaddestrato a tale scopo. La frontier AI, un sottoinsieme dell'AI ristretta, rappresenta la forma più avanzata di questi sistemi e ne amplia i limiti attuali mediante modelli GenAI. La frontier AI comprende sistemi su larga scala dotati di capacità decisionali altamente autonome; tali sistemi restano tuttavia specifici per determinate attività e non hanno ancora raggiunto la versatilità dell'AI generale.

L'AI generale, nota anche come AI forte, indica un sistema basato sull'AI capace di svolgere la maggior parte delle attività intellettuali che una persona è in grado di eseguire. L'AI generale sarebbe in grado di comprendere, apprendere e applicare conoscenze a un'ampia gamma di attività senza dover essere riaddestrata per ogni nuova attività. Questo tipo di AI mostrerebbe capacità di ragionamento e adattabilità simili a quelle umane e sarebbe in grado di risolvere problemi sconosciuti in domini diversi, analogamente alle persone. Nonostante i notevoli progressi dell'AI, oggi nessun sistema basato sull'AI possiede un'intelligenza generale.

La super AI (superintelligenza artificiale) è la forma di AI nella quale un sistema basato sull'AI migliora continuamente sé stesso senza la necessità dell'intervento o del controllo umano. Perché la super AI sia possibile, l'accesso a Internet non è strettamente necessario, ma tale accesso potrebbe ampliarne in misura significativa le capacità e l'influenza. Essa supererebbe l'intelligenza umana e l'AI generale e, secondo molti, rappresenterebbe un rischio esistenziale per l'umanità. Il punto in cui i sistemi basati sull'AI passerebbero dall'AI generale alla super AI, qualora ciò avvenisse, è comunemente denominato singolarità tecnologica.

1.1.3 Diversi Tipi di Tecnologie AI

L'intelligenza artificiale comprende una gamma di tecnologie, ciascuna adatta ad attività e sfide specifiche. Uno dei rami fondamentali dell'AI è il machine learning, che consente ai sistemi di apprendere dai dati e di costruire modelli senza una programmazione esplicita. Mentre alcuni MLS possono adattarsi e migliorare continuamente grazie a nuovi dati per tutta la durata del loro ciclo di vita, altri operano sulla base delle conoscenze inizialmente apprese e richiedono un riaddestramento esplicito per aggiornare le proprie capacità. Il machine learning comprende diversi approcci:

- L'apprendimento supervisionato utilizza dati etichettati e algoritmi, quali la regressione lineare e gli alberi decisionali, per attività come la previsione e la classificazione.
- L'apprendimento non supervisionato individua pattern nei dati non etichettati mediante tecniche quali il clustering, che si avvale di algoritmi di clustering.
- L'apprendimento per rinforzo consente agli agenti intelligenti di apprendere comportamenti ottimali attraverso interazioni per tentativi ed errori con il proprio ambiente.

Le principali tecnologie ML comprendono le reti neurali, i modelli bayesiani, le support vector machine (SVM) e le random forest. Per ulteriori informazioni sul machine learning, vedi il Capitolo 3.

Il deep learning (DL), un sottoinsieme dei metodi di machine learning, utilizza reti neurali profonde per risolvere problemi complessi. Ad esempio:

- Le reti neurali convoluzionali (CNN) sono particolarmente efficaci per il riconoscimento delle immagini e il rilevamento degli oggetti.
- Le reti neurali ricorrenti (RNN) sono specializzate nell'elaborazione di dati sequenziali, quali testo o serie temporali.

- I transformer gestiscono le dipendenze a lungo raggio nelle sequenze e costituiscono la base dei modelli per l'elaborazione del linguaggio naturale e dei vision transformer per le immagini.

La GenAI si basa su queste tecnologie per creare nuovi contenuti, inclusi testo, immagini e audio (vedi 1.1.4). Ciò è reso possibile da modelli quali gli LLM, che combinano reti neurali profonde (DNN) ed elaborazione del linguaggio naturale (NLP) per analizzare e generare un linguaggio simile a quello umano.

Altre tecnologie AI specialistiche comprendono:

- L'NLP per l'analisi del linguaggio, incluse attività quali l'analisi del sentiment e la traduzione automatica.
- La computer vision per analizzare i dati visivi, a supporto di applicazioni quali il riconoscimento facciale e la robotica.
- La logica fuzzy per il ragionamento in condizioni di incertezza.
- Gli algoritmi di ricerca per risolvere problemi di ottimizzazione, quali la navigazione, e per il processo decisionale strategico.
- I sistemi di ragionamento basati su regole, o sistemi esperti, per il supporto strutturato alle decisioni.

Sebbene la piena integrazione di queste tecnologie AI resti limitata, nuovi sviluppi quali gli LLM dimostrano la possibilità di combinare tecnologie AI differenti in sistemi intelligenti unificati. L'AI agantica estende tali tecnologie mediante agenti autonomi che pianificano, ragionano e agiscono in modo indipendente per raggiungere obiettivi in ambienti dinamici.

1.1.4 AI Generativa

L'AI generativa (GenAI) indica sistemi basati sull'AI specializzati nella creazione di nuovi contenuti, quali testo, immagini, video, musica o dati complessi; molti di essi supportano anche la classificazione e la previsione. Questi sistemi apprendono da grandi quantità di dati per produrre output simili ai propri dati di addestramento, rendendo possibile un'ampia gamma di applicazioni creative e pratiche.

Le principali tecnologie alla base della GenAI comprendono le reti generative avversarie (GAN), i modelli di diffusione e i transformer. Le GAN utilizzano due reti neurali in competizione per creare dati sintetici altamente realistici. I modelli di diffusione generano contenuti aggiungendo gradualmente rumore ai dati e rimuovendolo successivamente, producendo output di elevata qualità. I modelli transformer, sui quali si basano gli LLM, utilizzano meccanismi di self-attention per generare testo coerente e pertinente al contesto e vengono sempre più spesso adattati ad attività multimodali.

Al di là dei loro fondamenti tecnici, i sistemi GenAI sollevano importanti questioni sociali ed etiche. L'uso improprio costituisce una preoccupazione rilevante: questi modelli possono essere sfruttati per creare deepfake, diffondere disinformazione o generare contenuti fraudolenti convincenti, compromettendo così la fiducia nei media digitali e nel dibattito pubblico. La facilità con cui è possibile produrre contenuti sintetici amplifica i rischi relativi a privacy, sicurezza e manipolazione, ma può portare notevoli benefici e aprire opportunità di innovazione nell'intrattenimento, nel marketing, nell'istruzione e nella ricerca.

Anche l'impatto sull'occupazione è significativo, in particolare per le professioni impiegatizie e intellettuali. Poiché la GenAI automatizza attività quali la scrittura, la progettazione, la codifica e persino la produzione di documentazione giuridica o medica, cresce il dibattito sulla possibile perdita di posti di lavoro. Sebbene queste tecnologie possano accrescere la produttività e favorire nuove opportunità creative, possono

anche provocare trasformazioni nel mercato del lavoro e rendere necessaria una riqualificazione professionale su vasta scala.

La sostenibilità è un'altra questione urgente. L'addestramento e l'esecuzione di grandi modelli GenAI richiedono notevoli risorse computazionali, con conseguenti elevati consumi energetici e una significativa impronta di carbonio. Questo impatto ambientale ha generato richieste di progettazioni dei modelli più efficienti e di infrastrutture più sostenibili.

La maggior parte degli strumenti GenAI oggi utilizzati si basa su foundation model, successivamente sottoposti a fine-tuning per applicazioni specifiche. Il settore sta inoltre progredendo verso modelli multimodali in grado di elaborare e generare contenuti testuali, visivi e audio, rendendo possibili sistemi basati sull'AI più ricchi e flessibili. Stanno emergendo quadri normativi, quali l'EU AI Act [EU AI Act], per guidare lo sviluppo e l'utilizzo responsabili di queste tecnologie.

1.1.5 Hardware per i Sistemi di Machine Learning

Per gli MLS viene utilizzata una varietà di hardware. Per l'addestramento e per l'inferenza possono essere impiegati tipi di hardware differenti. Ad esempio, un modello di riconoscimento vocale può essere eseguito su uno smartphone di fascia bassa, sebbene per addestrarlo possa essere necessario accedere alla potenza del cloud computing.

Il machine learning trae generalmente vantaggio da hardware che supporta quanto segue:

- La capacità di operare con strutture di dati di grandi dimensioni.
- L'elaborazione massicciamente parallela (concorrente), ad esempio per supportare la moltiplicazione di matrici.
- Aritmetica a bassa precisione (quantizzazione): questo approccio utilizza un numero minore di bit per i calcoli (ad esempio 4 bit anziché 32), consentendo un'elaborazione più rapida, un consumo energetico inferiore, chip più piccoli ed economici e requisiti di larghezza di banda ridotti.

Le unità centrali di elaborazione (CPU) per uso generale supportano operazioni complesse ad alta precisione, che normalmente non sono necessarie per le applicazioni ML, ma dispongono in genere di pochi core. Di conseguenza, la loro architettura è meno efficiente per l'addestramento e l'esecuzione dei modelli ML rispetto alle unità di elaborazione grafica (GPU), che dispongono di migliaia di core e sono progettate per eseguire elaborazioni grafiche massicciamente parallele, ma relativamente semplici. Di conseguenza, nelle applicazioni ML le GPU superano generalmente le CPU in termini di prestazioni, anche se le CPU operano solitamente a frequenze di clock più elevate. Per attività ML di piccola scala, le GPU rappresentano in genere l'opzione migliore.

Alcuni hardware sono progettati specificamente per l'AI, come i circuiti integrati specifici per l'applicazione (ASIC) e i dispositivi System-on-a-Chip. Queste soluzioni specifiche per l'AI dispongono di più core, di una gestione specializzata dei dati e della capacità di eseguire elaborazioni in memoria. Sono particolarmente adatte all'edge computing, mentre l'addestramento del modello ML viene eseguito nel cloud mediante hardware specializzato.

Le architetture hardware specifiche per l'AI continuano a evolvere. Tra queste rientrano i processori neuromorfici, che non utilizzano la tradizionale architettura di von Neumann, bensì strutture ispirate al cervello che imitano quelle neuronali.

1.1.6 Sviluppo e Hosting dei Modelli AI

I sistemi basati sull'AI possono essere acquisiti da fornitori terzi oppure sviluppati privatamente all'interno di un'organizzazione. Generalmente si basano su modelli preaddestrati e possono essere distribuiti on-premises o nel cloud; anche i modelli stessi possono essere ospitati on-premises o nel cloud, dove le opzioni cloud sono spesso accessibili come servizio (AlaaS). I sistemi basati sull'AI di terze parti sono in genere forniti come modelli preaddestrati o come AI as a Service (AlaaS), consentendo un deployment più rapido e un time-to-market più breve. In alternativa, i sistemi basati sull'AI privati (on-premises o configurazioni cloud personalizzate) possono essere adattati meglio a requisiti specifici, ma il loro sviluppo richiederà probabilmente competenze specialistiche, fornite da esperti interni o da team esterni.

Lo sviluppo locale (codifica e addestramento eseguiti localmente) consente controllo diretto e privacy. I modelli di piccole dimensioni, quali gli alberi decisionali o le reti neurali compatte, possono essere sviluppati su personal computer, mentre i modelli di medie dimensioni possono richiedere GPU dedicate. Per i modelli su larga scala diventano necessari cluster di server on-premises ad alte prestazioni, con i relativi costi energetici, di raffreddamento e di hardware.

Lo sviluppo nel cloud offre una notevole flessibilità. In particolare, i cloud pubblici forniscono ambienti preconfigurati con tariffe pay-as-you-go, che limitano l'investimento hardware iniziale e consentono di scalare facilmente. I cloud privati, invece, possono offrire maggiore sicurezza e privacy alle applicazioni che le richiedono, ma questo controllo comporta un investimento iniziale più elevato in infrastruttura.

Molte organizzazioni adottano approcci ibridi, ad esempio sviluppando localmente i prototipi prima di passare a un'infrastruttura cloud, mantenendo on-premises i componenti sensibili, come la preparazione dei dati privati, e utilizzando le risorse cloud per le attività ad alta intensità computazionale.

I modelli AI possono essere ospitati in ambienti diversi, dalle configurazioni locali alle piattaforme basate sul cloud. L'hosting locale comporta l'esecuzione di modelli più piccoli su personal computer o smartphone, garantendo privacy ed eliminando i costi delle licenze cloud, sebbene offra capacità hardware limitate. Per i modelli AI di maggiori dimensioni, le organizzazioni possono predisporre server dedicati, che richiedono un investimento iniziale significativo ma assicurano un controllo maggiore.

L'hosting dei modelli AI nel cloud può avvenire su cloud pubblici o privati. I servizi di cloud pubblico forniscono un accesso scalabile a infrastrutture solide e potenti, eliminano le problematiche di manutenzione e sono quindi ideali per carichi di lavoro variabili. I cloud privati offrono vantaggi analoghi, con maggiori opzioni di sicurezza e personalizzazione, e sono gestiti internamente o da fornitori dedicati, sebbene a un costo superiore.

Gli approcci ibridi combinano questi metodi, consentendo alle organizzazioni di eseguire localmente alcune operazioni e di sfruttare l'elasticità del cloud per le attività intensive.

Le soluzioni ottimali di sviluppo e di hosting, che vengono generalmente decise separatamente, dipendono da fattori quali le dimensioni e la complessità del modello, i requisiti prestazionali, i vincoli di budget, le considerazioni relative a sicurezza e data privacy, le esigenze di deployment e i requisiti normativi. Alcune organizzazioni adottano strategie multilivello per bilanciare efficienza e controllo.

1.1.7 Framework di Sviluppo del Machine Learning

I framework di sviluppo ML forniscono un insieme di strumenti per costruire e addestrare modelli ML. Le funzionalità tipicamente offerte da questi framework comprendono:

- Gestione dei dati: supportano il caricamento, la pre-elaborazione e la gestione dei dati utilizzati per addestrare e testare il modello. Ciò può comportare la pulizia, la formattazione e la trasformazione dei dati in un formato adatto al modello scelto.

- **Costruzione del modello:** questi framework offrono librerie di algoritmi ML e strumenti per progettare l'architettura del modello ML costruito. Ciò comprende la specifica del tipo di modello (ad esempio rete neurale, albero decisionale), del numero di livelli e di connessioni e delle operazioni matematiche eseguite all'interno del modello.
- **Addestramento e ottimizzazione:** i framework forniscono algoritmi che regolano iterativamente i parametri interni del modello sulla base dei dati di addestramento e del risultato desiderato. L'obiettivo è ottimizzare le prestazioni del modello nello svolgimento dell'attività prevista (ad esempio classificazione, regressione ML). Alcuni framework possono supportare l'addestramento distribuito e migliorare o sottoporre a fine-tuning modelli preaddestrati.
- **Valutazione:** offrono strumenti per valutare le prestazioni del modello addestrato su dati mai visti. Ciò può comportare la misurazione di accuratezza, precision e recall per le attività di classificazione, oppure dei tassi di errore per le attività di regressione ML (vedi 3.1.1).
- **Deployment:** alcuni framework forniscono funzionalità per eseguire il deployment del modello addestrato e utilizzarlo in contesti reali. Ciò può comportare la conversione del modello in un formato adatto all'integrazione con applicazioni web, dispositivi mobili, dispositivi edge o sistemi embedded.

Questi framework possono operare a diversi livelli di astrazione. Alcuni offrono un'Application Program Interface (API) di basso livello, fornendo agli sviluppatori un maggiore controllo sulla costruzione del modello ma richiedendo competenze di codifica più avanzate. Altri offrono un'API di livello più elevato, semplificando la creazione del modello ma mettendo a disposizione meno opzioni di personalizzazione.

I diversi framework possono essere orientati a domini applicativi differenti. Alcuni sono general purpose e supportano un'ampia gamma di aree applicative. Altri, invece, sono più specializzati e si concentrano su ambiti specifici quali il riconoscimento delle immagini, il riconoscimento vocale e la traduzione linguistica.

La selezione del framework più appropriato può dipendere da diversi fattori, quali:

- l'area applicativa;
- la necessità di un'interfaccia intuitiva per la prototipazione rapida;
- la configurabilità per modelli complessi;
- le competenze degli utenti;
- le considerazioni relative al deployment, poiché alcuni framework sono più adatti ad ambienti con risorse limitate;
- il livello di supporto, anche da parte della comunità;
- la maturità dell'ecosistema.

1.1.8 Regolamenti e Standard per l'AI

I regolamenti e gli standard relativi all'AI sono fondamentali per uno sviluppo, un deployment e un utilizzo responsabili dell'AI. L'obiettivo principale è favorire la fiducia nell'AI e promuovere la realizzazione dei suoi benefici, mitigando al contempo i potenziali danni. Idealmente, la conformità a tali regolamenti e standard dovrebbe garantire che i sistemi basati sull'AI siano sicuri, equi, trasparenti, sostenibili, responsabili ed etici, e che vengano utilizzati in modo responsabile.

A livello internazionale, i Principi dell'OCSE sull'AI [OECD AI] e il rapporto delle Nazioni Unite Governing AI for Humanity [UN Gov AI] costituiscono influenti strumenti di soft law che favoriscono una

comprensione condivisa della gestione responsabile dell'AI. Fungono da bussola per i governi nazionali e per le organizzazioni nella definizione delle proprie strategie per l'AI. Questi principi pongono l'accento su un'AI antropocentrica, sulle considerazioni etiche e sull'importanza della cooperazione internazionale.

L'EU AI Act rappresenta una tappa normativa fondamentale e adotta un approccio basato sul rischio alla regolamentazione dell'AI. Classificando i sistemi basati sull'AI in base al rischio, da minimo a inaccettabile, la regolamentazione viene calibrata di conseguenza. I sistemi ad alto rischio, in particolare quelli che incidono sui diritti fondamentali o sulla sicurezza fisica, sono soggetti a requisiti rigorosi che comprendono testing approfondito, governance dei dati e supervisione umana. Le consistenti sanzioni pecuniarie per la mancata conformità, calcolate come percentuale del fatturato globale, sottolineano l'impegno dell'UE nell'applicazione della normativa. Al contrario, molti Paesi al di fuori dell'UE stanno adottando un approccio più permissivo, privilegiando una regolamentazione meno onerosa per incoraggiare l'innovazione.

Gli standard tecnici, sviluppati da organizzazioni quali ISO e IEEE, sono fondamentali per tradurre gli obiettivi di alto livello in implementazioni pratiche. Forniscono specifiche tecniche concrete e best practice, colmando il divario tra le politiche e la pratica. Ad esempio, ISO/IEC TR 29119-11 fornisce indicazioni dettagliate sul testing dei sistemi basati sull'AI, elemento essenziale per dimostrare la conformità normativa. Nel frattempo, la serie ISO/IEC 42119 è in fase di sviluppo per coprire diversi aspetti del testing dei sistemi basati sull'AI. Stanno inoltre emergendo regolamenti specifici di settore in ambiti quali la sanità e la finanza, che riconoscono i rischi peculiari posti dall'AI in questi domini.

Per orientarsi efficacemente in questo panorama in evoluzione della governance dell'AI sono fondamentali un dialogo e una collaborazione continui. Governi, industria, mondo accademico e società civile devono impegnarsi attivamente per conseguire un approccio armonizzato ed efficace alla governance dell'AI a livello mondiale. Inoltre, data la natura dinamica dell'AI, i regolamenti e gli standard devono essere riesaminati e aggiornati periodicamente per rimanere pertinenti ed efficaci nel guidare lo sviluppo, il deployment e l'utilizzo responsabili dell'AI.

2 Caratteristiche di Qualità per i Sistemi Basati sull'AI – 45 minuti

Parole Chiave

Adattabilità funzionale, correttezza funzionale dell'AI, intervenibilità, robustezza dell'AI, sicurezza fisica, mitigazione dei rischi sociali ed etici, trasparenza, controllabilità dell'utente

Parole Chiave Specifiche dell'AI

Nessuna

Obiettivi di Apprendimento per il Capitolo 2:

2.1 Caratteristiche di Qualità per i Sistemi Basati sull'AI

AI-2.1.1 (K2) Classificare i comportamenti dei sistemi basati sull'AI in base alle caratteristiche di qualità definite nella norma ISO/IEC 25059

AI-2.1.2 (K2) Spiegare le particolari considerazioni che emergono quando l'AI viene utilizzata in sistemi correlati alla sicurezza fisica

2.2 Criteri di Accettazione per i Sistemi Basati sull'AI

AI-2.2.1 (K2) Fornire esempi di criteri di accettazione per i sistemi basati sull'AI

2.1 Caratteristiche di Qualità per i Sistemi Basati sull'AI

Questa sezione tratta le caratteristiche di qualità specifiche dell'AI descritte nella norma ISO/IEC 25059, che estende i modelli tradizionali della qualità del software per considerare gli aspetti peculiari dei sistemi basati sull'AI. Introduce caratteristiche nuove e adattate, tra cui la correttezza funzionale dell'AI, l'adattabilità funzionale, la controllabilità dell'utente, la trasparenza, la robustezza dell'AI e l'intervenibilità, insieme alla mitigazione dei rischi sociali ed etici. Vengono inoltre affrontate le principali sfide dell'AI in materia di sicurezza fisica, quali specifiche vaghe, non determinismo, autoapprendimento, spiegabilità limitata e standard in evoluzione, ponendo l'accento sul loro impatto sul testing e sulla regolamentazione.

2.1.1 Caratteristiche di Qualità Specifiche dell'AI

La norma ISO/IEC 25059 estende il modello di qualità della norma ISO/IEC 25010 per considerare gli aspetti specifici dell'AI. Questa estensione valuta i sistemi basati sull'AI da due prospettive: qualità del prodotto e qualità in uso. Dal punto di vista del testing, tali caratteristiche di qualità influenzano direttamente il modo in cui vengono definiti gli obiettivi del test, formulati i criteri di accettazione e interpretati i risultati dei test per i sistemi basati sull'AI. Rispetto alla norma ISO/IEC 25010, le caratteristiche nuove e modificate comprendono:

- **Correttezza funzionale dell'AI (qualità del prodotto):** i sistemi basati sull'AI, in particolare quelli che utilizzano il machine learning probabilistico, non possono garantire un'accuratezza perfetta. Poiché negli output dell'AI è previsto un certo tasso di errore, il concetto di correttezza funzionale è stato adattato di conseguenza. La norma ISO/IEC 25059 valuta la correttezza funzionale considerando sia gli output corretti sia quelli errati e definendo soglie accettabili per i risultati errati, in modo da riflettere la variabilità intrinseca degli output dei sistemi basati sull'AI (vedi 3.3).
- **Adattabilità funzionale (qualità del prodotto):** una nuova sottocaratteristica dell'adeguatezza funzionale. È la capacità del sistema basato sull'AI di adattarsi autonomamente ai cambiamenti del proprio ambiente operativo dopo il deployment.
- **Controllabilità dell'utente (qualità del prodotto):** una nuova sottocaratteristica dell'abilità di interazione (si noti che "abilità di interazione" è a sua volta un nuovo termine che sostituisce "usabilità" nella versione 2023 della norma ISO/IEC 25010). È una proprietà di un sistema basato sull'AI grazie alla quale una persona o un altro agente esterno può intervenire tempestivamente sul suo funzionamento.
- **Trasparenza (qualità del prodotto e qualità in uso):** una nuova sottocaratteristica dell'abilità di interazione e una nuova sottocaratteristica della soddisfazione. Riguarda il grado con cui le informazioni appropriate sul sistema basato sull'AI vengono comunicate agli stakeholder (vedi 6.1.2).
- **Robustezza dell'AI (qualità del prodotto):** una nuova sottocaratteristica dell'affidabilità. Descrive la capacità di un sistema basato sull'AI di mantenere il proprio livello di correttezza funzionale dell'AI indipendentemente dalle circostanze, ad esempio in presenza di dati di input distorti, avversari o non validi, di interferenze esterne, di condizioni ambientali avverse e di utilizzo improprio da parte dell'operatore.

