POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Gestionale

Tesi di Laurea Magistrale

STIMA DEI STORY POINTS DI PROGETTI IT ATTRAVERSO L'UTILIZZO DEL MACHINE LEARNING

Relatore

Prof. Giulio Mangano

Candidato

Francesco Davì

ANNO ACCADEMICO 2022-2023

Sommario

1 Introduzione

Il presente lavoro di tesi nasce durante l'esperienza lavorativa svolta all'interno di una società internazionale di consulenza specializzata in innovazione tecnologica e dei processi. Nel dettaglio, l'esperienza si è svolta all'interno dell'ambito del project management all'interno del team che si occupa della fornitura di servizi IT per un grande istituto bancario.

Il fine con tale elaborato è quello di provare ad ottenere un'ottimizzazione dei processi aziendali dedicati alla gestione dei progetti. L'idea nasce dalla volontà da parte della società di migliorare il processo di stima dei tempi e dei costi delle attività progettuali in modo da definire con un livello di dettaglio elevato il peso di una commessa nell'ambito IT.

La prima parte dell'elaborato consisterà in una digressione dottrinale sul machine learning e dei suoi modelli. Si cercherà di fornire una panoramica generale per comprenderne caratteristiche, funzionalità e ambiti in cui può essere utilizzato. Proseguendo, sarà analizzato l'argomento core della tesi. Per raggiungere l'obiettivo primo vi sarà una prima parte dedita all'individuazione delle variabili di principale interesse derivabili da database di storici progetti IT, e l'analisi delle possibili relazioni tra esse. Il proposito è quello di individuare il modello più adeguato all'interpretazione dei vari fattori così da produrre una previsione accurata dei tempi e dei costi.

2 Il machine learning

Con il termine machine learning, o “apprendimento automatico” in italiano, ci si riferisce a una specifica branca dell’informatica che può essere vista come un parente dell’intelligenza artificiale. Non è sempre possibile definire le caratteristiche e le applicazioni del machine learning in maniera semplice perché questo campo è abbastanza ampio e prevede varie tecniche, modalità e strumenti per essere realizzato. In aggiunta, le diverse tecniche di apprendimento e sviluppo danno origine a diverse potenziali applicazioni che ampliano il campo di applicazione del machine learning, rendendo difficile fornire una definizione specifica. Tuttavia, può essere detto che quando il termine “machine learning” è usato, ci si riferisce a vari meccanismi che permettono a una intelligenza artificiale di migliorare le proprie capacità e prestazioni nel tempo. La macchina, quindi, sarà in grado di imparare a eseguire specifici compiti in maniera sempre più performante attraverso l’esperienza, le proprie capacità, risposte e funzioni. Il machine learning fa utilizzo di svariati algoritmi che sapranno prendere decisioni migliori rispetto ad altre o effettuare azioni apprese nel tempo sulla base di nozioni primitive.

2.1 Storia

Oggi l’utilizzo del computer e di strumenti che utilizzano l’intelligenza artificiale e il machine learning è molto diffuso ma il percorso per arrivare a questi risultati è stato molto complicato, visto che era diviso tra sperimentazione e scetticismo. I primi esperimenti per la creazione di macchine intelligenti risalgono ai primi anni Cinquanta del XX secolo, quando alcuni matematici e statistici iniziarono a pensare di utilizzare metodi probabilistici per creare macchine in grado di prendere decisioni in base alla probabilità che si verificasse un evento. Uno dei primi grandi nomi legati al machine learning è Alan Turing. Egli ipotizzò la necessità di creare algoritmi specifici per macchine in grado di apprendere. Negli stessi anni gli studi su intelligenza artificiale, sistemi esperti e reti neurali hanno visto periodi di grande crescita alternati a periodi di abbandono, causati principalmente dalle tante difficoltà incontrate nella possibilità di realizzare i vari sistemi intelligenti, nella mancanza di sussidi economici e nello scetticismo che spesso circondava coloro che cercavano di lavorarci. A partire dagli anni '80, una serie di interessanti risultati ha portato alla rinascita di questo settore della ricerca: una rinascita resa possibile da nuovi investimenti nel settore. Alla fine degli anni '90, l'apprendimento automatico ha trovato nuova linfa in una serie di tecniche innovative legate agli elementi statistici e probabilistici: è stato un passaggio importante che ha permesso quello sviluppo che oggi ha portato l'apprendimento automatico ad essere un ramo di ricerca riconosciuto e molto richiesto.

Introduzione alle reti neurali | MATEpristem (unibocconi.it)

2.2 Funzionamento

A seconda del tipo di algoritmo implementato, cioè in base alle modalità con cui la macchina impara ed accumula dati ed informazioni, si possono individuare differenti sistemi di apprendimento automatico:

- Apprendimento supervisionato, o apprendimento automatico supervisionato, è una sottocategoria del machine learning. È definito dall'uso di set di dati etichettati per addestrare algoritmi in grado di classificare i dati o prevedere i risultati in modo accurato. La centralità di questa tipologia risiede nella capacità di costruire un database di informazioni e di esperienze. Il dispositivo, trovandosi di fronte ad un problema, fa ricorso alle situazioni inserite nel proprio sistema, le analizza e decide quale risposta dare sulla base, appunto, del suo bagaglio di esperienze già codificate. Gli algoritmi che fanno uso di apprendimento hanno la capacità di effettuare ipotesi induttive, ossia ipotesi che possono essere ottenute scansionando una serie di problemi specifici per ottenere una soluzione idonea ad un problema di tipo generale.

- L'apprendimento non supervisionato, o apprendimento automatico non supervisionato, usa degli algoritmi di apprendimento di macchine per analizzare e raggruppare set di dati senza etichette. Questo metodo sfrutta un approccio più indipendente che prevede che le informazioni inserite nella macchina non siano codificate, e non vi è, pertanto, la possibilità di trarre informazioni da esempi precedentemente resi noti. La macchina stessa scheda tutte le informazioni in possesso, le organizza ed impara il loro significato, il loro utilizzo e il risultato a cui esse portano nelle differenti situazioni.

- L'apprendimento automatico di rinforzo è simile all'apprendimento supervisionato di machine learning, ma l'algoritmo non viene addestrato utilizzando dati di esempio. Questo modello impara man mano che procede, utilizzando tentativi ed errori. L'apprendimento per rinforzo rappresenta probabilmente il sistema di apprendimento più complesso, che prevede che la macchina sia dotata di sistemi e strumenti in grado di migliorare il proprio apprendimento e, soprattutto, di comprendere le caratteristiche dell'ambiente circostante. In questo caso, quindi, alla macchina vengono forniti una serie di elementi di supporto, quali sensori, telecamere, GPS eccetera, che permettono di rilevare quanto avviene nell'ambiente circostante ed effettuare scelte per un migliore adattamento all'ambiente intorno a loro.

I tre principali tipi di Machine Learning: Apprendimento Supervisionato, Non Supervisionato e per Rinforzo | by Jacopo Kahl | Medium [accesso novembre 2022]

Supervised learning, cos'è, apprendimento supervisionato - AI4Business [accesso novembre 2022]

Ogni algoritmo, secondo quanto riportato nell'articolo[1], possiede un sistema di apprendimento costituito da tre componenti principali:

- Un processo decisionale: gli algoritmi di apprendimento automatico vengono utilizzati per fare una previsione o una classificazione. Basandosi su alcuni dati in input, che possono essere etichettati o non etichettati, l'algoritmo produrrà una stima di un modello nei dati

- Una funzione di errore che dice quanto il modello si è rivelato accurato nel predire i dati. Se sono presenti esempi noti, una funzione di errore può effettuare un confronto per valutare l'accuratezza del modello

- Un processo di ottimizzazione del modello: se il modello può adattarsi meglio ai punti dati nel set di addestramento, i pesi vengono regolati per ridurre la discrepanza tra l'esempio noto e la stima del modello. L'algoritmo ripeter 'a questo processo di valutazione e ottimizzazione, aggiornando i pesi in modo autonomo fino al raggiungimento di una soglia di accuratezza.