- **Intervenibilità (qualità del prodotto):** una nuova sottocaratteristica della sicurezza. È il grado con cui un operatore può intervenire tempestivamente sul funzionamento di un sistema basato sull'AI per prevenire danni o pericoli.
- **Mitigazione dei rischi sociali ed etici (qualità in uso):** una nuova sottocaratteristica dell'“assenza di rischio”. Considera numerosi ambiti per mitigare i rischi sociali ed etici, tra cui responsabilità, equità e non discriminazione, responsabilità professionale, promozione dei valori umani, privacy, sicurezza fisica e sicurezza, controllo umano della tecnologia, coinvolgimento e sviluppo della comunità, progettazione basata sull'uomo, rispetto dello Stato di diritto, rispetto delle norme internazionali di comportamento, sostenibilità ambientale e pratiche lavorative.

2.1.2 AI e Sicurezza Fisica

I sistemi correlati alla sicurezza fisica possono causare lesioni o danni alle persone, ai beni o all'ambiente. Lo sviluppo e il testing di sistemi non basati sull'AI correlati alla sicurezza fisica possono richiedere un notevole impegno, ma sono fattibili; per i sistemi basati sull'AI esistono tuttavia diverse sfide aggiuntive:

- **Specifiche:** nei sistemi tradizionali correlati alla sicurezza fisica, i requisiti vengono definiti per l'intero sistema e raffinati finché lo sviluppatore non è in grado di trasformarli in codice. I requisiti di molti sistemi basati sull'AI spesso partono da obiettivi vaghi e vengono poi forniti implicitamente attraverso i dati di addestramento, che codificano pattern, regole e obiettivi senza formalizzare in anticipo ogni dettaglio. Ciò può comportare una tracciabilità inadeguata dai requisiti all'implementazione nei sistemi basati sull'AI.
- **Non determinismo:** questa caratteristica di molti sistemi basati sull'AI rende intrinsecamente difficile garantirne il comportamento preciso. Anche modelli sottoposti a testing rigoroso possono mostrare comportamenti inattesi a causa di fattori quali la generazione di numeri casuali o lievi variazioni dei valori di input.
- **Autoapprendimento:** il testing rigoroso viene utilizzato per dimostrare l'integrità della sicurezza fisica di un sistema prima del deployment. Nei sistemi basati sull'AI autoapprendenti tale dimostrazione viene compromessa, perché il comportamento del sistema si allontana progressivamente da quello originariamente sottoposto a testing. La gestione del modo in cui il modello apprende e dei dati che utilizza può talvolta contribuire a evitare l'emergere di nuovi comportamenti problematici. In alternativa, è possibile implementare protezioni di sicurezza fisica che impediscano al modello di apprendere o di prendere decisioni tali da compromettere la sicurezza fisica (ad esempio un componente di moderazione dei contenuti che filtra i prompt).
- **Spiegabilità e trasparenza:** per i sistemi correlati alla sicurezza fisica è essenziale comprendere come e perché il sistema prende le proprie decisioni. Tuttavia, i processi decisionali dei sistemi basati sull'AI spesso non sono trasparenti. Tecniche di AI spiegabile, quali LIME (Local Interpretable Model-agnostic Explanations), possono fornire indicazioni sul ragionamento del sistema basato sull'AI; tuttavia non sono ampiamente disponibili e possono comprometterne le prestazioni.
- **Regolamentazione in evoluzione:** il panorama normativo relativo ai sistemi basati sull'AI correlati alla sicurezza fisica è in continua evoluzione. L'utilizzo dell'AI non è attualmente contemplato dagli standard internazionali consolidati di safety funzionale e alcuni di tali standard ne vietano persino l'impiego in questi sistemi. L'EU AI Act [EU AI Act] (vedi 1.1.8) classifica come ad alto

rischio i sistemi AI utilizzati come componenti di sicurezza fisica, ad esempio nell'aviazione, nei dispositivi medici o nel settore automobilistico, e impone requisiti rigorosi per il loro sviluppo e testing.

2.2 Criteri di Accettazione per i Sistemi Basati sull'AI

Questa sezione descrive i criteri di accettazione relativi alle caratteristiche di qualità specificate nella norma ISO/IEC 25059 e alla sicurezza fisica. Per i sistemi basati sull'AI i criteri di accettazione devono spesso essere statistici, probabilistici o basati su soglie anziché binari, il che introduce ulteriori sfide di testing.

2.2.1 Criteri di Accettazione per i Sistemi Basati sull'AI

Nella valutazione della qualità di un sistema basato sull'AI è essenziale considerare sia le caratteristiche di qualità funzionali sia quelle non funzionali. Ciò aiuta a confermare che il sistema basato sull'AI funzioni come previsto e soddisfi requisiti di qualità più ampi. Le norme ISO/IEC 25010 e ISO/IEC 25059 forniscono un quadro completo per definire la qualità del software. In questa sezione l'attenzione è rivolta ai criteri di accettazione associati alle caratteristiche di qualità specifiche dell'AI (ossia quelle definite nella norma ISO/IEC 25059) e alla sicurezza fisica (vedi 2.1.2).

La tabella seguente elenca esempi di criteri di accettazione per la sicurezza fisica e per ciascuna delle caratteristiche di qualità definite nella norma ISO/IEC 25059.

Caratteristica	Esempi di criteri di accettazione
Correttezza funzionale dell'AI (vedi 3.3)	- Accuracy del 95% per un sistema di riconoscimento delle immagini. - Recall del 90% per un sistema di previsione dei difetti.
Adattabilità funzionale	- Un massimo di 20 secondi affinché il sistema di gestione del motore si adatti quando supera una soglia di altitudine specificata. - Un servizio di streaming video deve adattare la propria homepage in modo da raccomandare almeno il 40% di documentari dopo che un utente ha guardato integralmente tre documentari in una singola sessione.
Controllabilità dell'utente	- Un supervisore può assumere il controllo di un drone autonomo entro 0,5 secondi dal momento in cui questo invia un segnale di emergenza dovuto alla perdita della propria posizione GPS. - Il sistema di controllo dell'azienda agricola avvisa l'agricoltore quando le prestazioni visive del sensore si degradano di oltre il 30%, consentendo un override manuale immediato; si disattiva completamente se la degradazione supera il 50% senza una risposta dell'utente.

Trasparenza	- Vengono fornite informazioni sufficienti sul modello ML di terze parti e sulla provenienza dei relativi dati di addestramento per soddisfare i requisiti dello standard aziendale pertinente. - La dashboard operativa e l'API del sistema devono mettere a disposizione un endpoint che restituisca l'identificativo univoco di versione del modello di previsione attualmente utilizzato e un collegamento alla documentazione corrispondente.
Robustezza dell'AI	- Il tempo di risposta delle previsioni del sistema di allerta basato sull'AI per il penetration testing di sicurezza rimane inferiore a 1 secondo quando l'accesso al database centrale delle vulnerabilità viene interrotto per 30 secondi. - Il dispositivo AI edge deve passare automaticamente a una modalità di inferenza a fedeltà ridotta e a minore consumo energetico (anziché arrestarsi in modo anomalo) quando la propria temperatura operativa interna supera 85 °C per un periodo continuativo di 10 secondi.
Intervenibilità	- Se un robot viola la propria zona di sicurezza fisica, la linea di produzione può essere arrestata entro 0,5 secondi dall'avvio dell'arresto. - Per prevenire potenziali blackout, il sistema di gestione della rete elettrica deve fornire una finestra di conferma di 30 secondi, durante la quale un tecnico può porre il veto a qualsiasi azione proposta dall'AI e classificata come "critica", prima che venga eseguita automaticamente.
Mitigazione dei rischi sociali ed etici	- Il sistema automatizzato di determinazione delle pene detentive non discrimina tra gruppi razziali in base alla metrica di equità specificata. - Il chatbot deve superare una valutazione interna di red teaming con un punteggio pari o superiore al 95%, dimostrando di rifiutare la generazione di contenuti che promuovano violenza, autolesionismo o incitamento all'odio.
Sicurezza fisica	- I componenti non basati sull'AI del sistema di controllo dello sterzo basato sull'AI sono conformi alla norma ISO 26262-6 al livello ASIL (automotive safety integrity level) C. - Il 100% delle relazioni tra input e output del modello ML nel sistema di controllo della centrale nucleare può essere mappato, mediante uno strumento di spiegabilità, con un'accuratezza media non inferiore al 99,9%. - I segnali di controllo che superano di oltre il 10% i limiti di sicurezza fisica specificati vengono analizzati e regolati entro 0,15 secondi dal rilevamento da parte del sottosistema di monitoraggio della sicurezza fisica.

3 Machine Learning – 375 minuti

Parole Chiave

copertura k-multisezione dei neuroni, criteri di efficienza delle prestazioni funzionali di ML, metriche di efficienza delle prestazioni funzionali di ML, modello ML, copertura dei confini dei neuroni, copertura dei neuroni, perceptron

Parole Chiave Specifiche dell'AI

associazione, classificazione, clustering, preparazione dei dati, machine learning, algoritmo ML, framework di sviluppo ML, workflow ML, modello preaddestrato, regressione ML, apprendimento per rinforzo, apprendimento supervisionato, apprendimento non supervisionato

Obiettivi di Apprendimento per il Capitolo 3:

3.1 Introduzione al Machine Learning

- AI-3.1.1 (K2) Distinguere tra le diverse forme di ML
- AI-3.1.2 (K2) Riassumere il workflow utilizzato per creare un sistema ML
- HO-3.1.3 (H2) Creare un modello ML
- AI-3.1.4 (K2) Riassumere l'utilizzo dei modelli preaddestrati, del fine-tuning e della Retrieval-Augmented Generation

3.2 Dati per il Machine Learning

- AI-3.2.1 (K2) Spiegare le attività relative alla preparazione dei dati
- HO-3.2.2 (H2) Effettuare la preparazione dei dati a supporto della creazione di un modello ML
- AI-3.2.3 (K2) Mettere a confronto l'utilizzo dei dataset di addestramento, di validazione e di test nello sviluppo di un modello ML

3.3 Metriche di Efficienza delle Prestazioni Funzionali di ML per la Classificazione

- AI-3.3.1 (K3) Calcolare le comuni metriche di efficienza delle prestazioni funzionali di ML a partire da un insieme di dati di una matrice di confusione
- HO-3.3.2 (H2) Valutare un modello ML utilizzando metriche selezionate di efficienza delle prestazioni funzionali di ML
- HO-3.3.3 (H2) Mostrare l'impatto di differenti combinazioni di modelli ML e dataset sull'addestramento e sul comportamento dei modelli

3.4 Reti Neurali

- AI-3.4.1 (K2) Spiegare la struttura e il funzionamento di una rete neurale profonda
- HO-3.4.2 (H1) Sperimentare l'implementazione di un perceptron
- AI-3.4.3 (K2) Descrivere le diverse misure di copertura per le reti neurali

3.1 Introduzione al Machine Learning

Questa sezione presenta le principali categorie di algoritmi ML: apprendimento supervisionato, apprendimento non supervisionato e apprendimento per rinforzo, distinguendone i rispettivi tipi di problemi e le applicazioni tipiche. Descrive il workflow standard per lo sviluppo dei modelli ML, dalla definizione degli obiettivi e dalla preparazione dei dati fino all'addestramento, alla valutazione e al deployment dei modelli, prestando attenzione all'iterazione e all'integrazione nel sistema. Sono trattati anche aspetti pratici, quali la creazione diretta di un modello, l'utilizzo di modelli preaddestrati, il fine-tuning e la Retrieval-Augmented Generation, evidenziando gli approcci che consentono di adattare e migliorare in modo efficiente i modelli AI per nuove attività, gestendo al contempo le limitazioni ereditate.

La comprensione di questo workflow è essenziale per i tester, poiché attività di test differenti si applicano in fasi differenti e i failure hanno spesso origine in passi precedenti, quali la preparazione dei dati o la selezione del modello.

3.1.1 Diverse Forme di Machine Learning

Gli algoritmi ML si classificano in apprendimento supervisionato, apprendimento non supervisionato e apprendimento per rinforzo.

Nell'apprendimento supervisionato, gli algoritmi addestrano i modelli utilizzando dati etichettati, nei quali a ogni insieme di input corrisponde un'etichetta di output (ad esempio immagini etichettate come "cane" o "gatto"). Il modello apprende a mappare gli input sugli output individuando pattern nei dati di addestramento. L'apprendimento supervisionato è generalmente suddiviso in:

- **Classificazione:** consiste nell'assegnare gli input a classi predefinite, ad esempio classificando le e-mail come spam o non spam, oppure nel riconoscimento di oggetti nelle immagini.
- **Regressione ML:** consiste nel prevedere valori numerici continui, ad esempio stimando l'età di una persona sulla base dei dati relativi allo stile di vita oppure prevedendo l'andamento dei prezzi delle azioni.

Si noti che il termine regressione ML, quando utilizzato nel contesto del machine learning, differisce dal suo impiego negli altri syllabus ISTQB®, nei quali la regressione descrive il problema delle modifiche del software che causano difetti correlati alle modifiche.

Nell'apprendimento non supervisionato, l'algoritmo addestra i modelli utilizzando dati non etichettati e deduce pattern o strutture senza etichette di output esplicite. Il modello raggruppa input simili in base alle caratteristiche condivise. L'apprendimento non supervisionato è generalmente suddiviso in:

- **Clustering:** consiste nel raggruppare i punti dati in base alle somiglianze, ad esempio segmentando i clienti in gruppi differenti per attività di marketing mirate.
- **Associazione:** consiste nell'identificare relazioni o dipendenze tra gli attributi dei dati, ad esempio individuando pattern nel comportamento di acquisto dei clienti per raccomandare prodotti.

Nell'apprendimento per rinforzo, il sistema basato sull'AI (un "agente intelligente") apprende interagendo con il proprio ambiente. L'agente riceve feedback positivi (ricompense) o negativi (penalizzazioni) in base all'esito delle proprie azioni, e può così apprendere dall'esperienza anziché da un dataset. Le sfide dell'apprendimento per rinforzo comprendono la configurazione dell'ambiente, la progettazione della

funzione di ricompensa e la selezione della strategia migliore per raggiungere l'obiettivo desiderato. Le applicazioni comprendono la robotica, i veicoli autonomi e i sistemi adattivi quali i chatbot.

Ogni approccio ML affronta tipi di problemi differenti; la scelta dipende dalla natura dei dati disponibili e dalla specifica attività da svolgere.

3.1.2 Workflow del Machine Learning

Le attività del workflow ML, illustrate nella Figura 1, sono:

Comprendere gli Obiettivi (Understand the Objectives)

Lo scopo del modello ML viene compreso e concordato dagli stakeholder per verificarne l'allineamento con le priorità di business. Per il modello sviluppato vengono definiti criteri di accettazione (incluse le metriche di efficienza delle prestazioni funzionali di ML – vedi 3.3).

Selezionare un Framework (Select a Framework)

Viene selezionato un framework di sviluppo ML appropriato (vedi 1.1.7 per i dettagli sulle funzionalità fornite) sulla base degli obiettivi, dei criteri di accettazione (vedi 2.2) e delle priorità di business.

Selezionare e Costruire l'Algoritmo (Select & Build the Algorithm)

Viene selezionato un algoritmo ML in base a diversi fattori, tra cui gli obiettivi, i criteri di accettazione e i dati disponibili (vedi 3.2). L'algoritmo può essere codificato manualmente, ma spesso viene recuperato da una libreria software. Se necessario, l'algoritmo viene quindi compilato.

Preparare e Testare i Dati (Prepare & Test Data)

La preparazione dei dati (vedi 3.2) comprende l'acquisizione dei dati, la pre-elaborazione dei dati e il feature engineering. Parallelamente a queste attività può essere eseguita l'analisi esplorativa dei dati (EDA).

I dati utilizzati dall'algoritmo e dal modello si basano sugli obiettivi e vengono impiegati da tutte le attività del riquadro "generazione e test del modello" (Model Generation & Test) illustrato nella Figura 1.

I dati utilizzati per addestrare, valutare, mettere a punto e testare il modello devono essere rappresentativi dei dati che il modello utilizzerà durante l'operatività.

Viene eseguito il testing dei dati e di tutti i passi automatizzati di preparazione dei dati (vedi Capitolo 5).

Addestrare il Modello (Train the Model)

L'algoritmo ML selezionato utilizza i dati di addestramento per addestrare il modello.

I parametri che definiscono la struttura del modello (ad esempio il numero di strati di una rete neurale o la profondità di un albero decisionale) vengono forniti all'algoritmo. Questi parametri sono denominati iperparametri del modello.

All'algoritmo vengono forniti anche i parametri che controllano l'addestramento (ad esempio il numero di iterazioni da utilizzare durante l'addestramento di una rete neurale). Questi parametri sono denominati iperparametri dell'algoritmo.

Valutare il Modello (Evaluate the Model)

Il modello viene valutato rispetto alle metriche concordate di efficienza delle prestazioni funzionali di ML (vedi 3.3), utilizzando il dataset di validazione; i risultati vengono utilizzati per migliorare il modello nell'attività "mettere a punto il modello". Nella pratica vengono generalmente creati e addestrati diversi modelli utilizzando algoritmi differenti (ad esempio random forest, SVM e reti neurali) e vari dataset di addestramento, e la combinazione migliore viene scelta in base ai risultati della valutazione.

Mettere a Punto il Modello (Tune the Model)

I risultati della valutazione vengono utilizzati per regolare gli iperparametri del modello e gli iperparametri dell'algoritmo. Il modello viene quindi riaddestrato con queste impostazioni modificate, per migliorarne l'efficienza delle prestazioni funzionali di ML.

Le tre attività di addestramento, valutazione e messa a punto costituiscono la "generazione del modello" (Model Generation), come illustrato nella Figura 1.

Testare il Modello (Test the Model)

Una volta generato un modello accettabile mediante le attività di "generazione del modello", questo viene testato utilizzando un dataset di test indipendente per verificare che siano soddisfatti i criteri concordati di efficienza delle prestazioni funzionali di ML. I risultati dei test vengono inoltre confrontati con quelli della valutazione. Se le prestazioni del modello sui dati di test indipendenti sono significativamente inferiori a quelle rilevate durante la valutazione, può essere necessario tornare alle attività di "generazione del modello" o persino all'attività "preparare e testare i dati" per addestrare un nuovo modello.

Oltre ai test dell'efficienza delle prestazioni funzionali di ML, possono essere eseguiti anche test non funzionali, ad esempio sul tempo necessario per fornire una previsione. In genere il testing di questa attività viene eseguito da data engineer o data scientist; tuttavia anche i tester con una conoscenza sufficiente del dominio e con accesso alle risorse pertinenti possono eseguirlo.

Eseguire il Deployment del Modello (Deploy the Model)

Una volta completata la "generazione e test del modello", il modello messo a punto viene generalmente reingegnerizzato per il deployment insieme alla relativa pipeline di dati. Ciò avviene normalmente mediante il framework di sviluppo ML. Le piattaforme target possono comprendere sistemi embedded e il cloud, nel quale è possibile accedere al modello tramite una web API. Il modello reingegnerizzato e sottoposto a deployment viene testato per verificare che continui a soddisfare i propri criteri di accettazione.

Utilizzare il Modello (Use the Model)

Dopo il deployment, il modello viene generalmente integrato in un sistema operativo basato sull'AI più ampio. I modelli possono eseguire previsioni batch pianificate a intervalli prestabiliti oppure funzionare in tempo reale su richiesta.

Monitorare e Mettere a Punto il Modello (Monitor & Tune the Model)

Durante l'utilizzo del modello il contesto può evolvere e il modello può allontanarsi per deriva dalle prestazioni previste (vedi 6.1.7). Per verificare che ogni deriva venga identificata e gestita, il modello operativo viene valutato periodicamente rispetto ai propri criteri di accettazione.

Può essere ritenuto necessario creare un nuovo modello mediante riaddestramento con nuovi dati, con nuovi iperparametri o con entrambi. Il modello più recente può quindi essere confrontato con quello esistente mediante una forma di testing A/B (vedi 6.1.9).

Il workflow ML illustrato nella Figura 1 è una sequenza logica; nella pratica, tuttavia, il workflow viene applicato iterativamente e i passi vengono ripetuti.

I passi illustrati nella Figura 1 non comprendono l'integrazione del modello ML con le parti non ML del sistema complessivo. In genere i modelli ML non possono essere sottoposti a deployment isolatamente e devono essere integrati con componenti non ML (ad esempio, nelle applicazioni di visione viene utilizzata una pipeline di dati per pulire e modificare i dati prima di inviarli al modello ML). Quando il modello fa parte di un sistema di grandi dimensioni, deve essere integrato in tale sistema prima del deployment. In questo caso possono essere eseguiti i livelli di test di integrazione, di sistema e di accettazione.

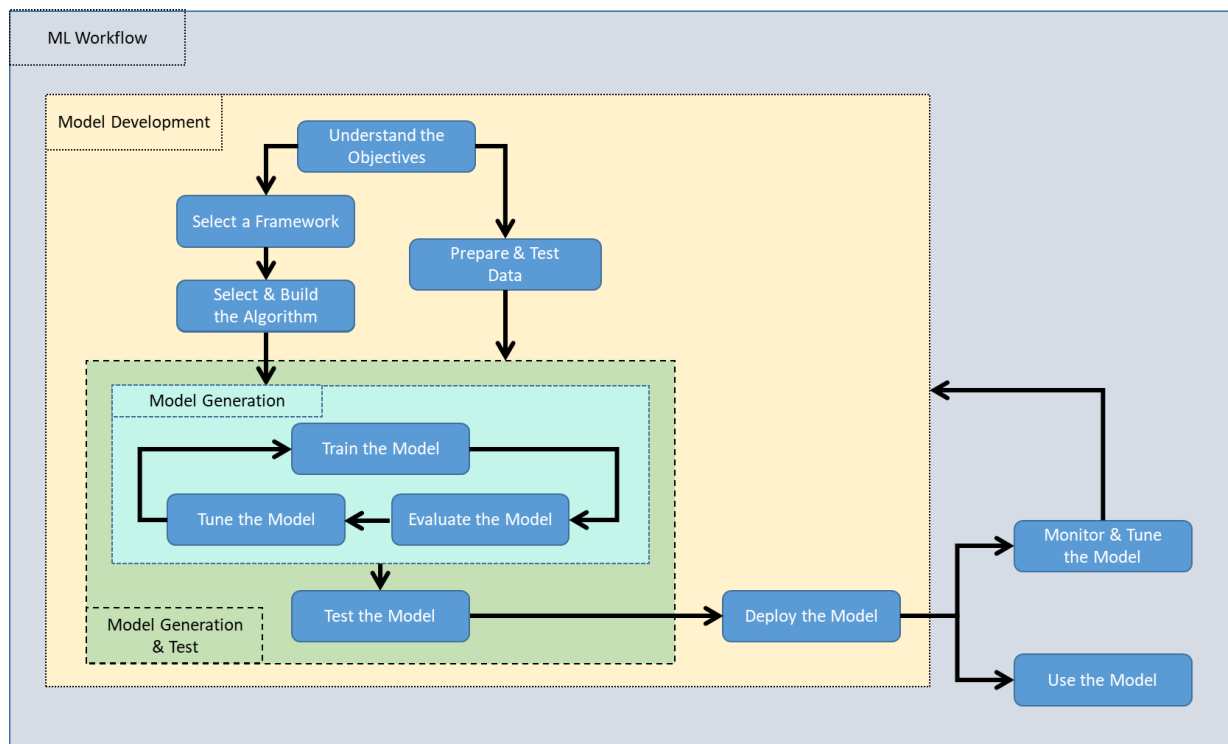


Figura 1: Workflow ML

3.1.3 Esercizio Pratico: Creare un Modello di Machine Learning

Selezionare, addestrare e testare un modello di classificazione mediante l'apprendimento supervisionato.

Spiegare la differenza tra valutazione/messa a punto e testing confrontando l'accuratezza ottenuta con i dataset di validazione e di test.

3.1.4 Modelli Preaddestrati, Fine-Tuning e Retrieval-Augmented Generation

L'addestramento di un nuovo modello AI da zero è costoso e richiede molto tempo. Per affrontare questo problema, una soluzione comune è il fine-tuning, che consiste nel prendere una rete neurale preaddestrata e adattarla per svolgere una nuova attività differente. Uno dei principali vantaggi è che

richiede molti meno dati di addestramento (e un minore effort di addestramento) rispetto alla costruzione di un modello da zero.

Il modello preaddestrato viene sottoposto a fine-tuning eseguendo un addestramento aggiuntivo con dati specifici della nuova attività. La messa a punto può essere applicata all'intera rete neurale, soltanto a strati specifici (generalmente vicini all'estremità di output della rete neurale) oppure a strati aggiuntivi. Dopo l'addestramento viene valutata l'efficienza delle prestazioni funzionali di ML del modello e, sulla base dei risultati, può essere eseguito un ulteriore fine-tuning finché il modello non soddisfa i criteri di accettazione necessari.

Il successo del fine-tuning dipende dalla somiglianza tra l'attività originale e quella nuova. Differenze ridotte possono consentire un fine-tuning molto efficace. Ad esempio, l'adattamento di un classificatore di immagini di razze feline al riconoscimento delle razze canine ha buone probabilità di funzionare bene. L'adattamento al riconoscimento degli accenti parlati è invece meno efficace, a causa della differenza maggiore. Analogamente, il fine-tuning di un LLM per l'analisi dei valori limite definita da ISTQB richiede una modifica limitata del modello ed è facilmente realizzabile con buoni dati di addestramento.

Un'alternativa al fine-tuning è la Retrieval-Augmented Generation (RAG), che consiste nel fornire all'LLM fonti di dati specifiche per l'attività richiesta. Tali fonti vengono trasformate in un formato ricercabile che consente di confrontarle con l'argomento del prompt. Una volta identificati i documenti pertinenti, questi vengono incorporati in un prompt arricchito, che viene inviato all'LLM. Poiché all'LLM vengono ora fornite informazioni più pertinenti, è probabile che la risposta corrispondente sia più precisa. Con la RAG non viene apportata alcuna modifica al modello preaddestrato.

Un modello preaddestrato può utilizzare la RAG, il fine-tuning oppure entrambi per migliorare le prestazioni.

In genere i bias o le vulnerabilità presenti nel modello preaddestrato vengono ereditati dal nuovo modello; è quindi necessario eseguire il testing per confermare che il modello operi in modo affidabile ed equo nella nuova attività.

3.2 Dati per il Machine Learning

La preparazione dei dati è riconosciuta come una delle attività più importanti e dispendiose in termini di risorse del workflow ML. Se i dati operativi differiscono significativamente dai dati di addestramento, le ipotesi relative all'efficienza delle prestazioni funzionali di ML e alla sicurezza fisica potrebbero non essere più valide. La preparazione dei dati assorbe normalmente una quota dell'effort complessivo significativamente maggiore rispetto ad altre fasi, quali la selezione e la costruzione del modello. La preparazione dei dati è intrinsecamente collegata alla pipeline di dati, che elabora i dati grezzi e li trasforma in un formato utilizzabile dai modelli ML sia per l'addestramento sia per la previsione.

3.2.1 Attività di Preparazione dei Dati

La preparazione dei dati contribuisce a conseguire la qualità e l'idoneità dei dati per l'addestramento del modello. Comprende diverse attività fondamentali:

- Acquisizione dei dati:
 - Identificazione dei tipi di dati rilevanti (ad esempio numerici, categorici, immagini, testo).
 - Raccolta dei dati da fonti eterogenee, quali database, API o sensori in tempo reale.

- Etichettatura dei dati per le attività di apprendimento supervisionato, verificandone l'accuratezza e la coerenza.

I dati acquisiti possono assumere forme diverse (ad esempio numerici, categorici, immagini, tabellari, testo, serie temporali, dati da sensori, dati geospaziali, video e audio).

- Pre-elaborazione dei dati:
 - Pulizia dei dati, che comprende:
 - la rimozione di difetti, duplicati e valori anomali, per verificare l'accuratezza e la coerenza dei dati;
 - l'imputazione dei valori mancanti mediante tecniche quali media, mediana o moda, per mantenere la completezza dei dati;
 - l'anonimizzazione o la rimozione delle informazioni personali, per tutelare la privacy e rispettare le normative.
 - Trasformazione dei formati dei dati, scalatura e normalizzazione, per conseguire la coerenza.
 - Aumento dei dati per incrementare la dimensione del campione, inclusione di adversarial example per migliorare la robustezza agli attacchi avversari e generazione di dati sintetici.
 - Campionamento di sottoinsiemi per ridurre i tempi di addestramento e i costi computazionali.
- Feature engineering:
 - Selezione delle feature rilevanti in base al loro contributo alle prestazioni del modello ML.
 - Estrazione, dalle feature esistenti, di un sottoinsieme di feature informative e non ridondanti, per ridurre i tempi di addestramento e i costi computazionali.

Parallelamente a queste attività di preparazione dei dati viene normalmente eseguita anche l'analisi esplorativa dei dati (EDA), per ricavare informazioni utili sui dati. Tale analisi comprende:

- l'individuazione di tendenze, pattern e anomalie nei dati;
- la visualizzazione dei dati mediante grafici e diagrammi, per facilitarne la comprensione.

La preparazione dei dati di addestramento è normalmente un processo iterativo, spesso eseguito manualmente, e le singole attività di preparazione dei dati possono essere riordinate oppure omesse in base ai requisiti specifici del progetto. I dati operativi dovrebbero corrispondere alle caratteristiche dei dati di addestramento (ad esempio distribuzioni dei dati e intervalli delle feature), affinché il modello si comporti in produzione come previsto. I passi di preparazione possono tuttavia essere adattati per migliorare l'efficienza e la scalabilità in produzione.

3.2.2 Esercizio Pratico: Preparazione dei Dati a Supporto della Creazione di un Modello di Machine Learning

Per un determinato insieme di dati, effettuare i passi di preparazione applicabili descritti nella Sezione 3.2.1, al fine di produrre un dataset che verrà utilizzato per creare un modello di classificazione mediante l'apprendimento supervisionato.

Questa attività costituisce il primo passo della creazione di un modello ML che verrà utilizzato negli esercizi successivi.

Per svolgere questa attività, agli studenti saranno forniti materiali appropriati (e specifici per la lingua), tra cui:

- librerie;
- framework di sviluppo ML;
- strumenti.

3.2.3 Dataset di Addestramento, di Validazione e di Test

Dal punto di vista logico, per sviluppare un modello ML sono necessari tre insiemi di dati equivalenti (ad esempio selezionati casualmente da un unico dataset rappresentativo):

- Un dataset di addestramento viene utilizzato per addestrare il modello.
- Un dataset di validazione viene utilizzato per valutare e successivamente mettere a punto il modello.
- Un dataset di test, noto anche come dataset holdout, viene utilizzato per testare il modello messo a punto.

Quando è disponibile una grande quantità di dati idonei, la quantità di dati utilizzata nel workflow ML per l'addestramento, la valutazione e il testing dipende normalmente dai fattori seguenti:

- La complessità prevista del modello.
- L'algoritmo utilizzato per addestrare il modello.
- La disponibilità di risorse, quali RAM, spazio su disco, capacità di calcolo, larghezza di banda della rete e tempo disponibile.
- Il livello di confidenza desiderato nel modello risultante.

Quando i dati sono limitati, una suddivisione in tre parti (addestramento, validazione e test) può lasciare una quantità di dati insufficiente per un addestramento efficace del modello, aumentando così il rischio di underfitting. Per affrontare questo problema, una strategia comune consiste nel riservare, quando possibile, un piccolo dataset finale di test holdout. I dati rimanenti (l'insieme combinato di addestramento e validazione) vengono quindi utilizzati per tecniche quali la cross-validation k-fold (dove k è un numero intero specificato dall'utente, normalmente 5 o 10).

Nella cross-validation k-fold questi dati vengono suddivisi in k "fold". Per ciascun fold il modello viene addestrato su k-1 fold e validato sul fold escluso. Il processo viene ripetuto k volte e ciascun fold funge una volta da insieme di validazione. Ciò consente una messa a punto robusta degli iperparametri e una stima affidabile dell'efficienza delle prestazioni funzionali di ML. I dati vengono normalmente assegnati casualmente ai fold, spesso mediante campionamento stratificato per rendere rappresentativo ciascun fold, in particolare in presenza di dati sbilanciati o di dataset di piccole dimensioni.

Le metriche delle prestazioni (ad esempio accuracy, F1-score – vedi 3.3.1) ottenute dalla validazione di ciascun fold vengono quindi mediate, per fornire una stima più affidabile della capacità di generalizzazione del modello. Dopo aver identificato gli iperparametri ottimali mediante cross-validation, viene normalmente addestrato un modello finale sull'intero insieme di addestramento e validazione (tutti i dati tranne il dataset di test holdout) utilizzando tali iperparametri. Questo modello finale viene quindi

valutato una sola volta sul dataset di test holdout, per una valutazione finale e non distorta delle prestazioni. Qualora non sia possibile predisporre un dataset di test holdout a causa di un'estrema scarsità di dati, la prestazione media ottenuta mediante cross-validation presenta una distorsione ottimistica e non può costituire una stima non distorta.

Tra gli altri metodi di ricampionamento applicabili quando i dati sono limitati rientrano la cross-validation leave-one-out (un caso particolare della cross-validation k-fold in cui k è uguale al numero dei campioni) e le tecniche di bootstrap.

3.3 Metriche di Efficienza delle Prestazioni Funzionali di ML per la Classificazione

Nelle attività di classificazione (vedi 3.1.1) è possibile utilizzare una matrice di confusione per valutare le previsioni di un modello, classificandole come veri positivi, veri negativi, falsi positivi o falsi negativi. Da questa matrice si ricavano metriche fondamentali quali accuracy, precision, recall e F1-score. Tali metriche misurano la qualità della classificazione, evidenziando i punti di forza e di debolezza del modello ML. Questa sezione esamina il calcolo e l'interpretazione di tali metriche per valutare l'efficienza delle prestazioni funzionali di ML.

3.3.1 Calcolo delle Metriche di Efficienza delle Prestazioni Funzionali di Machine Learning

In un problema di classificazione un modello raramente prevede sempre correttamente i risultati, in parte a causa della natura probabilistica dei modelli ML e del rumore nei dati. Per qualsiasi problema di questo tipo è possibile creare una matrice di confusione con le possibilità seguenti:

		Effettivo	
		Positivo	Negativo
Previsto	Positivo	Vero Positivo (TP)	Falso Positivo (FP)
	Negativo	Falso Negativo (FN)	Vero Negativo (TN)

Figura 2: Matrice di Confusione

Si noti che la matrice di confusione illustrata nella Figura 2 può essere rappresentata diversamente (ad esempio invertendo “previsto” ed “effettivo”), ma produrrà sempre valori per le quattro situazioni possibili: vero positivo (TP), vero negativo (TN), falso positivo (FP) e falso negativo (FN).

Sulla base della matrice di confusione vengono definite le metriche seguenti:

- $\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) * 100\%$

L'accuracy misura la percentuale di tutte le classificazioni corrette.

- $\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) * 100\%$

La precision misura la proporzione dei positivi previsti correttamente. È una misura del grado di certezza attribuibile alle previsioni positive.

- $\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) * 100\%$

La recall (nota anche come sensitivity) misura la proporzione dei positivi effettivi previsti correttamente. È una misura del grado di confidenza con cui si può ritenere che nessun positivo venga omissso.

- $\text{F1-score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$

L'F1-score è calcolato come media armonica di precision e recall e assume valori compresi tra 0 e 100. Un valore prossimo a 100 indica che il modello consegue sia una precision elevata sia una recall elevata, il che significa che gli errori di classificazione (falsi positivi e falsi negativi) hanno un impatto minimo. Al contrario, un F1-score basso indica che il modello ha difficoltà a identificare correttamente i positivi, perché non riconosce casi effettivi oppure genera molti falsi allarmi.

3.3.2 Esercizio Pratico: Valutare un Modello di Machine Learning utilizzando Metriche Selezionate di Efficienza delle Prestazioni Funzionali di ML

Utilizzando il modello di classificazione addestrato nell'esercizio precedente, calcolare e visualizzare i valori di accuracy, precision, recall e F1-score. Ove applicabile, utilizzare per i calcoli le funzioni di libreria fornite dal proprio framework di sviluppo ML.

3.3.3 Esercizio Pratico: Mostrare l'Impatto di Diverse Combinazioni di Modelli di Machine Learning e Dataset

Proseguendo l'esercizio precedente, utilizzare modelli ML e combinazioni di dataset differenti per osservare l'effetto sull'addestramento del modello ML e sul comportamento finale del modello. Osservare i tempi di addestramento e le metriche di efficienza delle prestazioni funzionali di ML.

3.4 Reti Neurali

Le reti neurali artificiali sono state inizialmente progettate per imitare il funzionamento del cervello umano, che può essere considerato una rete di neuroni biologici interconnessi.

Il perceptron a singolo strato è uno dei primi esempi di implementazione di una rete neurale artificiale ed è costituito da un solo strato. Può essere utilizzato per l'apprendimento supervisionato di classificatori binari per problemi linearmente separabili, che determinano se un input appartenga o meno a una determinata classe. Ad esempio, un perceptron può distinguere tra messaggi di posta elettronica che

sono spam e messaggi che non lo sono, apprendendo a separare con una linea retta, nello spazio di input, le feature delle due categorie.

La maggior parte delle reti neurali attuali è considerata una rete neurale profonda, perché comprende più strati. Le reti completamente connesse possono essere considerate perceptron multistrato (vedi Figura 3).

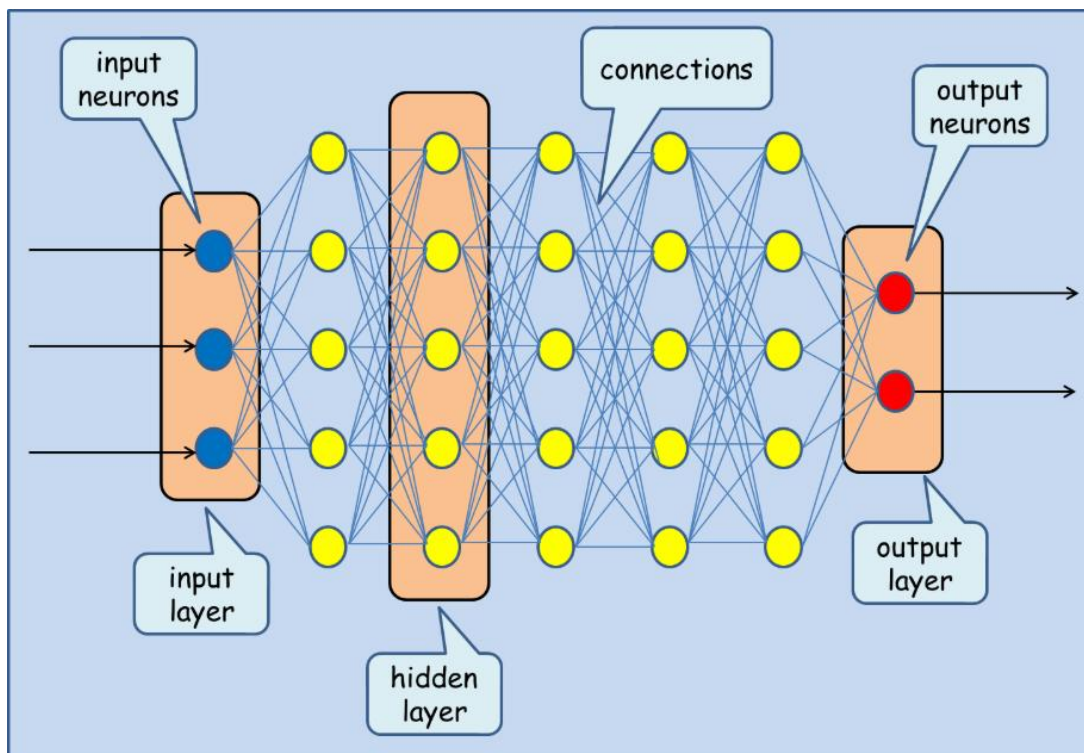


Figura 3: Struttura di una rete neurale profonda

3.4.1 Struttura e Funzionamento di una Rete Neurale Profonda

Una rete neurale profonda viene normalmente descritta come composta da tre tipi principali di strati. Lo strato di input (input layer) riceve gli input, ad esempio i valori dei pixel provenienti da una telecamera. Lo strato di output (output layer) fornisce i risultati al mondo esterno: potrebbe trattarsi, ad esempio, di un valore che indica la probabilità che l'immagine di input rappresenti un gatto. Tra lo strato di input e quello di output si trovano gli strati nascosti (hidden layer), composti da neuroni artificiali, detti anche nodi. In molte architetture comuni, quali le reti completamente connesse, i neuroni di uno strato sono collegati a ciascuno dei neuroni dello strato successivo e il numero di neuroni può variare tra uno strato e il successivo.

I neuroni eseguono calcoli e propagano le informazioni nella rete dai neuroni di input (input neurons) a quelli di output (output neurons), trasformando gradualmente i dati di input in rappresentazioni sempre più astratte fino al raggiungimento dell'output.

Il calcolo eseguito da ciascun neurone (esclusi quelli dello strato di input) genera quello che viene definito valore di attivazione (activation value). Tale valore si calcola determinando innanzitutto la somma

ponderata dei valori di attivazione di tutti i neuroni connessi nello strato precedente, dove ogni connessione (connections) ha un proprio peso (weight) indipendente, e aggiungendo il valore individuale di bias del neurone. La somma viene quindi elaborata mediante una formula non lineare, detta funzione di attivazione. Si noti che questo bias non è correlato al bias trattato in 5.1.2. Funzioni di attivazione differenti producono valori di attivazione differenti.

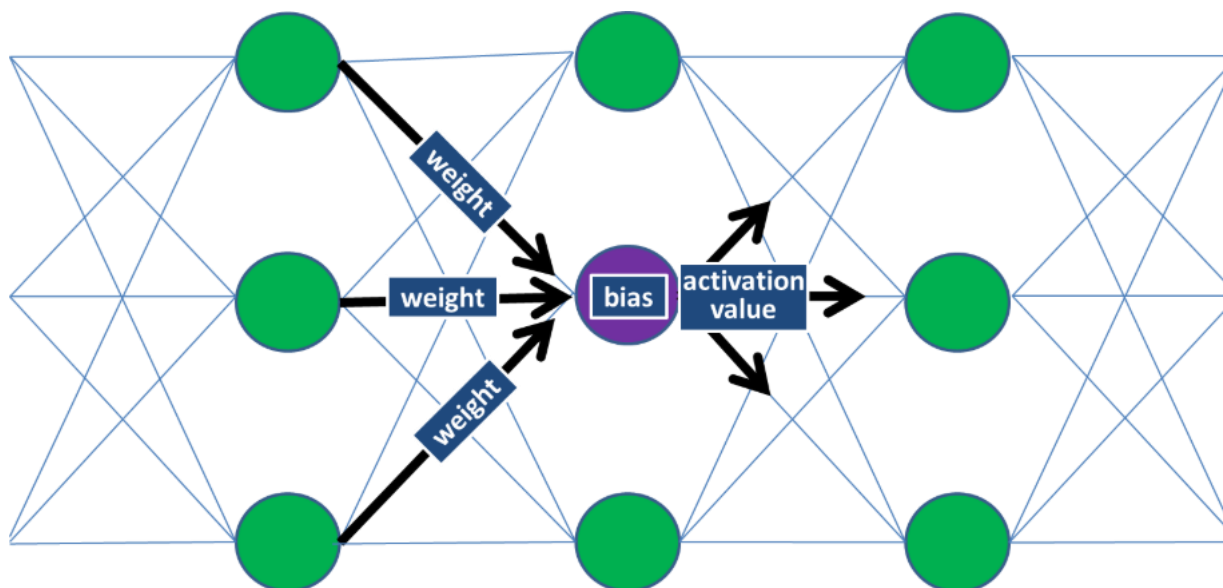


Figura 4: Calcolo eseguito da ciascun neurone

All'inizio dell'addestramento i pesi che collegano i neuroni e il valore di bias di ciascun neurone vengono normalmente inizializzati con piccoli valori casuali (talvolta i bias sono inizializzati a zero). I dati di addestramento vengono propagati attraverso la rete e ciascun neurone esegue la funzione di attivazione, fino a generare un output. L'output generato viene quindi confrontato con il risultato corretto noto. L'errore risultante (o loss), che quantifica tale differenza, viene retropropagato nella rete per modificare i valori dei pesi e dei bias e minimizzare così la differenza. Man mano che altri dati di addestramento vengono propagati nella rete (ogni passaggio attraverso il dataset di addestramento è detto epoca), i pesi e i valori di bias vengono gradualmente adattati mentre la rete apprende. Dopo un certo tempo, idealmente, l'output prodotto viene considerato sufficientemente valido da terminare l'addestramento.

3.4.2 Esercizio Pratico: Sperimentare l'Implementazione di un Perceptron

Gli studenti saranno guidati in un esercizio che dimostra come un perceptron apprenda una funzione semplice, quale la funzione AND.

L'esercizio dovrebbe illustrare come un perceptron apprenda modificando i propri pesi e valori di bias nel corso di più epoche, fino a ridurre a zero l'errore. Per questa attività possono essere utilizzati diversi strumenti (ad esempio un foglio di calcolo o una simulazione).