[1] WHAT IS MACHINE LEARNING (ML)? **HTTPS://ISCHOOLONLINE.BERKELEY.EDU/BLOG/WHAT-IS-MACHINE-LEARNING/** ACCESSO [SEPTEMBER 2022]

2.3 Applicazioni del Machine Learning

Una delle principali caratteristiche del machine learning è la sua stretta correlazione con svariati rami della statistica, dell'informatica, dell'ottimizzazione e di molti altri settori delle moderne scienze intelligenti. Incursioni nei diversi campi e settori, infatti, sono molto frequenti e importanti per poter realizzare strutture in grado di risolvere i più svariati problemi che permettono ad una macchina di apprendere secondo le tre modalità tipiche dell'apprendimento intelligente discusse precedentemente. Una sua classica applicazione è quella del riconoscimento vocale, di cui molti smartphone e dispositivi domotici sono dotati. Un altro utilizzo è quello che le aziende utilizzano per realizzare pubblicità online basata sugli interessi degli utenti. I siti web che consigliano articoli che potrebbero essere interessanti attraverso l'utilizzo di cookies basano la loro analisi sul machine learning. Anche il settore finance attraverso la correlazione di eventi, abitudini degli utenti e preferenza di spesa impiega sistemi di questo tipo per le prevenzioni di frodi e furti d'identità e di dati. Anche le banche ne fanno uso per scopi specifici: rilevare, nell'imponente mole di dati che possiedono e che producono, quelle informazioni importanti per identificare nuove opportunità di investimento e aiutare gli investitori o per misurare la solvibilità dei prestiti, il debito totale, la cronologia dei pagamenti e la durata della storia creditizia attraverso l'analisi dei punteggi del credito al consumo. Un altro campo di applicazione è la logistica, dove viene utilizzato per l'analisi dei dati cercando di identificare schemi, bisogni e tendenze per creare nuove rotte più efficienti al fine di incrementare i profitti nei trasporti. Uno dei rami in cui si sta provando sempre di più ad utilizzare l'apprendimento automatico è quello del project management. Grazie alla sua grande capacità di adattamento vi è la possibilità di applicazione in svariati campi che prevedono l'attività di project management, come la stima del tasso di generazione di rifiuti in un processo di demolizione [1] o la predizione dei prezzi delle case [2], fino allo sviluppo sofware. Alcuni esempi pratici di applicazioni sono:

Zillow.com(previsione), che è un marketplace per immobili operante negli Stati Uniti, in cui, oltre ad inserire il prezzo di vendita dell'immobile proposto dal venditore, viene suggerita una "Zestimate", ossia una stima del prezzo di vendita basata su una serie di caratteristiche oggettive. Il margine di errore mediano relativamente all'effettivo prezzo di vendita è di circa il 5%.

La piattaforma Kaggle.com (classificazione) ospita l'evento Data Science Bowl, durante il quale, esperti ed appassionati di data science possono lavorare a progetti socialmente utili.

Nielsen (clusterizzazione) è una multinazionale che raccoglie dati relativi ai prodotti del largo consumo e collabora con panel di consumatori al fine di studiarne le abitudini di acquisto ed il consumo mediatico. In un'analisi svolta nel 2016 sulla propensione allo shopping online, ha applicato tecniche di

clusterizzazione individuando quattro gruppi di e-shopper distinti per frequenza di acquisto online e condivisione di contenuti su internet

3. Stima dei story points nei progetti IT

Nello sviluppo agile, gli story points sono unità di misura per esprimere una stima del lavoro complessivo richiesto per implementare completamente un elemento del backlog di prodotto o qualsiasi altra porzione di lavoro [3]. La stima dei story points consente ai team di sviluppo software di definire meglio l'ambito dei prodotti, dare priorità ai requisiti, allocare risorse, misurare i progressi, prevedere le date di completamento e calcolare il volume di storie che un team può gestire, noto come velocità [4]. A causa della portata in rapida evoluzione dei progetti agile, storie scarsamente stimate possono portare a incertezze e aumento dei costi [5]. I punti della storia vengono in genere stimati attraverso sessioni di pianificazione in cui un team raggiunge un consenso su quanto sforzo che una storia richiede in termini di punti della storia. Un team tiene conto di fattori come la complessità del lavoro richiesto, le competenze del team e le incertezze che possono sorgere [6]. Le stime, tuttavia, sono spesso fatte da diversi membri del team che possono assegnare uno sforzo minimo a determinate storie che corrispondono al loro background. Queste differenze di stima possono portare a incongruenze e "stime".

3.1 Stime predittive mediante tecniche di machine learning

Come sottolineato in precedenza, un progetto software è difficile da valutare e controllare dal punto di vista dei tempi e dei costi. Per questo motivo, la pianificazione di un progetto software è una delle attività più critiche dell'intero processo ed eventuali carenze rischiano di generare inefficienza nella sua gestione. Diventa quindi fondamentale disporre di un dataset storico con le informazioni riguardanti l'effort, la durata e i costi dei progetti passati al fine di sviluppare dei modelli predittivi efficaci. Da un punto di vista ideale, l'utilizzo di metodologie di stima basate su algoritmi di machine learning permetterebbe di controllare e di diminuire in modo significativo i costi associati ai progetti software. Tuttavia, le maggiori criticità legate a questo tipo di progetti sono la scarsità di dati empirici e la difficoltà nel costruire un framework ben definito con relazioni ben chiare tra i descrittori chiave. Inoltre, le metodologie di stima dei costi più utilizzate si basano su un approccio top-down (project-based invece che product-based) che rende più difficile la raccolta di dati rispetto ad un approccio bottom-up in cui le metriche si basano direttamente sul prodotto. Per questi motivi, ad oggi, le metodologie di machine

learning sono scarsamente utilizzate a livello professionale e non risultano ancora abbastanza affidabili, soprattutto nelle fasi iniziali del ciclo di vita del software.

Al fine di sviluppare una metodologia di stima predittiva, è necessario utilizzare una funzione predittiva molto precisa che, a partire dai dati riguardanti i progetti passati, permetta di stimare l'effort e la durata dei progetti futuri. Questa capacità dell'algoritmo di imparare dai dati osservati rappresenta un grande vantaggio nella formulazione di una stima legata a progetti caratterizzati da un gran numero di relazioni complesse. Tra le metodologie di machine learning più utilizzate troviamo: feed-forward neural backpropagation networks, recurrent neural networks, feed-forward neural networks e fuzzy neural network.

3.2 Lavori correlati

Prima della visione di alcuni articoli che propongono una soluzione basata sull'apprendimento automatico per la stima dei story points/costi, è importante dire che esistono differenze importanti tra i progetti open source e quelli commerciali per quanto riguarda la natura dei contributori e delle parti interessate [7]. Ad esempio, i progetti commerciali tendono ad avere un gergo interno e acronimi che potrebbero non esistere o potrebbero essere meno diffusi nello spazio pubblico. Inoltre, un progetto open source di solito ha un gran numero di contributori con un alto tasso di turnover [8], mentre un tipico team agile nell'industria è composto da meno di 15 membri a tempo pieno [9]. Ciò può portare a una creazione di storie più omogenea nei progetti commerciali. La durata dell'iterazione può anche essere più breve nei progetti commerciali a causa delle pressioni sul marketing e sulla tempistica di consegna [10]. Infine, nel mondo open source, i problemi possono spesso rimanere aperti più a lungo [11]; questo può aumentare il livello di discussione e aiutare ad aggiungere più dettagli e contesto alle storie open source.

Sono diversi gli approcci utilizzati negli articoli esaminati:

Chongpakdee etc al [12] utilizzano le impronte digitali dei documenti - una tecnica tradizionalmente utilizzata per rilevare il plagio - per cercare storie simili e utilizzare i loro punti per stimare nuove storie. Questo approccio combina le tecniche Hypergeometric Language Model (HLM) e Random N-gram Sample (RNS) per produrre un punteggio di somiglianza tra i problemi del set di formazione e il problema di interesse del set di test. Gli autori valutano il loro approccio utilizzando la magnitudine media dell'errore relativo (MMRE), che è il rapporto medio dell'errore assoluto, ovvero la differenza

assoluta tra le stime previste e quelle effettive. La valutazione è suddivisa per tipo di problema (come bug, storia, richiesta di modifica). Il loro set di dati include 12 progetti open source, raggiungendo un MMRE medio di 1,135 con il tipo di problema "richiesta di modifica" che ha ottenuto la migliore performance MMRE a 0,26.

Porru et al [13] propongono un'altra soluzione basata su SVM. Includono un progetto industriale insieme a otto progetti open source nella loro valutazione. Gli autori isolano il codice dalle storie ed estraggono le sue caratteristiche separatamente, poiché i frammenti di codice sono semanticamente diversi dalle frasi in linguaggio naturale. Qui, la stima degli story points viene presentata come un problema di classificazione, in cui il modello fa le sue previsioni basandosi su un insieme multi-classe di etichette mappate sulla sequenza di Fibonacci. Il modello ottiene prestazioni migliori del 29% rispetto alla baseline più vicina sul progetto industriale con un'accuratezza media del 64% tra i progetti.