3.4.3 Misure di Copertura per le Reti Neurali

Le metriche di copertura strutturale delle reti neurali sono state introdotte per valutare in quale misura gli input di test esercitino i meccanismi interni di un modello. Poiché le reti neurali non seguono percorsi

codificati esplicitamente, ma il loro comportamento è determinato da pesi appresi, valori di bias e attivazioni, sono necessarie misure di copertura specializzate per valutare in quale misura le diverse parti della rete siano state attivate nelle condizioni di test.

Gli approcci tipici comprendono [COV_REF]:

- Copertura dei neuroni: misura la proporzione di neuroni della rete il cui output supera una soglia specificata durante il testing.
- Copertura k-multisezione dei neuroni (kMNC): il possibile intervallo di output di ciascun neurone viene suddiviso in k sezioni. La kMNC è la proporzione di tali sezioni attivate durante il testing.
- Copertura dei confini dei neuroni (NBC): misura la proporzione di neuroni della rete il cui output, durante il testing, supera il massimo raggiunto nell'addestramento oppure è inferiore al minimo raggiunto nell'addestramento.

Queste metriche di copertura possono essere utili nel testing dell'AI, principalmente perché rivelano aree del modello che rimangono non testate e che potrebbero nascondere difetti o comportamenti non appresi. Ad esempio, se determinati neuroni o strati non si attivano mai durante il testing, le prestazioni del modello in corrispondenza dei relativi confini decisionali potrebbero essere dubbie. Queste indicazioni aiutano i tester a progettare ulteriori input di test o condizioni di test, per esercitare le parti della rete esplorate in misura insufficiente. Attualmente gli strumenti commerciali che supportano queste specifiche misure di copertura sono limitati.

Tuttavia, la sola copertura strutturale non garantisce che una rete neurale generalizzi bene o gestisca le variazioni del mondo reale. Le reti neurali possono apprendere correlazioni spurie, producendo attivazioni corrette per motivi errati. Di conseguenza, i tester dovrebbero utilizzare anche altre tecniche di test, quali l'adversarial testing e il testing metamorfico, descritte nei Capitoli 5, 6 e 7.

4 Testing dei Sistemi Basati sull'AI – 195 minuti

Parole Chiave

attacco, testing esplorativo, testing basato sul rischio, oracolo del test

Parole Chiave Specifiche dell'AI

sistema basato sull'AI adattivo, sistema basato sull'AI, AI generativa, Large Language Model, sistema basato sull'AI bloccato

Obiettivi di Apprendimento per il Capitolo 4:

4.1 Introduzione al Testing dei Sistemi Basati sull'AI

- AI-4.1.1 (K2) Confrontare la testabilità dei sistemi basati sull'AI bloccati e adattivi
- AI-4.1.2 (K2) Spiegare perché nel testing dei sistemi basati sull'AI è spesso necessario un approccio statistico
- AI-4.1.3 (K2) Spiegare le sfide e le soluzioni relative agli oracoli del test per i sistemi basati sull'AI

4.2 Testing dell'AI Generativa e degli LLM

- AI-4.2.1 (K2) Spiegare come può essere testata l'AI generativa
- AI-4.2.2 (K3) Implementare il red teaming per i sistemi GenAI
- HO-4.2.3 (H2) Applicare il testing esplorativo a un LLM che esegue l'analisi dei valori limite

4.3 Livelli di Test e Sistemi di Machine Learning

- AI-4.3.1 (K2) Riassumere i livelli di test utilizzati per sviluppare i sistemi di machine learning
- AI-4.3.2 (K2) Spiegare come viene applicato il testing basato sul rischio ai sistemi di machine learning

4.1 Introduzione al Testing dei Sistemi Basati sull'AI

Il testing dei sistemi basati sull'AI presenta sfide specifiche rispetto al testing del software convenzionale. Innanzitutto, i sistemi basati sull'AI possono essere suddivisi in due ampie categorie: bloccati e adattivi. I sistemi basati sull'AI bloccati, il cui comportamento rimane fisso dopo il deployment, sono più facili da testare per la loro natura prevalentemente deterministica; i sistemi adattivi, che evolvono e apprendono, introducono invece complessità, poiché il loro comportamento può cambiare in modo imprevedibile. Inoltre, la natura probabilistica di molti modelli AI rende spesso necessario il testing statistico, poiché i metodi deterministici possono non essere sufficienti per valutare output influenzati da dati e probabilità. Ciò richiede la valutazione delle distribuzioni delle prestazioni e dei livelli di confidenza in scenari eterogenei.

Una sfida centrale del testing, nota come “problema dell'oracolo del test”, si presenta quando occorre determinare se l'output prodotto dal system under test sia corretto per un determinato input. Nel software tradizionale, specifiche ben definite rendono relativamente semplice specificare i risultati attesi e verificarne la correttezza. Nei sistemi basati sull'AI, invece, soprattutto quando affrontano attività complesse o soggettive, definire chiaramente i risultati attesi può essere difficile o persino impossibile. Questa incertezza aumenta per attività che superano le capacità umane, comportano vaghezza o non dispongono di una “ground truth” oggettiva; ciò rende difficile implementare oracoli del test automatizzati e può richiedere il giudizio di esperti o un ragionamento statistico.

Requisiti di sistema vaghi o incompleti aggravano ulteriormente il problema dell'oracolo del test nel testing dell'AI. Le soluzioni possono comprendere l'utilizzo di valutazioni statistiche, la consultazione di esperti di dominio oppure la definizione di una “ground truth” rispetto alla quale valutare gli output. Questi approcci mirano a stabilire aspettative affidabili e a valutare efficacemente i sistemi basati sull'AI.

4.1.1 Sistemi Basati sull'AI Bloccati e Adattivi

Molti sistemi basati sull'AI attualmente in uso sono sistemi basati sull'AI bloccati, che non modificano il proprio comportamento dopo il deployment. Un esempio di sistema basato sull'AI bloccato è un modello ML basato su una DNN e già sottoposto a deployment, nel quale i pesi e i bias della DNN sono fissati al termine dello sviluppo e possono essere modificati soltanto riaddestrando la DNN. Le tecnologie per veicoli a guida autonoma correlate alla sicurezza fisica si basano spesso su sistemi basati sull'AI bloccati per attività specifiche, quali il rilevamento della corsia o il riconoscimento dei segnali stradali.

Al contrario, un sistema basato sull'AI adattivo, quale un sistema di apprendimento per rinforzo, può adattare il proprio comportamento dopo il deployment. Le modifiche possono essere determinate da una funzione di ricompensa oppure dall'esigenza di adattarsi a un nuovo ambiente operativo, ma i dettagli specifici di tali modifiche non possono essere previsti in anticipo. Ad esempio, una piattaforma di e-commerce potrebbe utilizzare un'AI adattiva per raccomandare prodotti agli utenti sulla base del loro comportamento passato e dell'evoluzione delle loro preferenze.

Molti sistemi GenAI, quali i chatbot basati su LLM, vengono sottoposti a deployment come modelli bloccati in fase di esecuzione, ma sono aggiornati periodicamente: si collocano quindi tra i sistemi completamente bloccati e quelli completamente adattivi.

Nella pratica, i sistemi basati sull'AI si distribuiscono lungo un continuum che va dai sistemi completamente deterministici e bloccati, i quali producono lo stesso output per un determinato input, ai sistemi deliberatamente non deterministici e autoapprendenti, che adattano e fanno evolvere il proprio comportamento.

I sistemi basati sull'AI bloccati sono molto più facili da testare rispetto ai sistemi basati sull'AI adattivi, perché sono in larga misura deterministici e quindi i risultati attesi non cambiano. Un sistema basato sull'AI bloccato che viene aggiornato è normalmente considerato un nuovo sistema e richiede un nuovo ciclo di testing. Si noti tuttavia che, sebbene i sistemi basati sull'AI bloccati siano generalmente considerati deterministici, nella pratica le reti neurali di grandi dimensioni possono presentare comportamenti non deterministici a causa di fattori quali i limiti di precisione dei numeri in virgola mobile e le variazioni nell'esecuzione hardware, in particolare quando si utilizzano il calcolo parallelo o le GPU.

Un sistema basato sull'AI adattivo può essere sottoposto a testing rigoroso prima del deployment (ad esempio simulando variazioni delle condizioni ambientali, testando il meccanismo di apprendimento stesso oppure testandone la capacità di adattarsi in modo appropriato in scenari controllati). Questo testing è tuttavia più complesso di quello di un sistema basato sull'AI bloccato, poiché un sistema adattivo può modificare il proprio comportamento durante il testing o in conseguenza del testing.

Poiché non sempre è possibile prevedere i nuovi comportamenti di un sistema basato sull'AI adattivo, tali comportamenti imprevedibili non possono essere testati in anticipo, né è possibile predisporre test case per essi. È possibile creare una test suite automatizzata, focalizzata sulle funzionalità principali del sistema, ed eseguirla ogni volta che il sistema subisce modifiche significative, per verificare che gli adattamenti siano sicuri.

Il testing può essere utilizzato anche per verificare che le prestazioni del sistema non siano peggiorate oltre una determinata soglia. Questo testing può essere eseguito in risposta a modifiche significative del sistema oppure nell'ambito del monitoraggio continuo.

4.1.2 Motivazioni per un Approccio Statistico al Testing dei Sistemi Basati sull'AI

Il testing dei sistemi basati sull'AI presenta sfide specifiche dovute alla loro natura data-driven e probabilistica.

Tra i motivi per cui il testing dei sistemi basati sull'AI richiede un approccio statistico rientrano:

- **Non determinismo** – I sistemi basati sull'AI sono fondamentalmente probabilistici e pertanto presentano spesso un comportamento non deterministico: gli stessi input possono non produrre sempre lo stesso output. Ciò può dipendere da elementi stocastici della loro architettura oppure dall'implementazione di mappature probabilistiche apprese dai dati di addestramento. Di conseguenza, un singolo test, ad esempio un caso in cui un modello classifica erroneamente un gatto come cane, non può rappresentare accuratamente la correttezza funzionale dell'AI complessiva del modello. Per confermare che i risultati dei test superino in modo affidabile questa incertezza, la test suite deve essere sufficientemente ampia da fornire risultati statisticamente significativi.
- **Valutazione delle prestazioni rispetto alla distribuzione** – I modelli AI vengono addestrati su distribuzioni di dati specifiche, che non coincidono esattamente con il loro ambiente operativo. Per valutare come si comporterà un modello in condizioni reali è necessario testare un campione statisticamente significativo di scenari tratti dalle distribuzioni dei dati operativi rilevanti. Ciò conferma che il testing tenga conto della variabilità operativa dei dati e dei conseguenti comportamenti del modello.
- **Gestione dell'incertezza e dei bias** – I sistemi basati sull'AI sono esposti ai bias dei dati e possono produrre previsioni errate con un livello di confidenza elevato. Il testing statistico consente di quantificare e analizzare accuracy, equità e robustezza dell'AI del modello mediante metriche delle prestazioni quali intervalli di confidenza, testing delle ipotesi e analisi degli errori.

- Contesto normativo e di sicurezza fisica – Nei settori regolamentati (ad esempio sanità e trasporti) è spesso necessario dimostrare con un livello di confidenza elevato che un sistema basato sull'AI soddisfi soglie di sicurezza fisica o di equità. I metodi statistici supportano le affermazioni relative all'affidabilità di un modello su un'ampia gamma di scenari, anziché soltanto su esempi specifici.

Un approccio statistico al testing degli MLS probabilistici è descritto in 6.1.3.

4.1.3 Oracoli del Test per i Sistemi Basati sull'AI

Il testing dei sistemi basati sull'AI può essere complesso, in particolare quando occorre determinare i risultati attesi (il problema dell'oracolo del test). Tali difficoltà derivano da diversi fattori intrinseci all'AI:

- Natura probabilistica e non deterministica: gli output dell'AI possono variare anche con input identici. Sebbene molti sistemi (ad esempio nell'apprendimento supervisionato) abbiano un unico valore target corretto, gli output del modello sono probabilistici e la definizione di un oracolo rigoroso superato/fallito richiede spesso l'impostazione di soglie o di intervalli di tolleranza.
- Sviluppo esplorativo e specifiche incomplete: lo sviluppo dell'AI è spesso esplorativo. I requisiti di sistema possono evolvere, essere incompleti o semplicemente mancare, e non fornire le specifiche dettagliate necessarie per generare risultati attesi precisi.
- Complessità delle attività: i sistemi basati sull'AI affrontano spesso attività troppo complesse per una verifica umana diretta, il che rende impraticabile il controllo manuale degli output attesi.
- Soggettività del comportamento: la correttezza del comportamento di un sistema basato sull'AI può essere soggettiva. Ad esempio, le aspettative degli utenti nei confronti degli assistenti virtuali possono variare notevolmente, il che complica la definizione di risultati attesi universalmente condivisi.
- Sistemi autoapprendenti: questi sistemi basati sull'AI aggiornano continuamente i propri modelli interni sulla base dei nuovi dati incontrati dopo il deployment. Ciò fa sì che il comportamento "corretto" del sistema basato sull'AI cambi nel tempo, affinché le risposte del sistema rimangano efficaci e appropriate anche con l'evolversi della definizione di "corretto". Di conseguenza, un insieme iniziale di risultati attesi può rapidamente diventare non valido.

Il problema dell'oracolo del test nei sistemi basati sull'AI può essere affrontato in diversi modi:

- Definizione dei limiti dell'output: i tester possono utilizzare intervalli accettabili concordati, distribuzioni, limiti specificati e tolleranze, ad esempio un'automobile autonoma che si arresta entro una distanza massima.
- Definizione dei limiti ambientali: i tester devono specificare i valori delle condizioni dell'ambiente di test (ad esempio livelli di illuminazione, temperatura, latenza della rete) per garantire che gli output siano prevedibili e ripetibili.
- Consultazione di esperti: gli esperti di dominio possono contribuire a definire i risultati attesi, sebbene le loro opinioni possano differire o risultare fallibili.
- Testing specializzato: il testing A/B, il testing back-to-back e il testing metamorfico, tra gli altri, possono valutare i sistemi basati sull'AI confrontandone i comportamenti o verificandone le proprietà, spesso senza richiedere output attesi espliciti per ogni caso.

- Oracoli proxy: utilizzare sistemi o modelli secondari (compresi altri sistemi AI) per valutare o validare gli output quando non sono disponibili risultati attesi diretti, ad esempio addestrando un modello proxy su dati etichettati affinché produca previsioni per test non etichettati.

4.2 Testing dell'AI Generativa e dei Large Language Model

Questa sezione descrive metodi pratici per validare i sistemi GenAI, trattando la valutazione black-box, il red teaming e alcune tecniche pratiche. Evidenzia le sfide derivanti dalla varietà degli input, dai parametri configurabili e dalle finestre di contesto, e affronta misure di qualità sia funzionali sia non funzionali mediante benchmark ed esercizi mirati. Per maggiori dettagli sull'utilizzo dei sistemi GenAI per il testing, vedi il syllabus CT-GenAI [CT-GenAI].

4.2.1 Testing dell'AI Generativa

Il testing della GenAI comporta la valutazione della correttezza, della coerenza e della creatività dei suoi output, comprese l'originalità e la novità dei risultati generati, e la verifica del soddisfacimento dei requisiti specificati, sia funzionali sia non funzionali. Poiché i sistemi GenAI possono produrre testo, immagini, video e audio, le strategie di test devono essere adattate per valutare le caratteristiche di qualità di tali contenuti.

Un approccio comune è il testing black-box, nel quale i tester forniscono al sistema input diversi (prompt, immagini, dati parziali) e valutano i risultati dei test (vedi red teaming, 4.2.2). Vengono considerati fattori quali chiarezza, originalità e rispetto delle regole specifiche del dominio. Questo approccio, manuale o automatizzato, è particolarmente utile per le applicazioni rivolte agli utenti finali, quali chatbot o strumenti di progettazione, nelle quali la soddisfazione dell'utente dipende dall'utilità e dalla plausibilità del contenuto generato.

Una sfida significativa del testing di un modello GenAI è il "problema dell'esplosione degli input", per cui gli input possono essere estremamente eterogenei e complessi da controllare. Ad esempio, sono presenti prompt di sistema facoltativi e un prompt dell'utente, che può includere enormi quantità di dati eterogenei; inoltre è possibile accedere alla GenAI anche tramite un'API. Occorre considerare anche numerosi parametri, quali la temperature e il numero massimo di token, che influenzano l'output. Anche la finestra di contesto, che conserva le parti precedenti di una conversazione, influisce sull'output generato.

In molte situazioni la valutazione dell'output può richiedere una review manuale; tuttavia la determinazione del superamento o del fallimento di un test dipende da criteri di valutazione qualitativi definiti nei requisiti. In alternativa, spesso è possibile utilizzare un secondo sistema GenAI per determinare automaticamente il risultato del test. Ad esempio, la correttezza di un'immagine generata da un sistema GenAI può essere verificata mediante un sistema di riconoscimento delle immagini. Questi approcci devono essere utilizzati con cautela, poiché possono riprodurre bias o errori simili.

Per i sistemi GenAI il testing non-funzionale può essere necessario quanto il testing funzionale. Comprende la valutazione dell'utilizzo delle risorse, sia durante l'inferenza sia durante l'addestramento, per verificare l'efficienza operativa e la convenienza economica. Le misure comprendono l'utilizzo di CPU e GPU, il consumo di memoria, la larghezza di banda della rete e i tempi di risposta.

Le suite di benchmark forniscono framework di valutazione standardizzati per esaminare le capacità della GenAI. Questi dataset selezionati e le attività associate consentono un confronto coerente tra modelli differenti, misurando aspetti diversi, dalla comprensione del linguaggio alle capacità di ragionamento e di programmazione. Misurando la GenAI rispetto a questi benchmark è possibile identificare in modo sistematico e riproducibile le aree di miglioramento.

4.2.2 Red Teaming

Il Red Teaming (RT) è una forma sistematica, spesso black-box, di attacco basato sui guasti che sonda un sistema basato sull'AI per identificarne le capacità dannose, in particolare negli output. Ispirandosi a pratiche storiche quali i wargame militari e i tiger team della NASA, il RT dell'AI comporta l'"attacco" di un sistema basato sull'AI con l'obiettivo di indurlo deliberatamente a produrre risultati dannosi o indesiderati, quali violazioni della privacy, l'espressione di opinioni razziste o istruzioni per eseguire attacchi chimici, biologici, radiologici o nucleari (CBRN). Una volta identificate, tali capacità vengono utilizzate per aggiornare e rafforzare il sistema, rendendolo in futuro meno incline a produrre output di questo tipo. Sebbene il RT possa essere applicato a qualsiasi sistema basato sull'AI, è particolarmente critico per la GenAI, a causa dell'enorme varietà di input e output possibili, che crea un ampio "spazio di attacco". Il RT copre normalmente l'intero sistema basato sull'AI end-to-end, ma può essere applicato anche al solo modello AI.

Molte organizzazioni focalizzano il RT sulle vulnerabilità di sicurezza e di sicurezza fisica; esso può tuttavia essere utilizzato anche per rilevare capacità dannose in altri ambiti, quali affidabilità, privacy, equità, bias e generazione di disinformazione. Per alcuni tipi di sistemi basati sull'AI il RT sta diventando sempre più un'aspettativa normativa, come si osserva in framework quali l'EU AI Act [EU AI Act]. In quanto approccio di valutazione adattivo e dinamico, nel quale i prompt possono essere aggiornati immediatamente in risposta agli output, il RT integra approcci statici quali il benchmarking, testando i sistemi in condizioni estreme e inattese.

Quando il RT viene utilizzato per valutazioni di sicurezza, il sistema viene testato per identificare vulnerabilità ad attacchi esterni, considerando sia fattori di sicurezza non specifici dell'AI sia fattori specifici dell'AI, quali gli attacchi di prompt injection indiretta e l'occultamento di contenuti malevoli nei documenti utilizzati dalla RAG. Mentre il RT di sicurezza tende a focalizzarsi sugli input malevoli, nelle valutazioni di sicurezza fisica e di altri aspetti l'obiettivo è spesso identificare in che modo il sistema possa creare output dannosi durante l'uso ordinario, ad esempio generando consigli medici non sicuri, senza alcun intento avversario da parte dell'utente.

Il RT è più efficace se condotto prima del deployment del sistema, ma dopo il completamento delle valutazioni interne iniziali della qualità. L'attività centrale consiste spesso nel prompting interattivo, nel quale i red teamer conducono dialoghi multi-turno (ad esempio di 15-20 turni) per provocare comportamenti difettosi o contrari alle policy. L'approccio comprende normalmente:

- 1) La composizione di un team eterogeneo di tester, per coprire un'ampia gamma di prospettive e di vettori di attacco.
- 2) La messa a disposizione dell'accesso al sistema basato sull'AI in un ambiente di test sicuro.
- 3) Il prompting del sistema per identificare le vulnerabilità, mediante esplorazione aperta oppure utilizzando checklist.
- 4) L'analisi dei failure identificati, per comprendere le minacce.
- 5) La creazione di dataset a partire da tali minacce, per supportare le mitigazioni e i miglioramenti del sistema.

Per conseguire una copertura completa mediante il RT, le organizzazioni adottano diverse strategie che vanno oltre i piccoli team di esperti. Vi rientrano approcci manuali, quali la generazione di prompt in crowdsourcing, e approcci automatizzati, ad esempio l'utilizzo di un LLM per generare numerosi prompt di attacco, i cui output vengono poi verificati da un altro LLM. Gli approcci ibridi combinano la creatività dei tester umani con la scalabilità dell'automazione.

Il RT, che è proattivo e focalizzato sulla fase precedente al deployment, integra il Blue Teaming, che comporta il monitoraggio difensivo continuo e in tempo reale e il filtraggio degli input diretti a un sistema basato sull'AI operativo, per proteggerlo dagli attacchi. Le informazioni ricavate dal RT possono essere utilizzate per migliorare il monitoraggio e i filtri utilizzati nel Blue Teaming.

4.2.3 Esercizio Pratico: Testing Esplorativo di un Large Language Model

Gli studenti eseguiranno il testing esplorativo di un LLM. Riceveranno una scheda di sessione di testing esplorativo focalizzata sul testing della capacità dell'LLM di generare test case utilizzando l'analisi dei valori limite a 2 valori e a 3 valori. Nell'ambito del debriefing della sessione verificheranno la correttezza e la completezza dell'output.

4.3 Livelli di Test e Sistemi di Machine Learning

Gli MLS richiedono livelli di test specializzati per affrontare i rischi specifici del ML: il testing dei dati di input e il testing del modello ML. Restano inoltre rilevanti i livelli di test convenzionali, tra cui il testing di componente, il testing di integrazione dei componenti, il testing di sistema e il testing di accettazione.

4.3.1 Livelli di Test per i Sistemi di Machine Learning

I componenti non-AI di un MLS possono essere testati utilizzando i livelli di test convenzionali. I modelli ML e gli MLS richiedono inoltre ulteriore testing per affrontare i rischi specifici associati al ML.

Per affrontare i rischi specifici del ML vengono utilizzati due livelli di test specializzati:

- Testing dei dati di input (vedi Capitolo 5): riguarda i dati di addestramento utilizzati per addestrare un modello ML e i dati di produzione utilizzati da un MLS per generare una previsione nell'ambiente operativo.
- Testing del modello ML (vedi Capitolo 6): riguarda il testing dei modelli ML, che costituiscono l'output finale del workflow ML (vedi 3.1.2).

Non tutti i rischi associati al ML possono essere affrontati mediante questi due livelli di test specifici del ML. In base ai rischi percepiti sono normalmente necessari anche i livelli di test seguenti:

- Testing di componente: applicabile a qualsiasi componente non-AI, quali la user interface, la pipeline di dati e i componenti di comunicazione.
- Testing di integrazione dei componenti: comprende il testing volto a verificare che gli input provenienti dalla pipeline di dati siano ricevuti dal modello come previsto e che le previsioni generate dal modello vengano scambiate con i componenti di sistema pertinenti (ad esempio la user interface) e utilizzate correttamente. Quando l'AI è fornita come servizio (vedi 1.1.6), il testing delle API del servizio fornito viene eseguito nell'ambito del testing di integrazione dei componenti.
- Testing di sistema: comprende il testing confermativo volto a verificare che l'efficienza delle prestazioni funzionali di ML rilevata durante il testing iniziale del modello non venga compromessa quando il modello è integrato in un sistema completo. Questo testing è particolarmente importante quando il modello ML è stato modificato deliberatamente (ad esempio comprimendo una DNN per ridurne le dimensioni). Comprende anche il testing non-funzionale,

ad esempio il testing dell'efficienza delle prestazioni relativo al tempo necessario perché un sistema basato sull'AI completo fornisca una previsione.

- Testing di integrazione dei sistemi: si focalizza sulla verifica delle interfacce e degli scambi di dati tra il sistema basato sull'AI e sistemi o servizi esterni, utilizzando un ambiente rappresentativo delle condizioni operative.
- Testing di accettazione: quando l'AI è utilizzata come servizio, può essere necessario il testing di accettazione per determinare l'idoneità del servizio al sistema previsto e verificare, ad esempio, se siano stati conseguiti i criteri di efficienza delle prestazioni funzionali di ML.

4.3.2 Testing Basato sul Rischio dei Sistemi di Machine Learning

Il testing basato sul rischio dovrebbe essere applicato a tutti i sistemi, indipendentemente dalla presenza di componenti AI. La maggior parte dei framework normativi, già pubblicati o in corso di sviluppo, richiede un approccio basato sul rischio allo sviluppo e alla gestione dei sistemi basati sull'AI. Poiché i sistemi basati sull'AI presentano rischi specifici, il loro testing differisce da quello dei sistemi non basati sull'AI.

Non esiste un modo standard per classificare i rischi del ML; i rischi specifici del ML sono tuttavia associati sia allo sviluppo dell'MLS (rischi di progetto) sia all'MLS stesso (rischi di prodotto). Un modo per classificare questi rischi consiste nell'utilizzare il workflow ML, suddividendolo in tre aree principali:

- Sviluppo – riguarda l'algoritmo ML, lo sviluppo del modello e il framework di sviluppo ML. Esempi di rischi di progetto comprendono la selezione non ottimale dell'algoritmo, una scelta inadeguata dell'approccio di valutazione e le vulnerabilità di sicurezza del framework. Per maggiori dettagli vedi il Capitolo 7.
- Dati di input – riguarda la fornitura dei dati di addestramento a supporto del ML e la fornitura dei dati di produzione utilizzati dal modello nel proprio ambiente operativo. Esempi di rischi di prodotto comprendono dati di addestramento affetti da bias, difetti della pipeline di dati e dati di addestramento non rappresentativi. Per maggiori dettagli vedi il Capitolo 5.
- Modello – riguarda il modello ML generato. Esempi di rischi di prodotto comprendono il mancato conseguimento delle misure richieste di efficienza delle prestazioni funzionali di ML, un modello affetto da overfitting e la suscettibilità agli adversarial example. Per maggiori dettagli vedi il Capitolo 6.

Un altro modo per classificare i rischi associati all'AI si basa sulle caratteristiche di qualità definite nella norma ISO/IEC 25059.

Sono disponibili diverse forme di testing che affrontano tali rischi e sono progettate specificamente per il testing degli MLS: ad esempio il testing della pipeline di dati, l'adversarial testing e la review dell'idoneità dell'algoritmo o del modello. Questo syllabus ne tratta diverse nei Capitoli 5, 6 e 7.

5 Testing dei Dati di Input per i Sistemi di Machine Learning – 180 minuti

Parole Chiave

Testing della pipeline di dati, test di rappresentatività dei dati, testing dei vincoli di dataset, testing dei dati di input, testing di correttezza delle etichette, review, testing per bias

Parole Chiave Specifiche dell'AI

Analisi degli impatti disparati, annotazione multipla

Obiettivi di Apprendimento per il Capitolo 5:

5.1 Testing dei Dati di Input per i Sistemi di Machine Learning

- AI-5.1.1 (K2) Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo ai dati di input di un sistema di machine learning
- AI-5.1.2 (K2) Spiegare come eseguire il testing per bias
- AI-5.1.3 (K2) Riassumere le diverse forme di testing della pipeline di dati
- AI-5.1.4 (K2) Spiegare come eseguire il test di rappresentatività dei dati
- AI-5.1.5 (K3) Applicare il testing dei vincoli di dataset
- AI-5.1.6 (K2) Spiegare il testing di correttezza delle etichette
- HO-5.1.7 (H2) Effettuare il testing dei dati di input per i dataset ML

5.1 Testing dei Dati di Input per i Sistemi di Machine Learning

L'obiettivo del testing dei dati di input è confermare che i dati utilizzati dall'MLS per l'addestramento, il testing e la previsione abbiano una qualità sufficiente (vedi 3.2). Comprende review, tecniche statistiche (ad esempio il testing per bias dei dati), l'EDA dei dati di addestramento e il testing statico e dinamico della pipeline di dati.

5.1.1 Rischi Relativi ai Dati di Input e Relative Mitigazioni

La tabella seguente elenca esempi di rischi relativi ai dati di input e il corrispondente testing che potrebbe essere utilizzato per la mitigazione del rischio:

Rischi Potenziali	Possibili Mitigazioni dei Rischi
Difetti nei dati di addestramento che determinano bias Problemi nell'algoritmo, nel modello o nel framework di sviluppo ML che introducono iniquità sistemica	Testing per bias – vedi 5.1.2
Dati di addestramento acquisiti da fonti non affidabili Gestione inadeguata dei dati	Test di provenienza dei dati
Dati di addestramento avvelenati	Testing A/B – vedi 6.1.9 Test di provenienza dei dati EDA – vedi 3.2.1 Attacchi nell'ambito del red teaming – vedi 4.2.2
Dataset internamente incoerente Dati fuori intervallo Tipi di dati errati	Testing dei vincoli di dataset – vedi 5.1.5
Selezione non ottimale delle feature	Testing delle feature
Dataset sbilanciato a causa di una copertura insufficiente di tutte le classi target Dataset distorto a causa dell'aumento dei dati Dati mancanti	Test di rappresentatività dei dati – vedi 5.1.4

Dati di addestramento focalizzati su un sottoinsieme di tutti i casi d'uso Intervallo completo dei valori non coperto nel dataset	
Linee guida di etichettatura inadeguate Dati ambigui Annotazione inadeguata che determina etichette inaccurate o incoerenti	Testing di correttezza delle etichette – vedi 5.1.66
Progettazione o integrazione inadeguata che determina failure della pipeline di dati Difetti della qualità dei dati che compromettono gli output della pipeline di dati Peggioramento delle prestazioni nell'utilizzo operativo della pipeline di dati Violazioni della sicurezza o modifiche non controllate che incidono sulla pipeline di dati	Testing della pipeline di dati – vedi 5.1.3

5.1.2 Testing per Bias

Il bias in un MLS indica differenze di trattamento non casuali e inique basate su attributi sensibili quali genere, età o razza; spesso è illegale e rende il sistema discriminatorio.

Il testing per bias in un MLS richiede di comprenderne le potenziali fonti, che comprendono principalmente:

- Difetti nei dati di addestramento, quali mancanza di rappresentatività, distorsioni storiche o avvelenamento deliberato (bias dei dati).
- Difetti nell'algoritmo, nel modello o nel framework di sviluppo che introducono iniquità sistemica, ad esempio un algoritmo che utilizza una soglia decisionale per il punteggio di credito in un sistema di approvazione dei prestiti (bias algoritmico).

Gli approcci del test per rilevare il bias comprendono:

- Review dell'intero workflow ML e, in particolare, della preparazione dei dati, per identificarne la potenziale capacità di introdurre bias.
- Review della documentazione del dataset, per identificare e mitigare le potenziali fonti di iniquità esaminando come i dati siano stati raccolti e annotati e quali popolazioni siano rappresentate.

- Analisi statica sia dei programmi di preparazione dei dati sia del codice di implementazione del modello, per identificare anti-pattern o una gestione non corretta degli attributi sensibili.
- EDA dei dati di addestramento mediante metodi di visualizzazione e clustering, per evidenziare dati sbilanciati, distribuzioni distorte o raggruppamenti anomali rispetto ai diversi attributi sensibili.
- Testing dinamico, per contribuire al rilevamento del bias in un modello ML fornendo al sistema un dataset rappresentativo e notoriamente privo di bias e analizzandone le previsioni alla ricerca di differenze statisticamente significative nei risultati tra i diversi gruppi sensibili. In questo modo si identifica il bias negli output del modello, indipendentemente dal fatto che sia stato introdotto durante l'addestramento sui dati o durante lo sviluppo del modello.
- Testing di correttezza delle etichette (vedi 5.1.6), per identificare etichettature errate che causano l'apprendimento di associazioni non corrette tra attributi e risultati per determinati attributi sensibili.
- Analisi degli impatti disparati:
 1. Identificare gli attributi sensibili per il modello.
 2. Generare controfattuali per gli attributi sensibili (ad esempio scenari nei quali, in una richiesta di prestito, il genere viene modificato da maschile a femminile).
 3. Generare l'output del modello sottoponendogli i controfattuali.
 4. Analizzare i risultati di più test per ottenere un risultato statisticamente significativo, che determini se la modifica di un attributo sensibile produca una variazione dei risultati del modello, segnalando la presenza di bias.

Nota: l'analisi degli impatti disparati può essere applicata anche a combinazioni di attributi sensibili, consentendo di identificare bias nascosti associati a tali combinazioni. Occorre tuttavia verificare che gli esempi controfattuali non siano irrealistici, poiché in tal caso il modello potrebbe rispondere a essi anziché al bias sottostante.

5.1.3 Testing della Pipeline di Dati

Un testing della pipeline di dati efficace è essenziale non soltanto per confermare l'affidabilità e le prestazioni dei sistemi data-driven, ma anche per mantenere un'elevata qualità dei dati lungo l'intero workflow ML (vedi 3.1.2).

Viene adottato un approccio a livelli, che inizia con le review della progettazione della pipeline di dati durante la fase di progettazione.

Il testing di componente comprende il testing dei componenti di acquisizione dei dati, degli script di trasformazione e delle interfacce dei sensori. Questo testing utilizza review del codice, analisi statica e test specifici dell'hardware per verificare l'affidabilità dell'acquisizione dei dati. I test di componente validano la logica di trasformazione dei dati, testano l'implementazione delle regole di validazione dei dati, confermano una gestione robusta degli errori e verificano la presenza di vulnerabilità che potrebbero essere sfruttate per introdurre malware o dati avvelenati.

Il testing di integrazione dei componenti verifica il flusso continuo dei dati attraverso le interfacce interne e la corretta interpretazione dei dati durante il loro passaggio nella pipeline. Rileva difetti derivanti da interfacce non corrispondenti o da presupposti errati tra i componenti.

Il testing di sistema valuta la pipeline completamente assemblata, iniziando con lo smoke testing per confermarne le funzionalità di base. Il testing funzionale verifica la conformità della pipeline ai requisiti

specificati, comprese le trasformazioni e l'instradamento dei dati. Il testing non-funzionale valuta le prestazioni sotto carico, la scalabilità rispetto all'aumento dei volumi di dati e le misure di sicurezza per proteggere l'integrità dei dati. Il testing mediante fault injection misura la robustezza della pipeline simulando input di dati difettosi e valutando la capacità del sistema di mantenere l'integrità dei dati in presenza di dati inattesi o danneggiati. Il testing back-to-back (vedi 6.1.10) confronta la pipeline operativa con la pipeline di addestramento, per verificarne la coerenza funzionale.

Il testing di integrazione dei sistemi verifica la corretta interazione tra la pipeline di dati e i sistemi o servizi esterni, comprese le fonti di dati, le piattaforme di archiviazione, gli strumenti di monitoraggio e i consumer a valle, quali i modelli ML.

Il testing in produzione viene eseguito sul sistema operativo. Il testing back-to-back verifica che le prestazioni siano coerenti o migliorate rispetto alle versioni precedenti. Il testing A/B (vedi 6.1.9) consente di confrontare le nuove iterazioni della pipeline con le baseline, validando i miglioramenti e confermando al contempo l'assenza di peggioramenti nel flusso di dati reale. È possibile integrare strumenti per monitorare e osservare continuamente, in tempo reale, il comportamento e le prestazioni del modello e i potenziali difetti.

Le review della gestione della configurazione contribuiscono a verificare che, per l'addestramento, il testing e la produzione, vengano utilizzate le versioni corrette del codice della pipeline, delle configurazioni e dei dataset.

Infine, le strategie di test devono essere allineate allo scopo della pipeline. Le pipeline di addestramento, spesso prototipi esplorativi, hanno priorità differenti rispetto alle robuste pipeline operative. Il testing delle pipeline di addestramento può focalizzarsi sull'integrità dei dati, mentre il testing delle pipeline operative attribuisce priorità ad affidabilità, prestazioni e manutenibilità.

5.1.4 Test di Rappresentatività dei Dati

Il test di rappresentatività dei dati determina in quale misura le caratteristiche dei dataset utilizzati per l'addestramento, la validazione e il testing dei modelli ML corrispondano ai dati del mondo reale che il modello ML incontrerà durante il funzionamento. Questo testing affronta diverse forme di rappresentazione inadeguata, tra cui dataset distorti, dati mancanti, distribuzioni sproporzionate delle feature, copertura insufficiente degli scenari operativi e rappresentazione sbilanciata delle classi (vedi 5.1.1).

Questo testing comprende normalmente i passi seguenti:

1. Definire la popolazione target:

- Comprendere i casi d'uso previsti e il contesto operativo dell'MLS
- Analizzare le caratteristiche degli utenti finali e degli ambienti operativi
- Identificare le distribuzioni previste dei dati operativi e gli edge case critici mediante:
 - la consultazione di esperti di dominio, per comprendere i pattern dei dati del mondo reale
 - l'analisi dei dati provenienti da sistemi esistenti o da applicazioni simili
 - l'utilizzo di dataset di benchmark provenienti da fonti affidabili (ad esempio NIST, database di settore)

- Applicare il campionamento stratificato ai dati di riferimento rappresentativi, per creare una baseline che copra tutti i sottogruppi rilevanti del dominio target

2. Analizzare le caratteristiche dei dati:

- Applicare l'EDA (vedi 3.2.1) a:
 - i dataset di addestramento/test dei quali si valuta la rappresentatività
 - il dataset di riferimento che rappresenta i dati operativi previsti
 - visualizzare le distribuzioni mediante istogrammi, diagrammi a dispersione e altre tecniche grafiche
- Esaminare le relazioni tra le feature e, in particolare, le correlazioni, per identificare i pattern da preservare
- Identificare potenziali anomalie, lacune o concentrazioni insolite nei dati

3. Applicare tecniche di valutazione statistica:

- Utilizzare test statistici formali, quali il chi quadrato e il test di Kolmogorov-Smirnov, per confrontare le distribuzioni [STATS]
- Verificare la presenza di squilibri nei dati, in particolare nei problemi di classificazione
- Verificare una copertura adeguata sia degli scenari tipici sia degli edge case e dei casi limite

Il test di rappresentatività dei dati dovrebbe essere eseguito prima dell'addestramento del modello, per evitare di costruire modelli su dati non rappresentativi. Dopo il deployment, le proprietà dei dati operativi di input dovrebbero essere monitorate continuamente per rilevare variazioni che potrebbero indicare una deriva dei dati rispetto alle distribuzioni originarie dei dati di addestramento (vedi 6.1.7 sul drift testing).

5.1.5 Testing dei Vincoli di Dataset

Il testing dei vincoli di dataset verifica se i dati presenti in un dataset rispettino regole o vincoli predefiniti. L'obiettivo è confermare l'integrità e la coerenza dei dati utilizzati nel ML. Ad esempio, è possibile eseguire un test della coerenza dei valori di un dataset.

I vincoli sui dati sono normalmente definiti negli schemi dei database. Uno schema di database definisce la struttura, i tipi e le relazioni dei dati. Analogamente, per i dataset ML è possibile definire un insieme di vincoli che funge da modello logico dei dati del dataset e che dovrebbe essere soddisfatto se i dati sono corretti. Esistono diversi modi di classificare i vincoli di dataset. Un vincolo può essere applicato a un singolo valore di un attributo presente in una singola istanza (vincolo a valore singolo), ad esempio:

- Mancante – verifica la presenza di valori e attributi mancanti.
- Intervallo – verifica che un valore rientri in un determinato intervallo.
- Tipo – verifica che il valore di un attributo corrisponda al tipo specificato (ad esempio, se un attributo è specificato come numero intero, il valore fornito non deve essere una stringa o un numero reale).

In alternativa, un vincolo può essere applicato a più valori, normalmente considerando i valori di un singolo attributo in diverse istanze (vincolo multivalore), ad esempio:

- **Somma** – verifica che la somma di tutti i valori sia uguale, superiore o non superiore a un valore specificato (ad esempio, il valore totale di tutti i punti assegnati per una gara di Formula 1 non può superare 102 e deve superare 50,5 punti).
- **Conteggio** – verifica che il numero di tutti i valori non nulli degli attributi o delle istanze sia uguale, superiore o non superiore a un valore specificato.
- **Duplicato** – verifica la presenza, in un dataset, di valori di attributi o istanze identici o quasi identici e impone un limite al numero consentito (spesso pari a zero).
- **Utilità** – verifica che un attributo contenga alcuni valori ripetuti, poiché un attributo in cui ogni voce è univoca (come un ID o un timestamp) normalmente non offre pattern utili che un modello ML possa apprendere.
- **Valore anomalo** – identifica i valori che possono essere considerati valori anomali dal punto di vista statistico.

Una forma particolare di vincolo multivalore può confrontare valori differenti (vincolo di confronto), ad esempio:

- **Maggiore di** – verifica che il valore di un attributo sia maggiore del valore di un secondo attributo (ad esempio, il numero di linee di codice di un programma supera il numero delle sue linee di codice contenenti difetti).
- **Correlazione** – verifica che i valori di un attributo siano correlati ai valori di un secondo attributo (ad esempio, tutti gli studenti il cui valore dell'attributo "punteggio" è superiore alla media di almeno 1,33 deviazioni standard hanno anche il valore "A" per l'attributo "voto").

Il testing dei dati rispetto ai vincoli definiti può essere eseguito manualmente. Tuttavia, la portata dell'attività e le dimensioni normalmente elevate del dataset richiederebbero di automatizzarlo nell'ambito della pipeline di dati. Se implementato nella pipeline, lo strumento che esegue il testing dei vincoli di dataset può fornire report ai data scientist (per le anomalie nei dati di addestramento) oppure al personale operativo (per i problemi nei dati operativi).

5.1.6 Testing di Correttezza delle Etichette

La correttezza delle etichette dei dati è essenziale nell'apprendimento supervisionato. Etichette inaccurate o incoerenti compromettono direttamente le prestazioni e la capacità di generalizzazione dei modelli ML.

Gli approcci comuni al testing di correttezza delle etichette dei dati comprendono:

- **Review di esperti**: esperti di dominio o annotatori formati eseguono manualmente la review di un campione di dati etichettati. Valutano l'accuratezza delle etichette utilizzando la propria conoscenza del dominio e le linee guida.
- **Annotazione multipla**: i punti dati vengono etichettati indipendentemente da più annotatori e confrontati mediante una forma di testing back-to-back (vedi 6.1.10). I disaccordi evidenziano difetti che devono essere analizzati e risolti. L'accordo tra annotatori (IAA, Inter-Annotator Agreement) può essere misurato mediante metriche quali il kappa di Cohen o la semplice percentuale di accordo [STATS]. Valori IAA bassi possono indicare difetti nelle linee guida di etichettatura, dati ambigui o annotazioni inadeguate.

- Prioritizzazione basata sul rischio per la review e l'annotazione: sia le review di esperti sia le annotazioni multiple possono essere focalizzate mediante un approccio basato sul rischio, per identificare i campioni da sottoporre a review e ad annotazione multipla. La prioritizzazione può basarsi sulla probabilità di etichettatura errata, ad esempio in presenza di punti dati ambigui (nei quali non è chiara la categoria di appartenenza oppure che si trovano vicino a un confine tra classi), e sui dati con la maggiore probabilità di incidere sul successo o sulla sicurezza fisica dell'applicazione.
- Analisi della distribuzione dei dati: quando esistono dataset confrontabili, il confronto tra la distribuzione delle etichette del dataset sotto test e quella di dataset simili può evidenziare anomalie ed etichette potenzialmente errate.
- Test automatizzati basati su regole: per determinate attività è possibile implementare test automatizzati basati su regole o vincoli predefiniti relativi alle etichette. Ad esempio, verificare che i bounding box (i rettangoli utilizzati per localizzare gli oggetti nelle immagini) non si sovrappongano e non si estendano oltre i limiti dell'immagine.
- Analisi della loss del modello: i punti dati che presentano una loss elevata durante l'addestramento del modello – ossia per i quali le previsioni del modello si discostano significativamente dalle etichette effettive – possono indicare un'etichettatura errata. Una loss elevata riflette un errore significativo, indica che il modello ha difficoltà ad apprendere l'etichetta assegnata e segnala un potenziale difetto.
- Analisi del punteggio di confidenza del modello: i punti dati per i quali un modello addestrato produce previsioni con una confidenza bassa potrebbero essere etichettati in modo errato, essere ambigui oppure trovarsi al di fuori della distribuzione dei dati di addestramento del modello.

L'utilizzo di una combinazione di approcci è spesso la soluzione più efficace. Ad esempio, le review di esperti sono utili per definire fin dalle prime fasi linee guida chiare di etichettatura. I punteggi IAA ottenuti dalle annotazioni multiple possono quindi perfezionare il processo di etichettatura. In seguito, gli approcci basati sul modello possono identificare ulteriori potenziali difetti delle etichette durante l'evoluzione del modello.

5.1.7 Esercizio Pratico: Testing dei Dati di Input

Per un determinato dataset (ad esempio dati strutturati, tabellari), effettuare il testing dei dati di input per aspetti quali dati mancanti, dati duplicati e valori anomali.