Scott e Pfahl [14] confrontano due modelli di Support Vector Machine (SVM): uno che utilizza le funzionalità degli sviluppatori come input e un altro che utilizza le funzionalità testuali. Le caratteristiche degli sviluppatori incluse sono la reputazione dello sviluppatore, il carico di lavoro, il numero totale di punti e storie completate e il numero di commenti lasciati sui problemi. Le caratteristiche testuali includono il riepilogo e la descrizione dei problemi, il numero di caratteri e gli n-grammi utilizzati per calcolare la frequenza inversa della frequenza dei termini (TF-IDF). Gli autori eseguono una valutazione su otto progetti open source, con il modello basato sulle caratteristiche degli sviluppatori che supera la baseline in soli tre progetti su otto secondo l'errore assoluto medio (MAE). Il documento conclude che un modello che utilizza le caratteristiche degli sviluppatori come input può avere prestazioni fino al 10% migliori in media rispetto alle sole caratteristiche testuali.

3.3 Reti Neurali

Le reti neurali artificiali [19] sono modelli matematici composti da neuroni artificiali che si ispirano al funzionamento biologico del cervello umano, ossia modelli costituiti da interconnessioni di informazioni [Reti neurali, cosa sono? - Applicazioni, limiti ed evoluzione (intelligenzaartificiale.it)]. Sono un sottoinsieme del machine learning e risultano indispensabili per risolvere problemi ingegneristici di intelligenza artificiale legati a diversi ambiti tecnologici come l'informatica, l'elettronica, la simulazione o altre discipline. Il cervello umano elabora le informazioni provenienti dai vari sensi in modo parallelo e distribuisce le informazioni in tutti i vari nodi della rete, non in una memoria centrale; facendo il

paragone con l'informatica tradizionale, i calcoli avvengono in modo seriale e non in parallelo e i dati vengono immagazzinati in una memoria centrale. Allo stesso modo del cervello umano le reti neurali ricevono segnali esterni su uno strato di nodi; ognuno di questi nodi d'ingresso è collegato a svariati nodi interni della rete che, tipicamente, sono organizzati a più livelli in modo che ogni singolo nodo possa elaborare i segnali ricevuti trasmettendo ai livelli successivi il risultato delle sue elaborazioni, che saranno via via sempre più dettagliate. I modelli di apprendimento delle reti neurali, come visto precedentemente per il machine learning in generale, sono i modelli di apprendimento supervisionato, apprendimento non supervisionato e apprendimento automatico di rinforzo. Di seguito vedremo i principali tipi di reti neurali.

3.4 Feed-forward backpropagation networks

Il Feed-forward backpropagation neural network è uno degli strumenti più utilizzati nella formazione di reti neurali feedforward per l'apprendimento supervisionato. Essa serve a calcolare in modo efficiente i gradienti, i quali vengono poi utilizzati per effettuare il training della rete. Si parte dall'Output Layer e si segue una propagazione backwards, andando ad aggiornare i pesi e i bias di ciascun layer in modo da ottenere l'output desiderato.

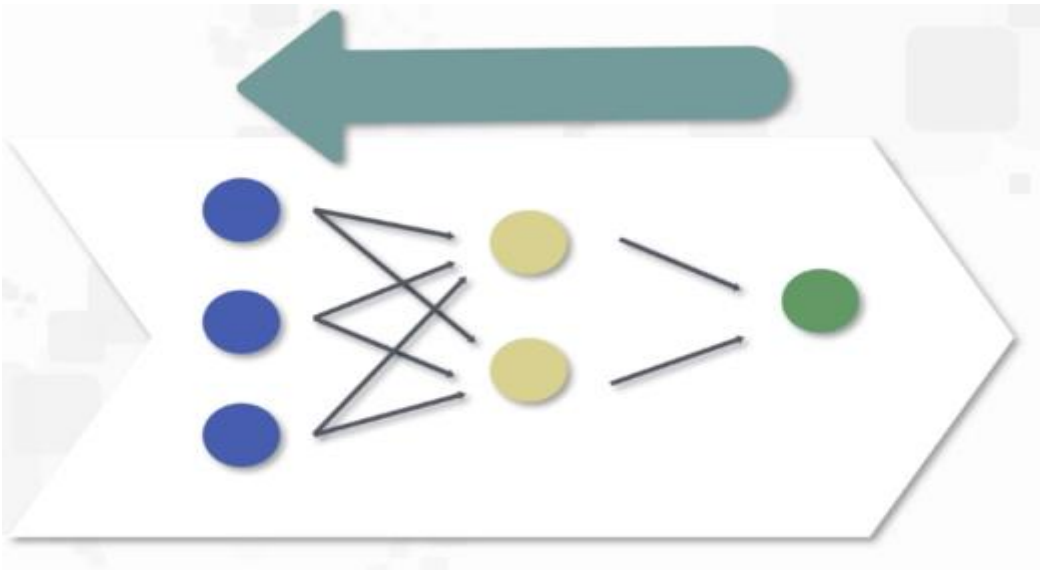


Figura 3.1 Funzionamento del feed-forward backpropagation network

Le reti neurale più avanzate si distinguono per la presenza di cicli che hanno un importante impatto sulle capacità di apprendimento e previsione della rete. In questo caso, la rete neurale seguirà i seguenti passaggi[2]:

1. L'assegnazione dei "pesi" in maniera casuale

2. La propagazione dei dati iniziali con relativa moltiplicazione per i "pesi", per poi passare i risultati attraverso la funzione di attivazione

3. La comparazione dei risultati ottenuti con quelli supervisionati

4 la valutazione dell'errore per capire la bontà dei "pesi" adottati

5 Fase di effettiva backpropagation di aggiustamento dei "pesi" se necessario

Infine, gli step da 1 a 5 vengono ripetuti e i pesi vengono aggiornati dopo ogni osservazione.

Quando tutto il trainset è stato presentato alla rete abbiamo una fase di apprendimento, l'algoritmo di backpropagation può essere interrotto quando l'errore diventa sufficientemente piccolo, deciso a seconda delle proprie necessità. Per effettuare il calcolo del gradiente si utilizzano le seguenti tre equazioni, basate su delle derivate parziali, che servono ad aggiornare i valori di bias, dei pesi e delle attivazioni:

$$\frac{\partial C}{\partial w^{(L)}} = \frac{\partial C}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial w^{(L)}}$$

$$\frac{\partial C}{\partial b} = \frac{\partial C}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial b}$$

$$\frac{\partial C}{\partial a^{(L-1)}} = \frac{\partial C}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial a^{(L-1)}}$$

dove L = Last Layer, L–1 = Second Last Layer, l = layer, C = Funzione di Costo, a = attivazione, w = peso, b = bias, z = w*a+b.

[2] L'ALGORITMO DI UNA BACKPROPAGATION IN UNA RETE NEURALE **L'ALGORITMO DI BACKPROPAGATION IN UNA RETE NEURALE – NETAI** ACCESSO [SETTEMBRE 2022]

Ciascuna derivata parziale legata ai pesi e ai bias viene salvata all'interno del vettore dei gradienti e raggruppata insieme ad altre in uno o più mini-batch. Successivamente, per ciascuna osservazione all'interno del mini-batch, viene eseguita la media degli output per ogni peso e ogni bias. Infine, l'output del vettore è dato dalla media tra gli output di ciascun mini-batch e i pesi e i bias vengono aggiornati di conseguenza secondo le seguenti formule:

$$\mathbf{w(l) = w(l) - learning\ rate * \partial C\ \partial w(L)}$$

$$\mathbf{b\ (l) = \mathit{b} - learning\ rate * \partial C\ \partial b(L)}$$

3.5 Feed-forward networks

Le feed-forward sono le reti neurali più semplici [Introduzione alle reti neurali | MATEpristem (unibocconi.it)], dove ogni input ed il relativo output si muovono in una sola direzione, ovvero non sono presenti cicli o connessioni a ritroso, né tantomeno tra nodi dello stesso layer. Esse consistono in un numero di diverse unità di elaborazione, organizzate per layers, simili a neuroni, un layer di input (Input Layer), uno o diversi layers nascosti (Hidden Layers) e un layer di output (Output Layer). L'input layer è costituito da neuroni che raccolgono i segnali in entrata, i layer nascosti essendo connessi totalmente o parzialmente all'input layer trasferiscono gli input all'output layer che ha uno o più neuroni di output a seconda all'output previsto del modello. L'attivazione o meno di ciascun neurone viene segnata col valore A, e si calcola come la somma dei segnali x in ingresso moltiplicati per un peso w.

$$\mathbf{A = (x_1 w_1 + \ldots + x_n w_n + b)}$$

Inoltre, ciascun neurone se connesso ad un altro ha un peso w (numero double di solito). Il neurone del leyer successivo si ottiene utilizzando la seguente formula:

$$\mathbf{F\ (w_1 A_1 + \ldots + w_n A_n + b) = new\ neuron}$$

dove b è una costante chiamata bias che serve a far adattare meglio la rete neurale e lo si può immaginare come un ulteriore neurone che non accetta valori in ingresso. Con F invece si indica la funzione di attivazione (generalmente una funzione sigmoide) che serve a scalare l'output in modo tale che sia di nuovo compreso tra 0 e 1.

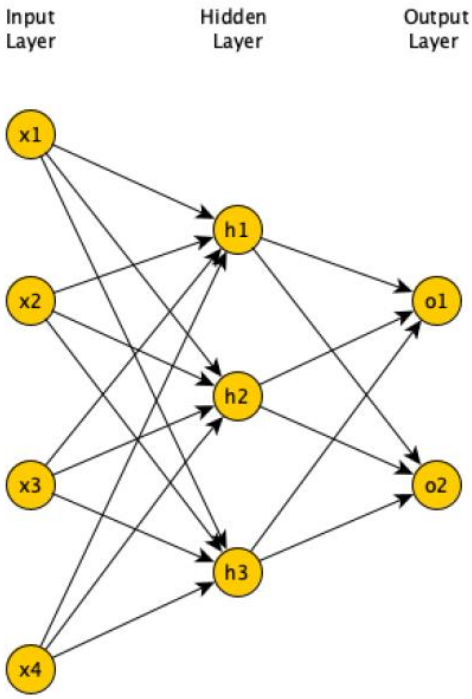


Figura 3.2. Rete neurale Feed-forward con 4 ingressi, 2 uscite e un hidden layer a 3 neuroni

In un sistema del genere l'uscita attuale dipende solo ed esclusivamente dall'ingresso attuale; quindi, in sostanza la rete non ha **memoria** di ingressi avvenuti in tempi precedenti. Tornano molto utili quando gli susseguono senza dipendenza.

3.6 Recurrent neural networks

Le **reti neurali ricorrenti** oltre a sfruttare i dati di addestramento per apprendere, si contraddistinguono per la loro **memoria** in quanto prendono informazioni da input precedenti per influenzare l'input e l'output correnti. È, quindi, una rete neurale in cui esistono **cicli**: i valori di uscita di un layer di livello superiore (più vicino all'uscita) vengono utilizzati come ingresso per un layer di livello inferiore (più vicino all'ingresso).

Il ciclo può essere così schematizzato[3]:

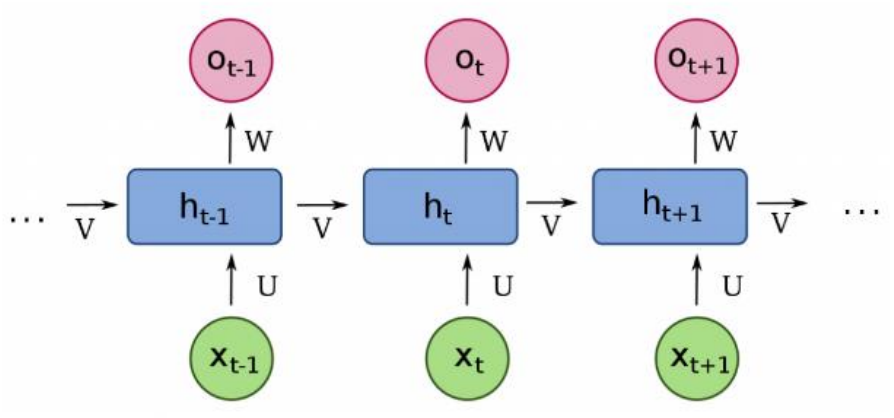


Figura 3.3. Rete neurale ricorrente bidirezionale

Dove è possibile osservare, considerando l'istante t, che l'output O(t)è funzione sia dell'ingresso X(t) sia di V, che per effetto del ciclo, corrisponde all'uscita precedente O(t-1).

La RNN può prendere uno o più vettori di input e produce uno o più vettori di output, i quali sono influenzati sia dai pesi legati agli input sia da vettori "nascosti" che rappresentano il contesto legato ai precedenti input e output e vengono aggiornati in modo ricorrente. Di conseguenza, lo stesso input può produrre output diversi a seconda degli input applicati nei cicli precedenti.

3.7 Fuzzy neural network

Si tratta di una logica sfumata con infiniti stati intermedi tra zero e uno, consente ai computer di rappresentare meglio la complessità della realtà ma anche il linguaggio naturale degli uomini. La logica fuzzy è una metodologia di calcolo basata su diversi livelli di verità piuttosto che sulla solita logica booleana "vero o falso" (1 o 0). In conseguenza di tali assiomi la logica fuzzy è adatta per quando si devono prendere decisioni con dati imprecisi ma con incertezze chiare.

[3]LE RETI NEURALI RICORRENTI **LE RETI NEURALI RICORRENTI - MAGGIOLIDEVELOPERS (DEVELOPERSMAGGIOLI.IT)** ACCESSO [OTTOBRE 2022]

Un sottoinsieme fuzzy A è descritto da una funzione di membership μ_A che mappa un più ampio insieme **U** all'interno dell'intervallo [0,1]. Può essere descritto con il formalismo seguente:

$\mu_A(x) = 0$ se x non appartiene ad A

$\mu_A(x) = 1$ se x appartiene ad A

$0 \leq \mu_A(x) \leq 1$ se è possibile che x appartenga ad A

Più il valore di μA è vicino a 1 e più è alta la possibilità che x appartenga ad A.

Una rete neurale fuzzy è costituita da un algoritmo di machine learning che prende i suoi parametri da un insieme fuzzy e può essere utilizzata per risolvere un problema matematico per cui non esiste un preciso modello. Un sistema neurofuzzy è costituito da una particolare rete neurale feed-forward formata da tre diversi layer:

il primo layer corrisponde alle variabili di input, il secondo rappresenta le regole dell'insieme fuzzy e il terzo corrisponde alle variabili di output. In qualsiasi momento del processo di learning, esso può essere rappresentato come un insieme di regole fuzzy, che devono essere inizializzate a monte del processo e possono essere considerate come dei prototipi dei dati di addestramento. Esistono due diversi tipi di reti neurali fuzzy (Nauck et al., 1997):

• Cooperative fuzzy neural network: la rete neurale e il sistema fuzzy lavorano in modo indipendente l'uno dall'altro. Le funzioni di membership dell'insieme fuzzy e i dati dell'addestramento vengono utilizzati dalla rete neurale per determinare le regole fuzzy e i pesi ad esse assegnate in base alla loro influenza.

• Hybrid fuzzy neural network: il sistema fuzzy è interpretato come una speciale rete neurale e l'architettura di questa tipologia di rete fa in modo che la rete neurale e il sistema fuzzy non debbano comunicare tra loro. I fuzzy set costituiscono i pesi mentre gli input, gli output e le regole sono modellate come neuroni. Le funzioni di membership vengono utilizzate come controller per formalizzare le regole fuzzy in termini di linguaggio.

Le reti neurali fuzzy vengono utilizzate per realizzare stime predittive dei costi di progetti software, dal momento che sono molto utili nel modellare e controllare sistemi non lineari. Tuttavia, questa metodologia presenta dei limiti: innanzitutto, essa non è universale in quanto si basa sulla self-

organization delle funzioni di membership e spesso non è facile associare un significato fisico ai parametri chiave che, di conseguenza, risultano difficili da inizializzare. Inoltre, spesso la rete presenta una struttura complicata, i parametri da calibrare ad ogni step risultano essere troppi e ciò causa un notevole rallentamento del processo.

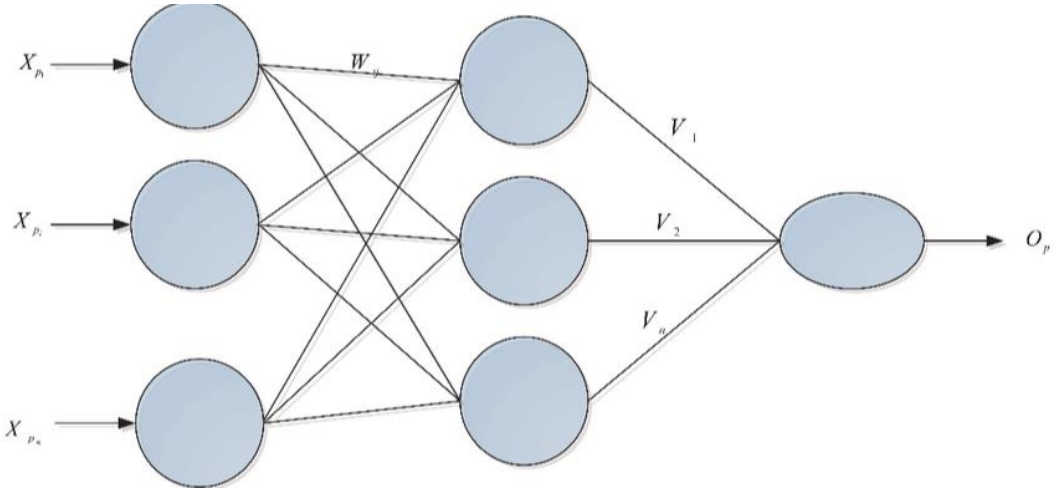


Figura 3.4. Fuzzy neural network

3.8 Modelli white-box e black-box

Saper fornire agli stakeholders una spiegazione esaustiva riguardo ai risultati della previsione ottenuti è una delle principali criticità legata all'utilizzo di algoritmi di machine learning. Per tale ragione, è indispensabile valutare attentamente il trade-off tra la precisione che l'algoritmo può garantire, e l'interpretabilità dei risultati che ne possono derivare, ogni qualvolta ci si trovi a dover scegliere il modello di apprendimento automatico da adoperare. È possibile suddividere i modelli in due macrocategorie principali[4]:

• Black-box model: il cosiddetto modello a scatola nera, [What is Black Box Algorithm - Definition and Examples (arimetrics.com)] capace di fornire un elevato livello di precisione ma risulta di difficile comprensione il funzionamento interno dell'algoritmo; gli utenti, infatti, possono solo osservare la relazione input-output. Inoltre, questo modello non fornisce una stima del peso di ciascuna caratteristica nelle previsioni del modello, ne è facile capire come interagiscano i diversi fattori. In questa categoria rientrano le reti neurali;

• White-box model: i modelli a scatola bianca sono decisamente più semplici da comprendere, ma offrono una capacità predittiva inferiore e non sono sempre in grado di modellare la complessità intrinseca del set di dati. In questa categoria rientrano modelli come la regressione lineare e gli alberi decisionali.

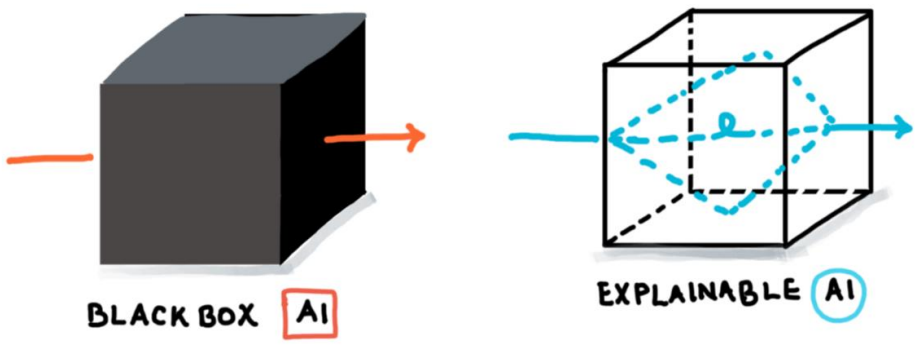


Figura 3.5. Operazioni nei modelli Black-box e White-box

La chiarezza di un modello è legata a due proprietà della funzione di risposta *f(x)*, la quale definisce la relazione input (caratteristica *x*) -output (target *f(x)*) del modello. La *linearità* è la proprietà per cui, in una funzione di risposta lineare, l'associazione tra una caratteristica e il target si comporta in modo lineare. Se una caratteristica cambia in modo lineare, ci si aspetta che anche il target cambi in modo lineare a una velocità simile. La *monotonia*, caratteristica che prevede che in una funzione di risposta monotona, la relazione tra una caratteristica e l'obiettivo sia sempre unidirezionale, crescente o decrescente.

L'interpretabilità del modello è intrinsecamente legata alla sua complessità. I modelli lineari presentano lo stesso comportamento nell'intero dominio delle caratteristiche e sono, quindi, interpretabili globalmente. La relazione input-output è spesso limitata in termini di complessità e le interpretazioni globali si basano su interpretazioni locali. Per modelli più complessi, il comportamento globale del modello è di più difficile lettura e sono necessarie interpretazioni locali di piccole regioni delle funzioni di risposta. I modelli di regressione lineare presentano funzioni di risposta lineari e monotone, mentre le reti neurali presentano funzioni di risposta non lineari e non monotone. Si può concludere che i modelli white-box sono preferibili quando sono richiesti risultati semplici e di facile interpretazione, ma i vincoli legati alla linearità della funzione di risposta possono portare alla perdita di alcuni dati. Al contrario, i modelli black-box, generalmente più precisi, vanno investigati a livello locale poiché solo in quella circostanza presentano caratteristiche di linearità e monotonia.

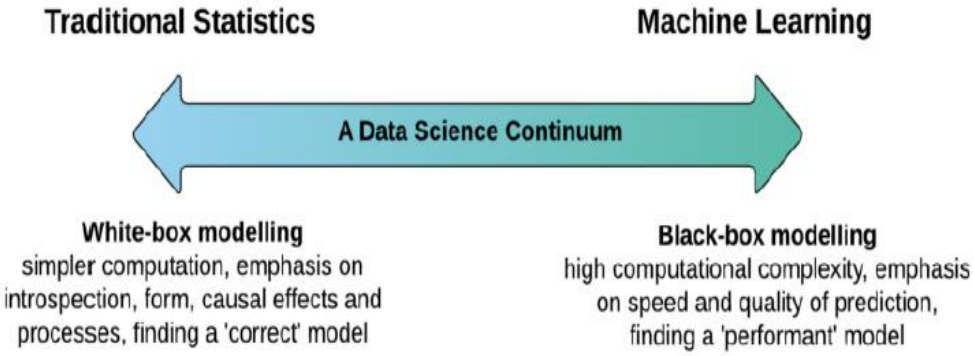


Figura 3.6. White Boe &Black Box

[4]BLACK-BOX VS. WHITE-BOX MODELS **BLACK-BOX VS. WHITE-BOX MODELS. MOST MACHINE LEARNING SYSTEMS REQUIRE… | BY LARS HULSTAERT | TOWARDS DATA SCIENCE** ACCESSO [OCTOBER 2022]

4 Caso di studio:

Il caso di studio di studio nasce durante un percorso lavorativo all'interno di una società di consulenza in ambito IT. L'azienda in questione opera nei settori di tecnologia, software embedded, reti di telecomunicazioni e ingegneria. La software factory in cui si è svolto il percorso lavorativo si occupa di realizzare attività per il supporto di processi aziendali e all' implementazione di nuovi processi innovativi per un noto istituto bancario italiano.
I vari progetti che vengono realizzati seguono un iter composta dalle seguenti fasi:

1. Analisi funzionale
2. Pianificazione e stima dei costi/tempi
3. esecuzione del progetto
4. consegna del prodotto finito al cliente

l'analisi funzionale viene fornita dalla direzione sistemi informativi della banca, al suo interno sono contenute le specifiche tecniche e i requisiti qualitativi del progetto da realizzare. È il primo contatto che mette in azione l'attività progettuale. Da questo momento i tecnici dell'azienda di occupano del calcolo dei costi e dei tempi necessari per la realizzazione del progetto fornendo una stima che passerà al vaglio del project manager e successivamente sarà fornita al cliente. Il cliente potrà così valutare e accettare la proposta, definendo delle deadline progettuali e di conseguenza il go ufficiale all'attività.

Seguiranno le seguenti attività preliminari:

- Analisi e pianificazione: l'obiettivo di questa fase è quello di analizzare i requisiti del progetto per creare una Work Breakdown Structure (WBS). Per effettuare una stima più precisa vengono individuati dei work packages al suo interno, ognuno dei quali può essere visto con un'attività da eseguire. I tecnici si occupano della stesura dei casi d'uso per poter determinare la dimensione e la complessità del codice e consentire la definizione accurata delle richieste del cliente. Gli use case diagrams, basandosi sugli standard UML, pongono in evidenza le interazioni tra il sistema con gli applicativi esterni, mostrano gli attori coinvolti e la sequenza di azioni da svolgere per la completa funzionalità del software, le potenziali condizioni da verificarsi, le motivazioni per cui l'utente desidera interfacciarsi con il programma, gli eventuali errori e le possibili risposte del sistema. L'insieme di tutti i casi d'uso rappresenta l'intera funzionalità da sviluppare e fornisce un'idea della dimensione totale del progetto. Sempre in questa fase, dal punto di vista del management, vengono definite le milestone di progetto, pianificate le fasi delle varie attività e assegnate le risorse ai vari work packages.