6 Testing del Modello per i Sistemi di Machine Learning – 225 minuti

Parole Chiave

Testing A/B, adversarial testing, testing back-to-back, deriva concettuale, deriva dei dati, drift testing, testing metamorfico, efficienza delle prestazioni funzionali di ML, testing del modello ML, review

Parole Chiave Specifiche dell'AI

Overfitting, underfitting

Obiettivi di Apprendimento per il Capitolo 6:

6.1 Testing del Modello per i Sistemi di Machine Learning

- AI-6.1.1 (K2) Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo ai modelli ML
- AI-6.1.2 (K2) Spiegare lo scopo e il focus della review della documentazione del modello ML
- AI-6.1.3 (K2) Spiegare come viene eseguito il testing dell'efficienza delle prestazioni funzionali di ML per i sistemi probabilistici di machine learning
- AI-6.1.4 (K2) Riassumere l'adversarial testing dei sistemi di machine learning
- AI-6.1.5 (K3) Utilizzare il testing metamorfico per ricavare test case per un determinato scenario
- HO-6.1.6 (H2) Applicare il testing metamorfico
- AI-6.1.7 (K2) Spiegare come viene utilizzato il drift testing sui sistemi di machine learning operativi
- AI-6.1.8 (K2) Spiegare come overfitting e underfitting vengono rilevati mediante il testing
- AI-6.1.9 (K2) Spiegare come viene utilizzato il testing A/B nel contesto dei sistemi di machine learning
- AI-6.1.10 (K2) Spiegare come viene utilizzato il testing back-to-back nel contesto dei sistemi di machine learning

6.1 Testing del Modello per i Sistemi di Machine Learning

Il testing del modello ML comporta la gestione di rischi specifici. Questi comprendono rischi funzionali quali bias, overfitting e vulnerabilità avversarie, rischi non funzionali quali la mancanza di robustezza dell'AI e di efficienza delle prestazioni, e rischi relativi al deployment.

6.1.1 Rischi Relativi ai Modelli di Machine Learning e Relative Mitigazioni

La tabella seguente elenca esempi di rischi relativi ai modelli ML e il corrispondente testing che potrebbe essere utilizzato per la mitigazione del rischio:

Rischio Potenziale	Possibile Mitigazione del Rischio
Modello ML affetto da bias o iniquo	Testing per bias – vedi 5.1.2
Modello non etico	Testing etico di sistema
Adversarial example	Adversarial testing – vedi 6.1.4
Modello affetto da overfitting	Testing per l'overfitting – vedi 6.1.8
Modello affetto da underfitting	Testing per l'underfitting – vedi 6.1.8
Deriva dei dati non accettabile Deriva concettuale non accettabile	Drift testing – vedi 6.1.7
Il modello causa effetti collaterali	Testing degli effetti collaterali
Il modello presenta reward hacking	Testing del reward hacking
Difetto dell'API del modello	Testing delle API – vedi 7.1.2
Mancato conseguimento delle misure richieste delle prestazioni del modello ML (ad esempio carenza di accuracy o di recall)	Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3
Incorrettezza funzionale Difetti non funzionali	Testing metamorfico – vedi 6.1.5
Problema dell'oracolo del test	Testing metamorfico – vedi 6.1.5 Testing back-to-back – vedi 6.1.10 Testing A/B – vedi 6.1.9
Requisiti di sistema inadeguati	Review dei requisiti Red teaming – vedi 4.2.2

	Testing esplorativo
Mancanza di robustezza dell'AI del modello dovuta a input inattesi	Adversarial testing – vedi 6.1.4 Fuzz testing
Efficienza delle prestazioni del modello ML inadeguata	Performance testing
Documentazione del modello inadeguata (ad esempio funzione, accuracy, interfaccia)	Review della documentazione del modello – vedi 6.1.2
Gli aggiornamenti del modello introducono difetti	Testing back-to-back – vedi 6.1.10
Gli aggiornamenti del modello riducono l'efficienza delle prestazioni funzionali del modello ML	Testing A/B – vedi 6.1.9
Il deployment del modello aggiornato causa un failure immediato	Smoke testing
Il deployment del modello aggiornato causa una regressione	Regression testing
Vulnerabilità di sicurezza Vulnerabilità di sicurezza fisica Violazioni della privacy Output dannosi o indesiderati (ad esempio opinioni razziste, indicazioni pericolose)	Red teaming – vedi 4.2.2

6.1.2 Documentazione e Review del Modello di Machine Learning

Una documentazione completa dei modelli ML non è una semplice formalità: è una risorsa essenziale. A differenza dei sistemi nei quali il codice sorgente può essere ispezionato direttamente, gli MLS presentano sfide specifiche dovute alla limitata comprensibilità del codice generato automaticamente, alla natura intrinsecamente black-box dei modelli e alla loro dipendenza dai dati. I modelli vengono inoltre aggiornati frequentemente: la documentazione diventa quindi lo strumento principale attraverso il quale sviluppatori, tester ed enti regolatori possono comprendere, valutare e considerare affidabili i sistemi basati sull'AI. La trasparenza svolge un ruolo centrale nel consentire tale comprensione, permettendo agli stakeholder di tracciare il comportamento del modello, la logica decisionale e la provenienza dei dati lungo l'intero ciclo di vita dell'AI.

Una documentazione standardizzata migliora la comunicazione, supporta decisioni informate e verifica la qualità e la manutenibilità dei modelli ML. È sempre più importante ai fini della conformità normativa, come dimostrano framework quali l'EU AI Act, che sottolinea gli obblighi di trasparenza e richiede una documentazione chiara delle decisioni del modello, dei suoi limiti e delle misure di interpretabilità. Per i

sistemi ad alto rischio il superamento di un audit della documentazione costituisce spesso un prerequisito per il deployment; per tutti i sistemi una review approfondita contribuisce a verificarne la qualità.

Esistono diversi framework di documentazione, tra cui le Model Card, che forniscono panoramiche concise degli utilizzi previsti di un modello, dei risultati della valutazione e delle considerazioni etiche [MODEL_DOC], e i Datasheets for Datasets, che offrono formati standardizzati per descrivere i dataset, comprese motivazione, composizione, processo di raccolta e utilizzi [DATA_DOC].

L'elenco seguente descrive i contenuti normalmente attesi per una documentazione completa del modello ML ed è strutturato in modo da poter essere utilizzato come checklist pratica sia dagli sviluppatori sia dai tester, per contribuire a conseguire completezza, chiarezza e testabilità:

- Informazioni generali: identificativi, descrizione, sviluppatore, versione, data, contatto, licenza, requisiti hardware
- Progettazione: presupposti, decisioni tecniche, algoritmo ML
- Utilizzo: scopo previsto, utilizzi primari/secondari, utenti, approccio autoapprendente, bias, etica, sicurezza fisica, trasparenza, soglie, piattaforma, deriva dei dati, deriva concettuale
- Dataset: feature, fonte, raccolta, disponibilità, pre-elaborazione, utilizzo, contenuto, etichette, dimensioni, privacy, sicurezza, bias/equità, limitazioni
- Testing: dettagli del dataset di test, indipendenza del testing, risultati dei test, attività di test (ad esempio funzionali, avversarie)
- Aspetti funzionali: misure, dataset di validazione, soglie, prestazioni effettive.
- Aspetti non funzionali: scalabilità, affidabilità, disponibilità, efficienza delle prestazioni (ad esempio latenza, utilizzo delle risorse), manutenibilità, robustezza dell'AI.
- Aspetti operativi: piano di deployment, ambiente di deployment, risorse computazionali, metriche/avvisi di monitoraggio, strategia di riaddestramento, piano di aggiornamento/rollback del modello, piano di ritiro, sicurezza (rischi avversari), metodi di spiegabilità

La review della documentazione rispetto a tali checklist è un'attività di test fondamentale e mira a:

- Identificare informazioni mancanti, inesattezze e incoerenze.
- Migliorare chiarezza e leggibilità.
- Migliorare la manutenibilità identificando dove la documentazione debba essere migliorata.
- Fornire informazioni sufficienti per le attività di test e di deployment.
- Verificare che tutti i requisiti normativi rilevanti siano stati soddisfatti.

6.1.3 Testing dell'Efficienza delle Prestazioni Funzionali di ML dei Sistemi Probabilistici di Machine Learning

Il testing dell'efficienza delle prestazioni funzionali di ML valuta in quale misura un modello ML svolga le funzioni previste, misurando metriche quali accuracy, recall, precision e F1-score (vedi 3.3) e confrontando i risultati con i criteri di accettazione definiti.

Per gli MLS probabilistici questo testing supera il semplice stato superato/fallito e misura statisticamente le prestazioni di un modello rispetto ai relativi criteri di accettazione [STATS]. Tale approccio è necessario

per affrontare il non determinismo intrinseco degli MLS, valutandone il comportamento su un dataset ampio e rappresentativo (vedi 4.1.2).

Una preconditione per questo testing è disporre di criteri di accettazione definiti in termini statistici. Anziché un semplice valore target, un requisito può specificare una metrica delle prestazioni, un margine di errore (MoE, Margin of Error) e un livello di confidenza (CL, Confidence Level). Un requisito di questo tipo può essere utilizzato per determinare il numero minimo di test necessari. All'aumentare del numero di test sono possibili due opzioni:

- Fissare il CL: il MoE diminuirà. Ciò significa che il risultato del test diventa più preciso.
- Fissare il MoE: il CL aumenterà. Ciò significa che il risultato del test diventa più certo.

Ad esempio, un criterio potrebbe essere “accuracy del 98% con un MoE di $\pm 4\%$ a un CL del 95%”. Per confermare che l'accuracy misurata abbia un MoE massimo di $\pm 4\%$ al CL del 95% è necessario un campione di 601 test case. Il valore si calcola utilizzando la formula della dimensione del campione per la stima di una proporzione di popolazione. Questa dimensione del campione si basa sull'ipotesi prudenziale che l'accuracy effettiva possa assumere qualsiasi valore tra 0% e 100%, il che produce la massima incertezza possibile ma garantisce comunque il MoE di $\pm 4\%$ in tutte le condizioni. Per raggiungere l'accuracy target del 98% devono essere superati almeno 589 dei 601 test case. Se, dopo l'esecuzione di tutti i 601 test case, l'accuracy osservata è pari al 98%, il MoE misurato al CL del 95% sarà inferiore a $\pm 4\%$ (circa $\pm 1,1\%$), poiché la varianza è più bassa in presenza di valori elevati di accuracy. Il MoE si calcola utilizzando la formula del margine di errore per una proporzione campionaria. Ciò significa che, durante il testing, non è sempre necessario eseguire tutti i 601 test case. Il testing sequenziale fornisce un framework statistico formale per questa interruzione anticipata. Anziché eseguire sempre l'intero campione di dimensione fissa, questi approcci analizzano i risultati man mano che si accumulano e interrompono anticipatamente il testing quando le evidenze sono sufficienti a supportare il target di accuracy (o a respingerlo). Se l'accuracy osservata rimane costantemente elevata (ad esempio non inferiore al 98%), l'incertezza statistica diminuisce con l'aumentare dei test eseguiti. In tale situazione il MoE richiesto di $\pm 4\%$ al CL del 95% viene raggiunto dopo circa 170 test case, consentendo di concludere prima il testing.

NOTA: le formule e i calcoli numerici di questo esempio (relativi alla dimensione del campione, al margine di errore e agli intervalli di confidenza) sono forniti esclusivamente a scopo illustrativo e di comprensione; all'esame ai candidati non sarà richiesto di ricavare o calcolare tali valori mediante formule statistiche.

Per i sistemi critici per la sicurezza fisica può essere utilizzato un requisito di affidabilità più rigoroso, quale “affidabilità del 99% con una confidenza del 95%”, che richiederebbe 299 test case, tutti da superare per soddisfare il criterio.

Per validare tali criteri, il testing richiede un ampio dataset di test completamente indipendente dai dataset di addestramento e di validazione. Questo dataset di test deve essere un campione rappresentativo del dominio operativo degli input (vedi 5.1.4), per verificare che la valutazione rifletta le condizioni del mondo reale. I test case vengono quindi eseguiti utilizzando il modello ML all'interno di un framework di sviluppo ML in grado di eseguire analisi statistiche.

Infine, i risultati aggregati vengono interpretati e riportati con una confidenza statistica, e non come un semplice rapporto tra superati e falliti. Un test report finale potrebbe indicare, ad esempio: “Il modello ha conseguito un'accuracy del 94% $\pm 4\%$ al CL del 95%”. Ciò consente agli stakeholder di comprendere l'intervallo delle prestazioni operative attese e di decidere in modo informato se l'efficienza delle prestazioni funzionali di ML del modello sia accettabile per il deployment.

6.1.4 Adversarial Testing dei Sistemi di Machine Learning

L'adversarial testing comporta la fornitura deliberata al modello di perturbazioni dei dati di input, spesso impercettibili per le persone. Questi input sono progettati per indurre il modello a produrre previsioni errate e, se hanno successo, sono chiamati adversarial example. Pertanto, gli input di test per l'adversarial testing, e quindi i potenziali adversarial example, consistono spesso in versioni lievemente modificate di input legittimi che inducono il modello a classificarli erroneamente.

L'identificazione delle vulnerabilità mediante adversarial testing consente agli sviluppatori di introdurre misure di protezione e di rendere il modello più robusto nei confronti degli adversarial example. Questi possono essere adversarial example accidentali che il modello incontra oppure attacchi avversari che comportano l'utilizzo malevolo di adversarial example. La generazione di input di test efficaci per l'adversarial testing è tecnicamente complessa e mantenersi aggiornati sull'evoluzione delle tecniche di attacco costituisce una sfida continua.

L'adversarial testing può essere eseguito mediante testing black-box, focalizzandosi sul comportamento di input e output del modello senza richiedere la conoscenza del suo funzionamento interno. A tale scopo è possibile creare un modello equivalente del quale siano noti gli elementi interni e, grazie a tale conoscenza, creare input di test avversari. Quindi, in base all'ipotesi che modelli equivalenti condividano gli stessi confini di classificazione (ossia la trasferibilità), gli adversarial test possono essere applicati al modello originale. In alternativa è possibile adottare un approccio brute force con un numero molto elevato di test, nella speranza che alcuni test casuali coincidano con un adversarial example.

L'adversarial testing può essere eseguito anche mediante testing white-box. La conoscenza degli elementi interni di un modello ML (architettura, parametri, processo di addestramento) rende normalmente più semplice creare adversarial example.

L'adversarial testing può essere eseguito manualmente, creando specifici adversarial example, oppure mediante algoritmi automatizzati che generano un numero elevato di variazioni per individuare input avversari efficaci.

6.1.5 Testing Metamorfico

Il testing metamorfico (MT) è una tecnica di test nella quale nuovi test case (di follow-up) vengono ricavati da un test case sorgente precedentemente superato. Uno o più test case di follow-up vengono generati modificando ("metamorfizzando") il test case sorgente mediante una relazione metamorfica (MR). La MR si basa su una proprietà di una funzione richiesta dell'oggetto di test e descrive come una modifica agli input di un test case si rifletta nei risultati attesi dello stesso test case.

Il MT può essere utilizzato per la maggior parte degli oggetti di test e può essere applicato sia al testing funzionale sia al testing non-funzionale. I tester verificano obiettivi del test quali la coerenza (gli output sono allineati fra input correlati), la monotonicità (gli output variano direzionalmente con l'input) e l'invarianza (gli output rimangono stabili in presenza di perturbazioni). È particolarmente utile quando la generazione dei risultati attesi è problematica perché non è disponibile un oracolo del test economicamente sostenibile. Ciò avviene per alcuni MLS che utilizzano big data oppure per sistemi nei quali i tester non hanno chiaro il modo in cui il modello ML ricava le proprie previsioni, il che accade spesso. Nell'ambito dell'AI e del ML, il testing metamorfico è stato utilizzato per testare il riconoscimento delle immagini, i motori di ricerca, l'ottimizzazione dei percorsi e il riconoscimento vocale.

Il MT viene normalmente preferito al testing tradizionale basato su oracoli quando:

- non esistono output attesi affidabili, a causa dell'opacità del modello o della scala dei dati;
- il sistema è una scatola nera; oppure

- le proprietà relazionali (anziché i valori assoluti) sono sufficienti a fornire confidenza.

Il MT si basa spesso su un test case sorgente superato e può essere utile anche quando non è possibile generare un risultato atteso: ad esempio quando il programma implementa una funzione troppo complessa perché un tester umano possa replicarla e utilizzarla come oracolo del test, come accade con alcuni MLS complessi. In questa situazione il MT può essere utilizzato per generare test case che, una volta eseguiti, producono un insieme di output nei quali si verifica la validità delle relazioni reciproche (anziché dei valori effettivi). Con questa forma di MT, se le relazioni tra gli output del test risultano valide, la confidenza nel programma aumenta. Ad esempio, un MLS di valutazione del rischio che prevede l'età del decesso, nel quale l'aumento del numero di sigarette fumate dovrebbe ridurre la previsione (monotonicità).

I tester ricavano le MR dalla conoscenza del dominio, dai requisiti o dalle proprietà del dominio (ad esempio le leggi della fisica). Le MR possono essere validate mediante review di esperti, eseguendole su modelli di riferimento e verificando la copertura degli edge case.

MR errate (ad esempio perché trascurano interazioni complesse tra le variabili) o insiemi incompleti di MR possono produrre una falsa confidenza. Il MT rileva difetti relazionali, ma non tutti gli errori assoluti: dovrebbe pertanto essere utilizzato in combinazione con altre tecniche di test.

6.1.6 Esercizio Pratico: Applicare il Testing Metamorfico

In questo esercizio gli studenti acquisiranno esperienza pratica di quanto segue:

- Ricavare diverse MR per un determinato MLS. Queste MR dovrebbero comprenderne alcune nelle quali i risultati attesi dei test case sorgente e di follow-up siano uguali e altre nelle quali siano differenti.
- Generare test case sorgente per l'MLS. Non è necessario che siano garantiti come superati, ma occorre ricordare agli studenti le limitazioni del MT quando non sono disponibili test case sorgente superati.
- Utilizzare le MR ricavate e i test case sorgente generati per ricavare test case di follow-up.
- Eseguire i test case di follow-up.

6.1.7 Drift Testing

Il drift testing può essere utilizzato per identificare due forme di deriva negli MLS operativi:

- Deriva dei dati – si verifica quando le proprietà statistiche dei dati operativi di input cambiano nel tempo. Ad esempio, i dati di input diventano significativamente differenti dai dati sui quali il modello è stato addestrato, a causa di fattori quali variazioni del comportamento degli utenti o stagionalità. Un esempio è un filtro antispam che incontra nuovi tipi di attacchi di phishing inesistenti durante il suo addestramento.
- Deriva concettuale – si verifica quando la relazione tra i dati di input e l'output corretto cambia nel tempo. I pattern o le regole appresi originariamente dal modello non riflettono quindi più la realtà attuale. Ad esempio, a causa di nuove normative finanziarie, un tipo di transazione precedentemente considerato “a basso rischio” potrebbe ora essere classificato come “ad alto rischio”. Il significato dei dati è cambiato, rendendo obsoleti i confini decisionali appresi dal modello e determinando un calo dell'accuracy delle sue previsioni.

Il drift testing dinamico si basa sulla disponibilità di feedback degli utenti, che fornisce la ground truth corrente. Tale ground truth viene confrontata con l'output del modello; la differenza tra i due viene determinata e confrontata con un valore soglia. Il feedback degli utenti può essere diretto o indiretto. Per un sistema di raccomandazione di film, un esempio di feedback diretto è la valutazione della raccomandazione da parte dell'utente. Un esempio di feedback indiretto per lo stesso sistema si ricava dai dati relativi ai film guardati dall'utente.

Il drift testing statico non dipende dalla ground truth corrente, ma confronta le proprietà statistiche delle distribuzioni dei dati di input e degli output previsti mediante un test quale quello di Kolmogorov-Smirnov [STATS]. Una differenza significativa in una delle due distribuzioni indica che si è verificata una deriva.

6.1.8 Testing per Overfitting e Underfitting

L'overfitting e l'underfitting sono due dei tre possibili esiti riscontrabili nei modelli ML; il terzo è un modello con un fitting corretto. Il testing per overfitting e underfitting dovrebbe essere eseguito durante l'addestramento, la valutazione e la messa a punto.

L'overfitting si verifica quando un modello apprende i dati di addestramento troppo bene, al punto da acquisire il rumore presente nei dati anziché il pattern sottostante. Ne consegue una scarsa generalizzazione su dati nuovi e mai osservati.

Per eseguire il testing per l'overfitting, l'efficienza delle prestazioni funzionali di ML del modello viene valutata su un dataset di test separato, non utilizzato durante l'addestramento. Tale dataset di test dovrebbe includere alcuni esempi meno comuni, difficilmente utilizzati durante l'addestramento. Il modello potrebbe presentare overfitting se ottiene prestazioni significativamente peggiori sul dataset di test rispetto al dataset di validazione.

L'underfitting si verifica quando un modello è troppo semplice per acquisire la struttura sottostante dei dati oppure quando i dati di addestramento non contengono feature che riflettano una relazione importante tra input e output; ne consegue una scarsa efficienza delle prestazioni funzionali di ML sia sul dataset di addestramento sia su quello di validazione.

Durante il testing, l'underfitting può essere rilevato valutando metriche di efficienza delle prestazioni funzionali di ML del modello quali accuracy, precision, recall o F1-score. Se tali metriche restano costantemente basse sia sull'insieme di addestramento sia su quello di validazione, ciò indica che il modello presenta underfitting.

Anche l'ispezione visiva delle curve di apprendimento del modello può contribuire a rilevare l'underfitting. Se gli errori di addestramento e di validazione restano elevati e relativamente vicini, senza miglioramenti significativi con il progredire dell'addestramento, ciò indica underfitting.

In sintesi, il rilevamento di overfitting e underfitting durante il testing comporta la valutazione dell'efficienza delle prestazioni funzionali di ML del modello sui dati di validazione, l'analisi delle metriche di efficienza delle prestazioni funzionali di ML e l'esame delle curve di apprendimento.

6.1.9 Testing A/B

Il testing A/B è un approccio nel quale vengono confrontate le risposte di due varianti del programma (A e B) agli stessi input, allo scopo di determinare quale delle due sia migliore. È un approccio di testing statistico che richiede normalmente il confronto dei risultati di più esecuzioni di test per determinare le differenze tra i programmi.

Un semplice esempio di questo approccio consiste nell'inviare via e-mail due offerte promozionali a una lista di marketing suddivisa in due gruppi. Metà della lista riceve l'offerta A e metà l'offerta B; il successo di ciascuna offerta aiuta a decidere quale utilizzare in futuro. Molte aziende di e-commerce e basate sul web utilizzano il testing A/B in produzione, indirizzando consumatori differenti verso funzionalità differenti, per contribuire a identificarne le preferenze.

Il testing A/B è uno degli approcci per affrontare il problema dell'oracolo del test e utilizza normalmente il sistema esistente come oracolo del test parziale. Il testing A/B non genera test case e non fornisce indicazioni su come progettare i test, sebbene nei test vengano spesso incorporati input operativi.

Il testing A/B può essere utilizzato per testare gli aggiornamenti di un sistema basato sull'AI, purché esistano criteri di accettazione concordati, quali le metriche di efficienza delle prestazioni funzionali di ML descritte in 3.3. Ogni volta che il sistema viene aggiornato, il testing A/B viene utilizzato per determinare che la variante aggiornata ottenga prestazioni pari o superiori a quelle della variante precedente.

Un approccio del test di questo tipo può essere utilizzato per un classificatore semplice, ma anche per testare sistemi molto più complessi. Ad esempio, un aggiornamento volto a migliorare l'efficacia del sistema di instradamento dei trasporti di una smart city può essere testato mediante testing A/B, ad esempio confrontando i tempi medi di percorrenza di due varianti del sistema in settimane consecutive.

Il testing A/B può essere utilizzato anche per testare i sistemi autoapprendenti. Quando il sistema apporta una modifica, vengono eseguiti test automatizzati e le caratteristiche risultanti del sistema vengono confrontate con quelle precedenti alla modifica. Se il sistema è migliorato, la modifica viene accettata; in caso contrario il sistema ritorna allo stato precedente.

Le tecniche statistiche più diffuse per il testing A/B sono il t-test, lo z-test, il test chi quadrato e il test U di Mann-Whitney [STATS].

6.1.10 Testing Back-to-Back

Il testing back-to-back offre una soluzione pratica al problema dell'oracolo del test.

Il testing back-to-back comporta l'utilizzo di una versione alternativa del sistema come punto di riferimento (uno pseudo-oracolo) e il confronto dei suoi output con quelli del sistema sotto test quando entrambi ricevono gli stessi input. Questo pseudo-oracolo può essere un sistema esistente oppure un sistema sviluppato specificamente per il testing, il che comporta però un costo.

Idealmente, lo pseudo-oracolo e il system under test non dovrebbero condividere componenti software comuni. In caso contrario entrambi i sistemi potrebbero contenere lo stesso difetto e produrre output corrispondenti anche quando entrambi sono errati. Ciò può essere particolarmente problematico, considerato l'ampio utilizzo di componenti AI riusabili e open source nello sviluppo degli MLS. Per questo motivo lo pseudo-oracolo viene spesso sviluppato da un team differente e idealmente indipendente, eventualmente utilizzando framework di sviluppo ML, algoritmi o impostazioni del modello differenti. Talvolta il software convenzionale può fungere da pseudo-oracolo, se risolve lo stesso problema.

Quando si esegue il testing back-to-back funzionale, lo pseudo-oracolo deve corrispondere soltanto al comportamento funzionale. Non deve soddisfare gli stessi requisiti non funzionali del sistema sotto test e può quindi essere meno costoso da realizzare.

Questo approccio del test richiede soltanto la generazione degli input di test, non dei risultati attesi, poiché lo pseudo-oracolo fornisce il punto di confronto. Tali input possono provenire da test case esistenti, quali le test suite di regression testing, oppure essere generati automaticamente dai dati di

addestramento, consentendo di eseguire un numero elevato di test quando è supportata l'esecuzione automatizzata dei test.

Il testing back-to-back offre un valore significativo durante la migrazione di un MLS verso un nuovo ambiente, ad esempio dallo sviluppo alla produzione, e nel confronto dei risultati dei test tra ambienti. Questo approccio del test può inoltre rivelare difetti sottili nel comportamento del modello che potrebbero non emergere con altri approcci del test, soprattutto confrontando le risposte su un'ampia gamma di edge case o di input insoliti.

Una differenza significativa tra il testing A/B (vedi 6.1.9) e il testing back-to-back consiste nell'utilizzo del testing A/B per confrontare due varianti dello stesso MLS mediante metriche di efficienza delle prestazioni funzionali di ML e tecniche statistiche, rispetto all'utilizzo del testing back-to-back per rilevare difetti.

7 Testing dello Sviluppo del Machine Learning – 30 minuti

Parole Chiave

Testing dello sviluppo ML, efficienza delle prestazioni funzionali di ML, shadow testing

Parole Chiave Specifiche dell'AI

Nessuna

Obiettivi di Apprendimento per il Capitolo 7:

7.1 Testing dello Sviluppo del Machine Learning

- AI-7.1.1 (K2) Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo allo sviluppo ML
- AI-7.1.2 (K2) Spiegare le diverse forme di testing del deployment dei sistemi ML

7.1 Testing dello Sviluppo del Machine Learning

Questo capitolo si focalizza specificamente sui rischi introdotti dagli strumenti di sviluppo ML, dalle scelte di configurazione e dai meccanismi di deployment, anziché sul modello ML stesso. Tratta approcci del test e tipi di test quali il testing delle API, il testing dell'efficienza delle prestazioni funzionali di ML, il testing A/B, il testing back-to-back e le review, per verificare la robustezza e l'efficienza dell'AI.

7.1.1 Rischi Relativi allo Sviluppo del Machine Learning e Relative Mitigazioni

La tabella seguente elenca esempi di rischi relativi allo sviluppo ML e il corrispondente testing che potrebbe essere utilizzato per la mitigazione del rischio:

Rischio Potenziale	Possibile Mitigazione del Rischio
Utilizzo errato o non previsto delle API di librerie o framework (ad esempio TensorFlow, PyTorch)	Testing delle API – vedi 7.1.2
Selezione non ottimale del framework	Review dell'idoneità del framework
Problemi nell'algoritmo, nel modello o nel framework di sviluppo che introducono iniquità sistemica	Testing per bias – vedi 5.1.2
Installazione o build difettosa del framework	Smoke testing
Implementazione difettosa della valutazione da parte del framework	Review del codice di valutazione del framework Controllo incrociato dei risultati della valutazione del framework (ad esempio rispetto a benchmark manuali)
Efficienza delle prestazioni inadeguata (ad esempio framework lento a rispondere)	Performance testing
Usabilità inadeguata del framework	Testing di usabilità
Difetto in una libreria utilizzata dal framework (ad esempio un difetto in PyTorch) Implementazione difettosa dell'algoritmo	Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3 Testing back-to-back – vedi 6.1.10

Vulnerabilità di sicurezza nel framework	Testing di sicurezza
Documentazione utente del framework inadeguata	Review della documentazione del framework
Selezione non ottimale dell'algoritmo	Review dell'idoneità dell'algoritmo Testing A/B – vedi 6.1.9
Selezione non ottimale degli iperparametri (ad esempio struttura della rete, learning rate)	Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3 Testing A/B – vedi 6.1.9
Allocazione difettosa dei dati ai dataset di addestramento, di validazione e di test	Review dell'allocazione dei dati
Selezione inadeguata dell'approccio di valutazione (ad esempio cross-validation k-fold)	Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3
Interpretazione errata dei risultati dei test a causa della natura stocastica del processo di apprendimento	Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3
Difetto di deployment (ad esempio derivante dalla generazione di una versione modificata per una piattaforma target)	Smoke testing Testing dell'efficienza delle prestazioni funzionali di ML – vedi 6.1.3 Testing A/B – vedi 6.1.9
Il modello sottoposto a deployment non è compatibile con l'ambiente operativo	Smoke testing Testing del deployment dell'MLS – vedi 7.1.2
Il modello sottoposto a deployment non costituisce un miglioramento rispetto al modello attuale	Shadow testing – vedi 7.1.2

7.1.2 Testing del Deployment dei Sistemi di Machine Learning

Il deployment degli MLS comprende diverse attività di test fondamentali, focalizzate sulla verifica che il sistema basato sull'AI funzioni correttamente e in modo affidabile nel proprio ambiente target (ad

esempio cloud, dispositivo edge, dispositivo mobile). Ciascuno dei tipi di test seguenti affronta rischi specifici durante il deployment:

- Testing di installabilità – verifica che l'MLS possa essere installato, configurato e successivamente disinstallato con successo in tutti gli ambienti supportati. Comprende il testing delle dipendenze di sistema (ad esempio i driver delle GPU), della compatibilità con i framework e della corretta esecuzione degli script di installazione.
- Testing del rollback – verifica la capacità del sistema di ritornare con successo a uno stato precedente stabile e operativo dopo un deployment degradato o fallito. Può riguardare soltanto il modello oppure l'intero sistema (ad esempio comprendendo la pipeline di dati). Si noti che il testing del rollback deve essere eseguito prima del deployment, per confermare che il rollback sia pronto.
- Canary testing – valida i nuovi deployment rilasciando un modello aggiornato a un piccolo sottoinsieme del traffico di produzione (ad esempio il 5% degli utenti). Metriche in tempo reale, quali latenza, accuracy e tassi di errore, vengono monitorate per rilevare regressioni prima del rollout completo.
- Shadow testing – esegue un nuovo modello in parallelo con il modello di produzione corrente e in tempo reale, instradando le stesse richieste verso entrambi i sistemi senza influire sulle risposte effettive. Consente di confrontare il modello nuovo e quello precedente utilizzando dati reali in un ambiente controllato e a basso rischio, e può individuare difetti quali regressioni delle prestazioni e deriva dei dati prima del deployment completo.
- Testing di conversione del modello – verifica che un modello ML mantenga un'accuracy predittiva accettabile, un comportamento coerente e un'efficienza operativa adeguata (ad esempio velocità di inferenza, utilizzo della memoria) dopo la conversione dal formato originale di addestramento a un formato di deployment adatto all'ambiente di produzione target.
- Testing cross-device – verifica che l'MLS funzioni correttamente nell'intera gamma prevista di target di deployment, dai dispositivi mobili e dai dispositivi edge ai server cloud.
- Testing delle API – verifica che l'MLS esponga interfacce ben definite e conformi agli standard. Valida la corretta gestione di input e output, dei messaggi di errore e dei workflow di integrazione con sistemi esterni quali feed di dati, client e pipeline.

8 Elenco delle Abbreviazioni

Abbreviazione	Descrizione
AI	artificial intelligence (intelligenza artificiale)
AlaaS	AI as a service
API	application programming interface (Application Program Interface)
CL	confidence level (livello di confidenza)
CNN	convolutional neural network (rete neurale convoluzionale)
CPU	central processing unit (unità centrale di elaborazione)
DL	deep learning
DNN	deep neural network (rete neurale profonda)
EDA	exploratory data analysis (analisi esplorativa dei dati)
FN	false negative (falso negativo)
FP	false positive (falso positivo)
GAN	generative adversarial network (rete generativa avversaria)
GenAI	generative AI (AI generativa)
GPU	graphics processing unit (unità di elaborazione grafica)
HO	hands-on objective (obiettivo pratico)
IAA	inter-annotator agreement (accordo tra annotatori)
kMNC	k-multisection neuron coverage (copertura k-multisezione dei neuroni)
LIME	Local interpretable model-agnostic explanations
LLM	large language model(s)
LO	learning objective (obiettivo di apprendimento)
ML	machine learning
MLS	machine learning system(s) (sistema/sistemi di machine learning)
MoE	margin of error (margine di errore)

NBC	neuron boundary coverage (copertura dei confini dei neuroni)
NLP	natural language processing (elaborazione del linguaggio naturale)
RAG	retrieval-augmented generation
RNN	recurrent neural network (rete neurale ricorrente)
RT	red teaming
SVM	support vector machine
TN	true negative (vero negativo)
TP	true positive (vero positivo)

9 Termini Specifici dell'AI

Termine	Definizione
accordo tra annotatori	Il grado di consenso o di somiglianza tra le annotazioni eseguite da annotatori differenti sugli stessi dati. (ISO/IEC TS 12791)
accuracy	La metrica di efficienza delle prestazioni funzionali di ML utilizzata per valutare un classificatore, che misura la proporzione di previsioni corrette. (Adattato da ISO/IEC TR 29119-11)
acquisizione dei dati	L'attività di acquisire i dati rilevanti per il problema di business da risolvere mediante un modello ML.
agente intelligente	Un programma autonomo che orienta la propria attività al conseguimento di obiettivi mediante osservazioni e azioni.
AI as a Service	Un modello di concessione in licenza e di erogazione del software nel quale l'AI e i servizi di sviluppo dell'AI sono ospitati centralmente.
AI generale	Un tipo di AI in grado di eguagliare le capacità cognitive umane nella maggior parte delle attività intellettuali.
AI generativa	Un tipo di AI che crea nuovi contenuti apprendendo pattern dai dati esistenti.
AI ristretta	AI focalizzata su una singola attività ben definita, per affrontare un problema specifico. Sinonimo: AI debole (ISO/IEC TR 29119-11)
AI spiegabile	Il campo di studio relativo alla comprensione dei fattori che influenzano gli output dei sistemi AI.
albero decisionale	Un modello ML ad albero i cui nodi rappresentano decisioni e i cui rami rappresentano possibili esiti.
algoritmo di clustering	Un tipo di algoritmo ML utilizzato per raggruppare oggetti simili in cluster.
algoritmo di ricerca	Un algoritmo che visita sistematicamente un sottoinsieme di tutti gli stati o di tutte le strutture possibili, finché non viene raggiunto lo stato o la struttura obiettivo. (Adattato da ISO/IEC TR 29119-11)

algoritmo ML	Un algoritmo utilizzato per creare un modello ML a partire da un dataset di addestramento.
analisi degli impatti disparati	Una tecnica per rilevare il bias confrontando le decisioni relative a scenari originali con quelle relative alle loro versioni controfattuali, nelle quali gli attributi sensibili sono scambiati.
analisi del punteggio di confidenza del modello	Una tecnica che identifica i punti dati con punteggi di confidenza bassi durante l'addestramento, che sono indicatori di punti dati etichettati in modo errato.
analisi del sentiment	L'utilizzo dell'elaborazione del linguaggio naturale e del machine learning per identificare e classificare il tono emotivo espresso in un testo.
analisi della loss del modello	Una tecnica che identifica i punti dati con valori di loss elevati durante l'addestramento, che sono indicatori di punti dati etichettati in modo errato.
analisi esplorativa dei dati	Un processo interattivo, visivo e guidato da ipotesi, volto a sintetizzare, esplorare e comprendere le principali caratteristiche e i principali pattern dei dati.
annotazione	L'attività di identificare oggetti nelle immagini mediante bounding box, per fornire dati etichettati destinati alla classificazione.
annotazione multipla	Un approccio nel quale vengono confrontati punti dati etichettati provenienti da più annotatori.
apprendimento non supervisionato	Un approccio all'addestramento di un modello ML mediante l'utilizzo di un dataset non etichettato.
apprendimento per rinforzo	Un approccio nel quale un modello ML, detto agente, apprende utilizzando un ciclo di feedback per tentativi e ricompense con il proprio ambiente, allo scopo di conseguire obiettivi specifici.
apprendimento supervisionato	Un approccio all'addestramento di un modello ML mediante l'utilizzo di un dataset etichettato.
architettura di von Neumann	Un'architettura di elaboratore costituita da cinque componenti principali: memoria, unità centrale di elaborazione, unità di controllo, input e output.
associazione	Una tecnica di apprendimento ML non supervisionato che identifica relazioni e dipendenze tra i campioni.
attacco avversario	L'utilizzo deliberato di adversarial example per far fallire un modello ML.
attacco di cross-prompt injection	Un attacco nel quale istruzioni malevole contenute in un prompt o in un segmento di contesto alterano il comportamento di un

	modello AI in un prompt, un turno o un segmento di contesto differente.
attributi sensibili	Caratteristiche che definiscono gruppi e individui protetti sul piano legale o etico, che devono essere controllate per prevenire discriminazioni.
aumento dei dati	L'attività di creare nuovi punti dati a partire da un dataset esistente.
bias	La differenza sistematica di trattamento di determinati oggetti, persone o gruppi rispetto ad altri. (Adattato da ISO/IEC TR 24027)
bias algoritmico	Un tipo di bias causato dall'algoritmo ML.
bias dei dati	Un errore sistematico causato da dati di addestramento inaccurati, incompleti o non rappresentativi, che determina risultati iniqui nei modelli ML.
big data	Dataset estesi le cui caratteristiche in termini di volume, varietà, velocità e/o variabilità richiedono tecnologie e tecniche specializzate per essere elaborati.
campionamento stratificato	Una tecnica per confermare se un campione rappresenta proporzionalmente le diverse sottopopolazioni all'interno della popolazione complessiva.
chatbot	Un'applicazione utilizzata per condurre una conversazione tramite testo o sintesi vocale.
classificatore	Un modello ML utilizzato per la classificazione. Sinonimo: modello di classificazione
classificazione	Una funzione ML che prevede la classe di output per un determinato input. (Adattato da ISO/IEC TR 29119-11)
clustering	Una funzione ML che raggruppa punti dati simili.
componente AI	Un componente che fornisce funzionalità AI.
dataset	Un insieme di dati utilizzati per l'addestramento, la valutazione, il testing e la previsione nel ML.
dataset di addestramento	Un dataset utilizzato per addestrare un modello ML.
dataset di validazione	Un dataset utilizzato per valutare un modello ML addestrato allo scopo di metterlo a punto.

deep learning	ML che utilizza reti neurali profonde per apprendere automaticamente feature e rappresentazioni complesse da dataset di grandi dimensioni.
deepfake	Contenuti multimediali sintetici, quali video, audio o immagini, creati o modificati mediante AI per imitare o impersonare in modo convincente persone o eventi reali.
deriva concettuale	Una variazione nel tempo delle prestazioni funzionali percepite di un modello ML, causata da cambiamenti nelle aspettative degli utenti, nei loro comportamenti e nell'ambiente operativo.
deriva dei dati	Una variazione nel tempo della distribuzione dei dati di input, che può incidere negativamente sulle prestazioni funzionali di un modello ML operativo.
deterministico	Che produce lo stesso insieme di output e lo stesso stato finale a partire da un determinato insieme di input e da un determinato stato iniziale.
edge AI	Il deployment di modelli AI su dispositivi edge locali, che consente l'elaborazione in tempo reale in prossimità della fonte dei dati senza dipendere da sistemi basati sul cloud.
edge computing	La parte di un'architettura distribuita nella quale l'elaborazione delle informazioni viene eseguita in prossimità del luogo in cui tali informazioni vengono utilizzate.
elaborazione del linguaggio naturale	Un settore dell'informatica che fornisce la capacità di leggere, comprendere e ricavare significato dai linguaggi naturali.
epoca	Un'iterazione dell'addestramento ML sull'intero dataset di addestramento.
F1-score	Una metrica di efficienza delle prestazioni funzionali di ML utilizzata per valutare un classificatore, che fornisce un equilibrio tra recall e precision.
falso negativo	Una previsione di un modello ML nella quale il modello prevede erroneamente la classe negativa.
falso positivo	Una previsione di un modello ML nella quale il modello prevede erroneamente la classe positiva.
feature	Un singolo attributo misurabile dei dati di input, utilizzato per l'addestramento da parte di un algoritmo ML e per la previsione da parte di un modello ML.
feature engineering	L'attività nella quale gli attributi dei dati grezzi che meglio rappresentano le relazioni sottostanti che dovrebbero comparire

	nel modello ML vengono identificati per essere utilizzati nei dati di addestramento. (ISO/IEC TR 29119-11)
foundation model	Un modello ML su larga scala addestrato su dataset di grandi dimensioni mediante apprendimento auto-supervisionato, progettato come base versatile che può essere sottoposta a fine-tuning o adattata a un'ampia gamma di attività in domini differenti.
framework di sviluppo ML	Una piattaforma software che fornisce strumenti e librerie per costruire, addestrare e sottoporre a deployment modelli ML.
frontier AI	Un sistema basato sull'AI di uso generale che supera le capacità dei sistemi AI più avanzati oggi disponibili.
funzione di attivazione	La formula associata a un neurone di una rete neurale che determina l'output del neurone a partire dai suoi input.
funzione di ricompensa	Una funzione che definisce il successo dell'apprendimento per rinforzo.
funzione ML	Funzionalità implementata da un modello ML, quale la classificazione, la regressione ML o il clustering.
ground truth	Le informazioni fornite dall'osservazione e dalla misurazione diretta, delle quali è nota la realtà o la veridicità.
intelligenza artificiale	La capacità di un sistema ingegnerizzato di acquisire, elaborare, creare e applicare conoscenze e abilità. (ISO/IEC TR 29119-11)
iperparametro	Un parametro utilizzato per controllare l'addestramento di un modello ML oppure per impostarne la configurazione.
Large Language Model	Un sistema di AI generativa da testo a testo, addestrato su raccolte molto ampie di dati linguistici.
logica fuzzy	Un tipo di logica basato sul concetto di verità parziale, rappresentata da fattori di certezza compresi tra 0 e 1.
matrice di confusione	Una tecnica per sintetizzare l'efficienza delle prestazioni funzionali di ML di un algoritmo di classificazione.
messa a punto degli iperparametri	L'attività di determinare gli iperparametri ottimali sulla base di obiettivi specifici.
MLS	Un sistema che integra uno o più modelli ML.
modello AI	Un programma per elaboratore che implementa l'AI

modello bayesiano	Un modello statistico che utilizza la probabilità per rappresentare l'incertezza sia degli input sia degli output del modello.
modello multimodale	Un modello ML progettato per gestire più tipi di modalità di dati, quali testo, immagini, audio e video.
modello preaddestrato	Un modello ML già addestrato su un dataset ampio e di uso generale, che può essere riutilizzato o sottoposto a fine-tuning per attività specifiche.
neurone	Un nodo di una rete neurale, che riceve solitamente più valori di input e genera un valore di attivazione.
non determinismo	Una proprietà di un sistema o di un processo nella quale un esito non è determinato univocamente dalle sue condizioni iniziali.
overfitting	La generazione di un modello ML che corrisponde troppo strettamente al dataset di addestramento, con il risultato di un modello che ha difficoltà a generalizzare su dati nuovi. (Adattato da ISO/IEC TR 29119-11)
perceptron	Una rete neurale con un solo strato e un solo neurone
peso	Una variabile interna di una connessione tra neuroni di una rete neurale, che influisce sul modo in cui questa calcola i propri output e che cambia man mano che la rete neurale viene addestrata.
pipeline di dati	L'implementazione delle attività di preparazione dei dati per fornire i dati di input a supporto dell'addestramento da parte di un algoritmo ML o della previsione da parte di un modello ML.
pre-elaborazione dei dati	Le attività di pulizia dei dati, trasformazione dei dati, aumento dei dati e campionamento dei dati nel workflow ML.
precision	Una metrica di efficienza delle prestazioni funzionali di ML utilizzata per valutare un classificatore, che misura la proporzione di positivi previsti che erano corretti. (Adattato da ISO/IEC TR 29119-11)
preparazione dei dati	Le attività di acquisizione dei dati, pre-elaborazione dei dati e feature engineering nel workflow ML.
previsione dei difetti	Una tecnica per prevedere le aree dell'oggetto di test nelle quali si verificheranno difetti oppure la quantità di difetti presenti
probabilistico	Comportamento descritto in termini di probabilità, nel quale gli esiti sono incerti e sono descritti da verosimiglianze anziché da certezze.