- Stima dei tempi e dei costi: una volta definiti i work packages e valutate le relative durate, si analizzano le possibili relazione e interdipendenze operative che vi sono tra le varie attività, in modo da stabilire l'ordine di esecuzione in cui queste devono essere realizzate. Ad ogni work package viene assegnato un numero di story points relativo all'effort richiesto per essere realizzato. Gli story points vengono calcolati attraverso l'utilizzo di due principali metodologie, il metodo dei fuction points, e il metodo per analogia. Il primo metodo si basa sull'analisi della tipologia dei requisiti forniti, sul numero di informazioni in input e in output e sulla quantità di memoria necessaria; queste informazioni vengono elaborate e viene generata una misurazione quantitativa che definisce il numero di fuction points totale che occorrono per ogni work package e conseguentemente per il progetto. Il secondo metodo considera le analogie con progetti passati e l'effort che è stato richiesto precedentemente.

  La stima fornita dai tecnici sotto forma di effort viene valutata e validata dal project manager, che ha il compito di convertire l'effort in termini di costi e tempi necessari alla realizzazione del progetto. La conversione avviene tramite un fattore di conversione costaste, che sulla base dell'esperienza del management è stato definito di 1:1,5, ovvero ad 1 story points corrispondono 1,5 giorni lavorativi per ogni risorsa impiegata. Una volta definito lo sforzo necessario in termini di tempo si moltiplica il numero totale di giorni necessari per una tariffa fissa giornaliera che cambia a seconda del livello di seniority del gruppo di lavoro impiegato. A seguire viene riportato un esempio di stima.

Stima Costi Progetto NOME PROGETTO				
Funzioni	**SP**	**GG/Uomo a Tariffa Media**	**Importo**	**Area**
Requisiti				
Recepimento requisiti e coordinamento	1	X	X	Requisiti
Analisi				
Documentazione tecnica	0	X	X	Requisiti
Sviluppo software				
GOLIA				
Adeguamento OM	1	X	X	Sviluppo
Implementazione connettore JDBC	4	X	X	Sviluppo
Modifica flusso	1	X	X	Sviluppo
Adeguamento logiche di business	2	X	X	Sviluppo
MOCK	2	X	X	Sviluppo
Adeguamento connettore WS	2	X	X	Sviluppo
Test Unitari				
Implementazione ed esecuzione test dei flussi	1,5	X	X	Test
esecuzione test	0,5	X	X	Test
Implementazione ed esecuzione test dei WS	1	X	X	Test
Implementazione ed esecuzione test dei JDBC	1	X	X	Test
Rilascio				
Gestione dei processi di change	0,5	X	X	Change
Garanzia 1 anno	2	0	X	
Totale	17,5	0	X	
	IVA	22%	X	
	Totale IVA inclusa		X	

Figura 4.1 Template di stima dei costi e dei tempi

4.1 Realizzazione del progetto

Ogni progetto viene realizzato attraverso il passaggio nelle seguenti fasi:

- Sviluppo
- Application test
- System test
- User acceptance testing
- Produzione

Figura 4.2 Milestones della fase di esecuzione del progetto

4.1.1 Sviluppo

La parte di sviluppo è la fase la parte centrale del progetto in cui l'attività core è quella della stesura del codice sorgente, che permette l'esecuzione delle funzionalità e dei servizi richiesti del cliente. Questa fase viene eseguita all'interno di un ambiente locale che si appoggia a dei server fisici e a un cloud in cui sono installati gli applicativi necessari alla stesura del codice. La programmazione totale del codice viene suddivisa in moduli più piccoli, ognuno dei quali è testabile dallo sviluppatore che può valutarne la correttezza mediante Unit Test. I singoli moduli verranno successivamente assemblati per comporre il prodotto finale. La maggiore criticità di questa fase è scrivere un codice in grado di funzionare anche quando verrà integrato con il codice preesistente.

4.1.2 Application test

Sulla base di quelle che sono le specifiche recepite dal programmatore, egli stesso scrive i test basilari che il suo prodotto dovrà superare in modo da essere conforme ai requisiti. Tali test vengono avviati nel momento in cui avviene la compilazione del codice e questa fase prende il nome di Application test. L'ambiente in cui viene realizzata questa fase viene definito *Svio* e rappresenta il primo livello di validazione interna del software. In questa fase si cerca di individuare l'eventuale presenza di grandi

errori legati alla scrittura del codice e alla sua correzione tramite debug manuale. Questo passaggio è molto utile in quanto si riescono a verificare le funzioni fondamentali molto velocemente. Una volta finito il passaggio dalla fase di Application il codice viene inserito all'interno di un'unità di cambiamento (UDC), che incapsula tutte le evolutive del software relative ad un progetto e le accompagnerà verso le fasi successive.

4.1.3 System test

Una volta all'interno dell'UDC le evolutive passeranno dall'ambiente *svii* all'ambiente di *Test*. In questa fase saranno effettuati dei test puntuali per verificare che tutti i requisiti definiti nell'AFU siano soddisfatti.

Una volta all'interno dell'ambiente di *Test,* un passaggio chiave è quello dell'*Integration Test,* dove vengono verificate le funzionalità delle varie parti sviluppate e la corretta compatibilità tra esse una volta che vengono integrate e assemblate per formare il prodotto finale. In questo processo di "*debug*" vengono così rimosse le prime anomalie di interazione. A questo punto si procede integrando il nuovo codice con quello preesistente. I test eseguiti in questa fase verificano sia l'integrazione tra il nuovo codice e le periferiche esterne, sia tutte le funzionalità che il software deve rispettare. Ciò avviene fornendo un input al sistema e verificando che esso fornisca un output coerente con la funzionalità richiamata (black-box testing). In questa fase vengono anche testati gli aspetti legati all'utilizzatore finale, quali l'usabilità, le interfaccia grafiche ed eventuali tempi di recovery. Altri test molto importanti sono quelli di *No Regression,* utili ad individuare possibili anomalie dovute all'interazione tra il nuovo e il vecchio codice. Ogni qualvolta che un test fallisce verrà aperto un defect, che comunica e formalizza un'anomalia che dovrà essere risolta dai programmatori che si occupano dello sviluppo. Una volta effettuati tutti i casi di test programmati e chiusi i defects aperti a causa delle anomalie riscontrate, potrà essere redatta la documentazione di riepilogo delle operazioni effettuate in questa fase, che servirà alla fase di User Acceptance Test. Con la consegna di tale documentazione si conclude la fase di *System test.*

4.1.4 User Acceptance Test

Per quanto un requisito possa essere dettagliato nell'AFU fatta inizialmente, gli sviluppatori devono realizzare una funzionalità inizialmente inesistente, e che spesso, quindi, può essere soggetta ad interpretazione. Proprio per la consapevolezza di queste possibili difformità, viene definita un'ultima fase di UAT prima del lancio in produzione affinché chi ha effettuato le richieste funzionali convalidi il software sviluppato. Inoltre, tramite questa fase l'eventuale riscontro di una difformità e una modifica del codice risultano nettamente meno costose per l'azienda rispetto ad una modifica del software una volta che è stato rilasciato in produzione. Ovviamente, è importante definire il tempo necessario a questa fase di test e alle eventuali risoluzioni di anomalie, in modo da rispettare la pianificazione iniziale del progetto. In questa fase un campione di utenti business viene selezionato per svolgere delle prove di user experiece sulle funzionalità sviluppate. Anche alla fine di questa fase è necessario rilasciare una documentazione che descriva dettagliatamente i test effettuati e le eventuali variazioni apportate.

4.1.5 Rilascio in produzione

Il processo visto finora descrive tutti i passaggi attraverso cui ogni progetto passa per arrivare alla produzione. Ovviamente all'interno della software factory si in contemporanea più progetti relativi al software in lavorazione e, quindi, più UDC in parallelo. Per rilasciare tutte le UDC in produzione nello stesso momento, è stato creato un ambiente di *Pre-Production,* un ambiente analogo a quello di produzione, all'interno del quale convergono tutte le UDC pronte al rilascio e che conseguentemente non devono subire più modifiche. Così facendo, per rilasciare tutte le UDC in simultanea, è sufficiente sincronizzare l'ambiente di *Production* con quello di *Pre-Production* modificando i referenti ai database. A seguito del rilascio in produzione, inizia una fase di un anno detta *Follow Up,* che nasce dall'esigenza di far fronte ai problemi che potrebbero sorgere lato utente con l'utilizzo del sito. In questo anno il fornitore si impegnerà ad effettuare la manutenzione necessaria in caso di malfunzionamenti del codice prodotto.

4.2 Analisi del database

Il metodo di stima utilizzato dall'azienda presso cui è stata svolta l'esperienza lavorativa è il processo di analogia. Tale metodologia risulta molto semplice ed evita la consultazione di esperti o l'utilizzo di algoritmi complessi ma porta con sé limiti insiti nella soggettività di valutazione.

Il processo per analogia è un processo prolisso e dispendioso di raccolta di dati storici che può portare ad imprecisioni dettate da errori di analisi. Inoltre, può capitare che due progetti presentino analogie solo in linea teorica, e dimostrarsi, solo in un secondo momento molto differenti tra loro.

Tuttavia, l'integrazione di un sistema di intelligenza artificiale richiede, a monte, uno studio profondo dei fattori che descrivono un progetto, in modo tale da creare un framework solido e delineato ed analizzare ed interpretare i suoi risultati in output.

Per la creazione di un framework che rispecchi le caratteristiche sopraindicate è stato necessario in primis raccogliere dati storici utili. A questo punto i dati sono stati analizzati e filtrati, eliminando gli outliers. Per l'eliminazione degli outliers e stato definito un intervallo di accettazione come lo spazio compreso tra i quantili q_{10} e q_{90}. In questo modo si è creato un database su cui basare lo studio. A questa fase segue una fase di individuazione delle variabili prendere in considerazione. Solo in seguito sarà possibile osservare le relazioni emerse tra i vari descrittori, analizzando i risultati della stima elaborata attraverso l'utilizzo di algoritmi di tipo black-box. Infine, si verificherà l'affidabilità degli output ricavati confrontando l'errore standard ottenuto con quello di una metodologia classica come la regressione lineare.

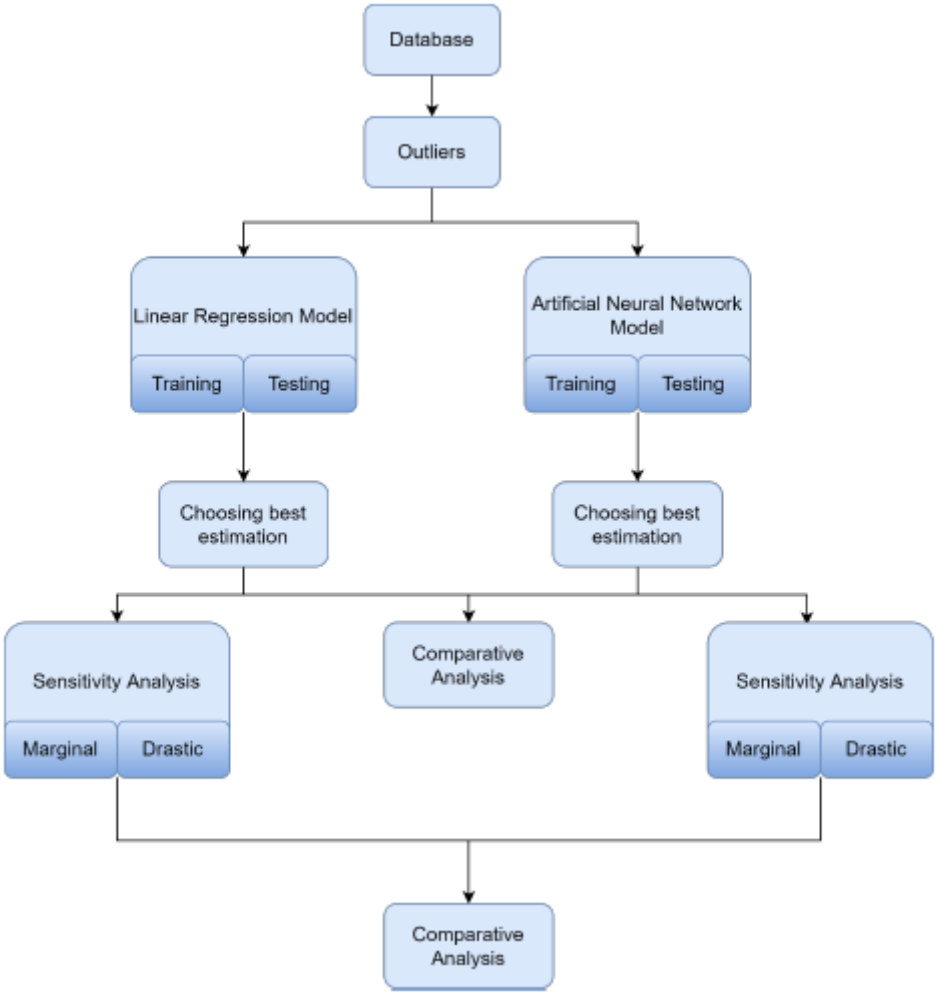


Figura 4.3 Modello della procedura da eseguire per la creazione del framework di stima

Di seguito verranno analizzate le variabili che si è ritenuto essere più esaustive e rilevanti nella determinazione di una stima per un progetto IT; verranno illustrati i risultati ottenuti con l'utilizzo di diversi algoritmi ed analizzati i risultati e i relativi errori. Così facendo sarà possibile stabilire il modello che meglio si adatta alla spiegazione delle variabili prese in considerazione e delle loro relazioni. Il proposito è quello di riuscire a creare un modello migliore per l'elaborazione di una stima di un progetto, che possa essere integrato all'interno del processo attuale.

4.3 Individuazione e analisi delle variabili

L'identificazione delle variabili coinvolte nell'analisi è il primo step chiave per raggiungere l'obbiettivo.

In questa fase si cercherà di identificare i parametri che consentono di valutare le relazioni rilevanti per la creazione del modello.

Il database preso in esame si compone di 800 attività circa, sviluppate all'interno dell'azienda al momento del caso di studio. I parametri presi in considerazione sono i seguenti:

- COMPLESSITA': è una metrica che indica il grado di difficoltà del progetto. Considera il numero di requisiti relativi alle singole attività progettuali che caratterizzano il progetto. In generale, una maggiore dimensione funzionale del prodotto è associata a una maggiore complessità e porta a una maggiore durata e costo [17]. Baccarini [18] ha proposto di definire la complessità del prodotto come le «molte e varie parti interconnesse» della differenziazione e dell'interdipendenza. Sulla base dei requisiti necessari viene quantificato lo sforzo necessario tramite la tecnica dei Fuction points (1979 A. J. Albrect, della IBM Corporation). Con tale tecnica si contano le funzioni in output da fornire all'utente, il numero di funzioni in input da fornire al software e il numero di interfacce; dividendo il tutto per la produttività attesa degli sviluppatori (numero dei punti funzionali che sono in grado di sviluppare in un mese) si ottiene la stima dell'impegno tecnico necessario. Su tale assunto, basandosi sui dati storici in possesso sono state create 4 fasce di differente complessità su cui viene realizzata la classificazione delle attività.

- CONOSCENZA PERIMETRO: è la misura del grado di conoscenza che il project manager e il team di sviluppo hanno sul perimetro del progetto. Tale metrica si ottiene moltiplicando le ore consuntivate dagli sviluppatori sul perimetro di riferimento con la media dei Cost Indexes associati alle varie attività progettuali. Tale risultato viene poi confrontato con i dati storici in possesso del project manager e viene espresso in una scala che va da 1 a 5 dove 5 sta ad indicare una conoscenza ottimale del perimetro.

$$\sum_{i=0}^{n} hi \times \frac{\sum_{i=0}^{n} CIi}{n}$$

- ESPERIENZA GRUPPO: tale metrica rappresenta il grado di esperienza media del gruppo di lavoro a cui un'attività verrà assegnata. Per questa metrica sono stati considerati 4 possibili livelli di esperienza, dove 4 viene assegnato a risorse con grado più alto di seniority. Facendo infine la media si ottiene un livello di esperienza del gruppo compreso tra 1 e 4.

- NUMERO MEDIO DEFECTS: tale metrica considera il numero medio dei defects storici per una determinata attività. Per defect si intende un'anomalia riscontrata durante la fase di test del software. Trattandosi di prodotti poco tangibili, vi è un'intrinseca complessità nel definire il grado di bontà. Pertanto, il numero dei defects rilevati è una valutazione del livello di qualità raggiunto. Per ogni attività avviene una tracciatura dei defects ricevuti dove vengono inseriti nello storage aziendale sia le problematiche riscontrate sia il programmatore che si è occupato di risolverla.

- SP: story points, è la variabile target del modello. Sono unità di misura molto utili nella stima di progetti software. Esprimono una stima del lavoro complessivo richiesto per implementare completamente un elemento del backlog di prodotto. Il project manager assegna gli story points in base alla complessità, alla quantità di lavoro richiesto e all'incertezza. I valori vengono assegnati per suddividere in modo più efficace il lavoro in porzioni più piccole, così da poter affrontare gli aspetti più incerti. Con l'analisi che verrà condotta si tenterà di stabilire le relazioni e le dipendenze tra questo fattore e le altre variabili prese in esame, in modo da individuare un modello capace di stimare tali story points.