problema dell'oracolo del test	La sfida di determinare se un test sia stato superato o fallito per un determinato insieme di input di test e per un determinato stato.
processore neuromorfico	Un circuito integrato progettato per imitare i neuroni biologici del cervello umano.
punto dati	Un insieme di una o più misure che costituiscono una singola osservazione utilizzata come parte di un dataset.
random forest	Tecnologia ML di tipo ensemble per la classificazione, la regressione ML e altre attività, che opera costruendo ed eseguendo molti alberi decisionali e restituendo poi la moda della classe oppure la media delle previsioni dei singoli alberi.
recall	Una metrica di efficienza delle prestazioni funzionali di ML utilizzata per valutare un classificatore, che misura la proporzione di positivi effettivi previsti correttamente. Sinonimo: sensitivity (Adattato da ISO/IEC TR 29119-11)
regressione lineare	Una tecnica statistica che modella la relazione tra variabili adattando un'equazione lineare ai dati osservati, quando la variabile target è numerica.
regressione ML	Un tipo di funzione ML che produce un valore di output numerico o continuo per un determinato input. (Adattato da ISO/IEC TR 29119-11)
rete neurale	Una rete di elementi di elaborazione primitivi connessi da collegamenti pesati con pesi regolabili, nella quale ciascun elemento produce un valore applicando una funzione non lineare ai propri valori di input e lo trasmette ad altri elementi oppure lo presenta come valore di output. Sinonimo: rete neurale artificiale (ISO/IEC 2382)
rete neurale convoluzionale	Un tipo di modello di deep learning progettato per elaborare dati a griglia, quali le immagini, che gli consente di riconoscere pattern e feature spaziali mediante operazioni a strati.
rete neurale profonda	Una rete neurale composta da più strati di neuroni. Sinonimo: perceptron multistrato

rete neurale ricorrente	Un tipo di modello di deep learning progettato per elaborare dati sequenziali, che gli consente di riconoscere pattern e dipendenze nel tempo.
Retrieval-Augmented Generation	Una tecnica ML nella quale un sistema GenAI migliora dinamicamente il proprio output recuperando informazioni esterne rilevanti che integrano la propria conoscenza interna.
reward hacking	L'attività eseguita da un agente intelligente per massimizzare la propria funzione di ricompensa a scapito del conseguimento dell'obiettivo originale. (Adattato da ISO/IEC TR 29119-11)
rumore	Una distorsione o un'alterazione dei dati.
singularità tecnologica	Un momento futuro nel quale i progressi tecnologici non saranno più controllabili dalle persone. (Adattato da ISO/IEC TR 29119-11)
sistema autoapprendente	Un sistema adattivo che modifica il proprio comportamento apprendendo per tentativi ed errori. (Adattato da ISO/IEC TR 29119-11)
sistema basato sull'AI	Un sistema che incorpora uno o più componenti AI.
sistema basato sull'AI adattivo	Un sistema basato sull'AI che adegua il proprio comportamento in risposta ai cambiamenti del proprio ambiente operativo.
sistema basato sull'AI bloccato	Un sistema basato sull'AI deterministico, con un modello fisso e immutabile, che non modifica il proprio comportamento dopo il deployment.
sistema esperto	Un sistema basato sull'AI per risolvere problemi in un particolare dominio o area applicativa, che trae inferenze da una base di conoscenza sviluppata a partire dall'esperienza umana.
super AI	Un tipo di AI che supera ampiamente le capacità umane.
support vector machine	Un algoritmo ML supervisionato che individua l'iperpiano ottimale tra i punti dati per attività di classificazione o di regressione ML.
tecnica bootstrap	Una tecnica di ricampionamento che estrae ripetutamente campioni con reimmissione da un dataset di addestramento, per stimare i criteri di efficienza delle prestazioni funzionali di ML di un modello ML.
temperature	Un'impostazione che controlla la casualità degli output della GenAI: valori più bassi generano risultati più prevedibili, valori più alti generano risultati più creativi.

testing delle feature	Un tipo di test volto a determinare se un dataset di addestramento di un modello AI contenga un insieme appropriato di feature.
transformer	Un'architettura di rete neurale che elabora dati sequenziali per cogliere dipendenze a lungo raggio, alla base di attività di NLP, computer vision e applicazioni multimodali.
underfitting	La generazione di un modello ML che non riflette la tendenza sottostante del dataset di addestramento, con il risultato di un modello che produce previsioni inaccurate. (adattato da ISO/IEC TR 29119-11)
unità di elaborazione grafica	Un circuito integrato specifico per l'applicazione, progettato per manipolare e modificare la memoria allo scopo di accelerare la creazione di immagini in un frame buffer destinato alla visualizzazione su un dispositivo di output.
valore anomalo	Un'osservazione che si colloca al di fuori del pattern complessivo della distribuzione dei dati.
valore di attivazione	L'output della funzione di attivazione di un neurone di una rete neurale.
vero negativo	Una previsione nella quale il modello prevede correttamente la classe negativa.
vero positivo	Una previsione nella quale il modello prevede correttamente la classe positiva.
workflow ML	L'insieme delle attività utilizzate per sviluppare, sottoporre a deployment e far funzionare un modello ML.

10 Riferimenti

10.1 Standard

- **ISO/IEC/IEEE 12207** (2017), Systems and software engineering — Software life cycle processes
- **ISO/IEC 2382** (2015), Information technology — Vocabulary
- **ISO/IEC 22989** (2022), Information technology — Artificial intelligence — Artificial intelligence concepts and terminology
- **ISO/IEC TR 24027** (2021) Information technology — Artificial intelligence (AI) — Bias in AI systems and AI aided decision making
- **ISO/IEC 25010** (2023), Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) - Product quality model
- **ISO/IEC 25059** (2023), Software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Quality model for AI systems
- **ISO 26262-6** (2018), Road vehicles — Functional safety — Part 6: Product development at the software level
- **ISO/IEC TR 29119-11** (2020), Software and systems engineering — Software testing — Part 11: Guidelines on the testing of AI-based systems
- **ISO/IEC TS 42119-2** (2025) Artificial intelligence — Testing of AI – Part 2: Overview of testing AI systems
- **Serie ISO/IEC 42119** (in fase di sviluppo) Artificial intelligence — Testing of AI

10.2 Documenti ISTQB®

- [CTFL] ISTQB® Certified Tester Foundation Level v4.0.1, https://istqb.org/wp-content/uploads/2024/11/ISTQB_CTFL_Syllabus_v4.0.1.pdf (consultato il 29/07/2025)
- [CT-GenAI] ISTQB® Certified Tester – Testing with Generative AI (CT-GenAI), <https://istqb.org/certifications/gen-ai/> (consultato il 19/12/2025)

10.3 Riferimenti al Glossario

Riferimento per la terminologia utilizzata in questo syllabus:

- Glossario ISTQB® <https://glossary.istqb.org/>

10.4 Libri, Articoli e Pagine Web

- [COV_REF] An Overview of Structural Coverage Metrics for Testing Neural Networks, Usman et al, Aug 2022, [arXiv:2208.03407](https://arxiv.org/abs/2208.03407) (consultato il 29/07/2025)
- [DATA_DOC] Gebru, T., Morgenstern, J., Vecchione, B., et al. (2021). Datasheets for Datasets. Communications of the ACM, 64(12), 86-92, <https://dl.acm.org/doi/pdf/10.1145/3502158> (consultato il 07/10/2025)
- [EU AI Act] EU Artificial Intelligence Act, July 2024, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L_202401689 (consultato il 30/07/2025)
- [MODEL_DOC] Mitchell, M., Wu, S., Varma, R., et al. (2019). Model Cards for Model Reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency, <https://dl.acm.org/doi/10.1145/3287560.3287596> (consultato il 07/10/2025)
- [OECD AI] OECD Recommendation of the Council on Artificial Intelligence, May 2024, Organisation for Economic Co-operation and Development (OECD), <https://legalinstruments.oecd.org/en/instruments/oecd-legal-0449> (consultato il 07/10/2025)
- [UN Gov AI] Governing AI for Humanity: Final Report, September 2024, United Nations, https://www.un.org/sites/un2.un.org/files/governing_ai_for_humanity_final_report_en.pdf (consultato il 29/07/2025)
- [STATS] An Introduction to Statistical Learning: With Applications in R (2nd ed.) by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, Springer.

11 Marchi Commerciali

ISTQB® è un marchio registrato di International Software Testing Qualifications Board

12 Appendice A – Obiettivi di Apprendimento/Livello Cognitivo di Conoscenza

Gli obiettivi di apprendimento specifici applicabili a questo syllabus sono riportati all'inizio di ciascun capitolo. Ogni argomento del syllabus sarà oggetto d'esame in base al relativo obiettivo di apprendimento.

Gli obiettivi di apprendimento iniziano con un verbo d'azione corrispondente al relativo livello cognitivo di conoscenza, come indicato di seguito.

Livello 1: Ricordare (K1)

Il candidato ricorderà, riconoscerà e richiederà alla memoria un termine o un concetto.

Verbi d'azione: Ricordare, riconoscere.

Esempi
Ricordare i concetti della piramide di test.
Riconoscere gli obiettivi tipici del testing.

Livello 2: Comprendere (K2)

Il candidato è in grado di selezionare i motivi o le spiegazioni relativi ad affermazioni sull'argomento e di riassumere, confrontare, classificare e fornire esempi del concetto di testing.

Verbi d'azione: Classificare, confrontare, differenziare, distinguere, spiegare, fornire esempi, interpretare, riassumere

Esempi	Note
Classificare gli strumenti di test in base al loro scopo e alle attività di test che supportano.	
Confrontare i diversi livelli di test.	Può essere utilizzato per cercare somiglianze, differenze o entrambe.
Differenziare il testing dal debugging.	Ricerca le differenze tra i concetti.
Distinguere tra rischi di progetto e rischi di prodotto.	Consente di classificare separatamente due o più concetti.
Spiegare l'impatto del contesto sul processo di test.	

Esempi	Note
Fornire esempi dei motivi per cui il testing è necessario.	
Dedurre la root cause dei difetti a partire da un determinato profilo di failure.	
Riassumere le attività del processo di review dei prodotti di lavoro.	

Livello 3: Applicare (K3)

Il candidato è in grado di eseguire una procedura quando deve affrontare un'attività nota, oppure di selezionare la procedura corretta e applicarla a un determinato contesto.

Verbi d'azione: Applicare, implementare, preparare, utilizzare

Esempi	Note
Applicare l'analisi dei valori limite per ricavare test case da determinati requisiti.	Dovrebbe fare riferimento a una procedura, una tecnica, un processo o simili.
Implementare metodi di raccolta delle metriche a supporto dei requisiti tecnici e di gestione.	
Preparare test di installabilità per applicazioni mobili.	
Utilizzare la tracciabilità per monitorare l'avanzamento del test rispetto alla completezza e alla coerenza con gli obiettivi del test, la strategia di test e il test plan.	Può essere utilizzato in un LO nel quale si richiede al candidato di saper utilizzare una tecnica o una procedura. È simile ad "applicare".

Livello 4: Analizzare (K4)

Il candidato è in grado di separare le informazioni relative a una procedura o a una tecnica nelle loro parti costitutive, per comprenderle meglio, e di distinguere tra fatti e inferenze. Un'applicazione tipica consiste nell'analizzare un documento, un software o una situazione di progetto e proporre azioni appropriate per risolvere un problema o svolgere un'attività.

Verbi d'azione: Analizzare, scomporre, delineare, prioritizzare, selezionare.

Esempi	Note
Analizzare una determinata situazione di progetto per stabilire quali tecniche di test black-box o basate sull'esperienza debbano essere applicate per conseguire obiettivi specifici.	Esaminabile soltanto in combinazione con un obiettivo misurabile dell'analisi. Dovrebbe assumere la forma "Analizzare xxxx per xxxx" (o simile).

Esempi	Note
Prioritizzare i test case di una determinata test suite ai fini dell'esecuzione, sulla base dei rischi di prodotto correlati.	
Selezionare i livelli di test e i tipi di test appropriati per verificare un determinato insieme di requisiti.	Necessario quando la selezione richiede un'analisi.

Riferimento

(Per i livelli cognitivi degli obiettivi di apprendimento)

Anderson, L. W. e Krathwohl, D. R. (a cura di) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Allyn & Bacon

13 Appendice B – Matrice di Tracciabilità dei Business Outcome rispetto agli Obiettivi di Apprendimento

Questa sezione elenca la tracciabilità tra i Business Outcome e gli Obiettivi di Apprendimento del Certified Tester AI Testing.

La prima parte della tabella mostra il numero di Obiettivi di Apprendimento per ciascun Business Outcome, mentre la seconda parte mostra quali Obiettivi di Apprendimento sono associati a ciascun Business Outcome.

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
BO1	Comprendere lo stato attuale dell'AI, inclusa l'AI generativa.		8							
BO2	Sperimentare l'implementazione e il testing di modelli di machine learning.			10						
BO3	Comprendere il funzionamento e il testing di reti neurali semplici.				2					
BO4	Comprendere le caratteristiche di qualità specifiche dell'AI definite dalla norma ISO/IEC 25059.					3				
BO5	Calcolare e interpretare le metriche di efficienza delle prestazioni funzionali di ML per i modelli di machine learning.						1			

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
BO6	Riconoscere l'ambito e l'importanza dei due livelli di test specifici del testing dei sistemi di machine learning.							3		
BO7	Contribuire allo sviluppo di una strategia di test efficace per un sistema di machine learning.								5	
BO8	Progettare ed eseguire test case per i sistemi di machine learning.									14
LO univoco	learning objective (obiettivo di apprendimento)	Livello K								
1	Introduzione all'Intelligenza Artificiale									
1.1	Introduzione all'AI									
AI-1.1.1	Differenziare i sistemi basati sull'AI dai sistemi convenzionali	K2	X							
AI-1.1.2	Distinguere tra AI ristretta, AI generale e super AI	K2	X							
AI-1.1.3	Spiegare i diversi tipi di tecnologie AI	K2	X							
AI-1.1.4	Spiegare l'AI generativa	K2	X							

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
AI-1.1.5	Confrontare le opzioni hardware disponibili per implementare sistemi di machine learning	K2	X							
AI-1.1.6	Confrontare le opzioni per lo sviluppo e l'hosting dei modelli AI	K2	X							
AI-1.1.7	Riassumere le funzionalità fornite dai framework di sviluppo ML	K2	X							
AI-1.1.8	Spiegare in che modo regolamenti e standard influenzano lo sviluppo e il testing dei sistemi basati sull'AI	K2	X							
2	Caratteristiche di Qualità per i Sistemi Basati sull'AI									
2.1	Caratteristiche di Qualità per i Sistemi Basati sull'AI									
AI-2.1.1	Classificare i comportamenti dei sistemi basati sull'AI in base alle caratteristiche di qualità definite nella norma ISO/IEC 25059	K2				X				
AI-2.1.2	Spiegare le particolari considerazioni che emergono quando l'AI viene utilizzata in sistemi correlati alla sicurezza fisica	K2				X				

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
2.2	Criteri di Accettazione per i Sistemi Basati sull'AI									
AI-2.2.1	Fornire esempi di criteri di accettazione per i sistemi basati sull'AI	K2				X				
3	machine learning									

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
3.1	Introduzione al Machine Learning									
AI-3.1.1	Distinguere tra le diverse forme di ML	K2		X						
AI-3.1.2	Riassumere il workflow utilizzato per creare un sistema ML	K2		X						
AI-3.1.4	Riassumere l'utilizzo dei modelli preaddestrati, del fine-tuning e della Retrieval-Augmented Generation	K2		X						
3.2	Dati per il Machine Learning									
AI-3.2.1	Spiegare le attività relative alla preparazione dei dati	K2		X						
AI-3.2.3	Mettere a confronto l'utilizzo dei dataset di addestramento, di validazione e di test nello sviluppo di un modello ML	K2		X						
3.3	Metriche di Efficienza delle Prestazioni Funzionali di ML per la Classificazione									
AI-3.3.1	Calcolare le comuni metriche di efficienza delle prestazioni funzionali di ML a partire da un insieme di dati di una matrice di confusione	K3					X			

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
3.4	Reti Neurali									
AI-3.4.1	Spiegare la struttura e il funzionamento di una rete neurale profonda	K2			X					
AI-3.4.3	Descrivere le diverse misure di copertura per le reti neurali	K2			X					
4	Testing dei Sistemi Basati sull'AI									
4.1	Introduzione al Testing dei Sistemi Basati sull'AI									
AI-4.1.1	Confrontare la testabilità dei sistemi basati sull'AI bloccati e adattivi	K2		X						
AI-4.1.2	Spiegare perché nel testing dei sistemi basati sull'AI è spesso necessario un approccio statistico	K2		X						
AI-4.1.3	Spiegare le sfide e le soluzioni relative agli oracoli del test per i sistemi basati sull'AI	K2		X						
4.2	Testing dell'AI Generativa e degli LLM									
AI-4.2.1	Spiegare come può essere testata l'AI generativa	K2		X						

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
AI-4.2.2	Implementare il red teaming per i sistemi GenAI	K3		X						

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
4.3	Livelli di Test e Sistemi di Machine Learning									
AI-4.3.1	Riassumere i livelli di test utilizzati per sviluppare i sistemi di machine learning	K2							X	
AI-4.3.2	Spiegare come viene applicato il testing basato sul rischio ai sistemi di machine learning	K2							X	
5	Testing dei Dati di Input per i Sistemi di Machine Learning									
5.1	Testing dei Dati di Input per i Sistemi di Machine Learning									
AI-5.1.1	Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo ai dati di input di un sistema di machine learning	K2						X	X	
AI-5.1.2	Spiegare come eseguire il testing per bias	K2								X
AI-5.1.3	Riassumere le diverse forme di testing della pipeline di dati	K2								X
AI-5.1.4	Spiegare come eseguire il test di rappresentatività dei dati	K2								X

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
AI-5.1.5	Applicare il testing dei vincoli di dataset	K3								X
AI-5.1.6	Spiegare il testing di correttezza delle etichette	K2								X
6	Testing del Modello per i Sistemi di Machine Learning									
6.1	Testing del Modello per i Sistemi di Machine Learning									
AI-6.1.1	Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo ai modelli ML	K2						X	X	
AI-6.1.2	Spiegare lo scopo e il focus della review della documentazione del modello ML	K2								X
AI-6.1.3	Spiegare come viene eseguito il testing dell'efficienza delle prestazioni funzionali di ML per i sistemi probabilistici di machine learning	K2								X
AI-6.1.4	Riassumere l'adversarial testing dei sistemi di machine learning	K2								X

Business Outcome: AI Testing			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8
AI-6.1.5	Utilizzare il testing metamorfico per ricavare test case per un determinato scenario	K3								X
AI-6.1.7	Spiegare come viene utilizzato il drift testing sui sistemi di machine learning operativi	K2								X
AI-6.1.8	Spiegare come overfitting e underfitting vengono rilevati mediante il testing	K2								X
AI-6.1.9	Spiegare come viene utilizzato il testing A/B nel contesto dei sistemi di machine learning	K2								X
AI-6.1.10	Spiegare come viene utilizzato il testing back-to-back nel contesto dei sistemi di machine learning	K2								X
7	Testing dello Sviluppo del Machine Learning									
7.1	Testing dello Sviluppo del Machine Learning									
AI-7.1.1	Fornire esempi di approcci del test utilizzati per la mitigazione del rischio relativo allo sviluppo ML	K2						X	X	
AI-7.1.2	Spiegare le diverse forme di testing del deployment dei sistemi ML	K2								X

14 Appendice C – Release Note

ISTQB CT-AI v2.0 è un aggiornamento e una riscrittura sostanziali della v1.0. A causa della rapida evoluzione della tecnologia AI si è reso necessario un aggiornamento importante. La v2.0 si focalizza chiaramente sul testing dei sistemi basati sull'AI. In seguito alla pubblicazione di ISTQB CT Testing with Generative AI, il capitolo relativo al testing con l'AI è stato completamente rimosso.

Questa release principale ha introdotto le modifiche seguenti:

- Riduzione dell'introduzione generale all'AI
- Inclusione dell'AI generativa e del testing dell'AI generativa
- Consolidamento delle caratteristiche di qualità dell'AI e delle relative sfide
- Riduzione del focus sulle metriche delle prestazioni di ML
- Perfezionamento dei livelli di test: testing dei dati di input e testing del modello ML
- Perfezionamento dei tipi di test
- Rimozione del testing con l'AI
- Esclusione degli ambienti di test per il testing dei sistemi basati sull'AI
- Riduzione del tempo minimo di formazione richiesto da 4 giorni a 3 giorni

15 Indice Analitico

Tutti i termini del testing sono definiti nel Glossario ISTQB® (<https://glossary.istqb.org/>).

accuracy; 36	deriva concettuale; 62
adattabilità funzionale; 23; 25	deriva dei dati; 62
adversarial testing; 61	documentazione del modello ML; 58
AI as a Service; 19	drift testing; 62
AI generale; 16	efficienza delle prestazioni funzionali di ML; 59; 67
AI generativa; 17; 44	EU AI Act; 21
AI ristretta; 15	F1-score; 36
algoritmo ML; 29	fine-tuning; 31
analisi degli impatti disparati; 51	framework di sviluppo ML; 19; 29
analisi esplorativa dei dati; 29; 33; 51	frontier AI; 16
analisi statica; 51	intelligenza artificiale; 15
annotazione multipla; 54	intervenibilità; 24; 26
apprendimento non supervisionato; 16; 28	Large Language Model; 32; 44
apprendimento per rinforzo; 16; 28	machine learning; 15; 16; 28
apprendimento supervisionato; 16; 28	matrice di confusione; 36
associazione; 28	metrica di efficienza delle prestazioni funzionali di ML; 35
attacco; 45; 62	metriche di efficienza delle prestazioni funzionali di ML; 29; 30
autoapprendimento; 24	mitigazione dei rischi sociali ed etici; 22; 24; 26
canary testing; 69	modello ML; 18; 29; 31; 34
classificazione; 28; 35	modello preaddestrato; 32
clustering; 28	non determinismo; 24
controllabilità dell'utente; 23; 25	oracolo del test; 43; 61
copertura dei confini dei neuroni; 39; 71	overfitting; 63
copertura dei neuroni; 39	perceptron; 38
copertura k-multisezione dei neuroni; 39; 70	precision; 36
correttezza funzionale; 22; 25	preparazione dei dati; 32
correttezza funzionale dell'AI; 23	recall; 36
criteri di efficienza delle prestazioni funzionali di ML; 30	red teaming; 45
dataset di addestramento; 34	regressione ML; 28
dataset di test; 30; 34	
dataset di validazione; 30; 34	

relazione metamorfica; 61
rete neurale; 36; 37
Retrieval-Augmented Generation; 32
review; 50; 58
robustezza; 26
robustezza dell'AI; 23
shadow testing; 69
sicurezza fisica; 24; 26
sistema basato sull'AI; 15; 41
sistema basato sull'AI adattivo; 42
sistema basato sull'AI bloccato; 41
sistemi basati sull'AI; 15
spiegabilità; 24
super AI; 16
test case di follow-up; 61
test case sorgente; 61
test di rappresentatività dei dati; 52
test non funzionale; 30; 61
testing A/B; 63
testing back-to-back; 64

testing basato sul rischio; 47
testing dei dati di input; 46; 49
testing dei vincoli di dataset; 53
testing del modello; 46
testing del modello ML; 57
testing del rollback; 69
testing della pipeline di dati; 51
testing delle API; 69
testing dello sviluppo ML; 67
testing di compatibilità dei dispositivi; 69
testing di conversione del modello; 69
testing di correttezza delle etichette; 51; 54
testing di installabilità; 69
testing dinamico; 49; 51
testing esplorativo; 46
testing metamorfico; 61; 62
testing per bias; 50
trasparenza; 23; 24; 26
underfitting; 63
workflow ML; 29; 31; 34