PROGETTO	ATTIVITA'	CONOSCENZA PERIMETRO	NUMERO MEDIO DEFECTS	ESPERIENZA GRUPPO	COMPLESSITA'	SP	TOT SP
Progetto 1	attività 1	4	3	2	1	2	75
	attività 2	4	1	2	1	4	
	attività 3	4	6	2	4	8	
	attività 4	4	2	3	2	11	
	attività 5	4	6	2	1	2	
	attività 6	4	8	2	1	2	
	attività 7	4	2	3	4	8	
	attività 8	4	9	2	1	12	
	attività 9	4	2	2	3	3	
	attività 10	4	2	2	4	11	
	attività 11	4	1	2	1	10	
	attività 12	4	1	3	4	2	
Progetto 2	attività 13	3	9	4	2	12	49
	attività 14	3	6	2	4	5	
	attività 15	3	7	2	3	4	
	attività 16	3	3	2	4	6	
	attività 17	3	9	4	3	10	
	attività 18	3	1	4	1	12	
Progetto 3	attività 19	2	9	3	2	3	109
	attività 20	2	5	3	1	2	
	attività 21	2	2	3	4	1	
	attività 22	2	2	3	3	5	
	attività 23	2	7	3	3	11	
	attività 24	2	8	3	2	6	
	attività 25	2	4	3	4	7	
	attività 26	2	5	2	1	8	
	attività 27	2	3	3	3	10	
	attività 28	2	6	3	4	4	
	attività 29	2	7	2	4	3	
	attività 30	2	1	2	1	10	
	attività 31	2	8	2	1	9	
	attività 32	2	7	2	3	9	
	attività 33	2	9	3	2	5	
	attività 34	2	6	4	2	6	
	attività 35	2	4	4	1	10	

Tabella 1. Database delle attività progettuali

4.4 Implementazione dei modelli

L'implementazione dei modelli è stata effettuata attraverso l'utilizzo del software *Anaconda*, una distribuzione open source dei linguaggi di programmazione Python e R. molto utile per il data scientist. In particolare, è stato utilizzato l'ambiente interattivo *Jupyter* per la creazione di algoritmi in grado di analizzare ed elaborare il database utilizzato.

Il primo step del processo di studio del database consiste nell'analisi della correlazione tra le variabili individuate, ovvero la tendenza che hanno due variabili a variare insieme. Il coefficiente di correlazione può essere compreso in un intervallo [-1.00; +1.00] dove con -1.00 si indica una correlazione perfetta negativa e con +1.00 si indica una correlazione perfetta positiva.

Fornendo una veloce illustrazione, valgono le seguenti regole:

- Coefficiente di correlazione > 0 evidenzia una correlazione positiva tra le due variabili analizzate; all'aumentare del valore di una, si ha un incremento del valore dell'altra.
- Coefficiente di correlazione = 0 evidenzia una totale assenza di correlazione tra le due variabili analizzate.
- Coefficiente di correlazione < 0 evidenzia una correlazione negativa tra le due variabili analizzate; all'aumentare del valore di una, si ha un decremento del valore dell'altra.

Inoltre, si ricorda che si possono distinguere tre entità di correlazione:

- Forte: se il valore assoluto del coefficiente di correlazione è maggiore di 0.7.
- Moderata: se il valore assoluto del coefficiente di correlazione è compreso tra 0.3 e 0.7.
- Debole: se il valore assoluto del coefficiente di correlazione è minore di 0.3.

In figura 4.3 è riportata la matrice di correlazione ottenuta:

Matrice di Correlazione					
	SP	N.MEDIO DEFECT	COMPLEX	EXP.GRUPPO	CONOS.PERIM
SP	1.00	0.431	0.863	-0.799	-0.324
N.MEDIO DEFECT	0.431	1.00	0.702	-0.367	-0.563
COMPLEX	0.863	0.702	1.00	0.395	-0.035
EXP.GRUPPO	-0.799	-0.367	0.395	1.00	-0.031
CONOS.PERIM	-0.324	-0.563	-0.035	-0.031	1.00

Figura 4.4 Matrice di correlazione

Analizzando l'output ottenuto, si evince che la variabile *Story Points* presenta una moderata correlazione positiva con la variabile *Numero medio dei defects*, una correlazione forte positiva con la *Complessità*, una forte correlazione negativa con l'*Esperienza del gruppo* e una correlazione moderata negativa con la variabile *Conoscenza del perimetro*.

I risultati ottenuti sono in linea con quanto era possibile aspettarsi. Analizzando le correlazioni tra le variabili indipendenti e la variabile target, è logico pensare che all'aumentare della complessità aumenti notevolmente il numero di story points necessari, così come ci si può aspettare che aumentando il livello di esperienza del gruppo di lavoro l'effort risulti diminuire.

Sempre in linea con le previsioni, la variabile numero medio dei defects risulta sì mediamente correlata positivamente agli story points ma risulta molto correlata alla complessità dell'attività da svolgere, in quanto se è più incertezza aumenta molto il numero dei defects prodotti. Contrariamente, tende a diminuire all'aumentare dell'esperienza del gruppo e della conoscenza del perimetro in quanto avendo più informazione e più esperienza si tende a commettere meno errori.

Ciò che sorprende un po' è la relazione tra story points e conoscenza del perimetro, che mostra come non è detto la conoscenza del perimetro tende sì a far diminuire il numero di story points necessari ma non è un fattore di cruciale importanza.

Un altro modo di vedere la matrice di correlazione è una raffigurazione grafica chiamata heat map, dove l'intensità del colore aumenta con crescere delle relazioni. Si avrà una bassa intensità per le relazioni deboli e un'alta intensità per le relazioni forti.

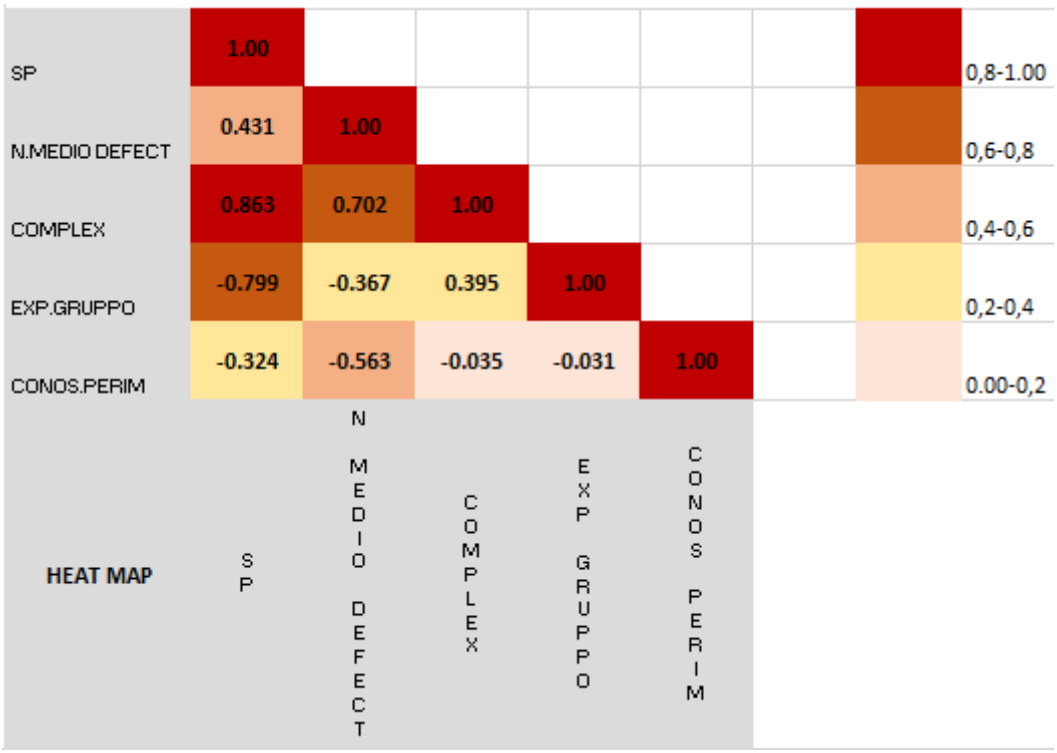


Figura 4.5 Heat map

L'analisi effettuata finora tramite la correlazione, tuttavia, presenta dei limiti legati all'impossibilità di verificare la presenza o l'effetto di altre variabili oltre alle due esaminate, e alla mancanza di informazioni riguardo la tipologia di relazione che c'è tra le variabili.

Per superare questo problema ci si avvarrà di modelli più sofisticati, quali, *Regressione lineare multipla, Regressione polinomiale multipla, Albero decisionale, Foresta casuale* e una *rete neurale Feedforward*, ognuno dei quali analizzerà i dati in modo diverso. In questo modo, si potranno osservare e mettere a confronto modelli statistici come le regressioni con algoritmi di machine learning. Un metodo simile è stato utilizzato da Berlin Stanislav, Raz Tzvi, Moshe Zviran [15], tra i primi a mettere a confronto modelli di regressione con ANN, confrontando i risultati ottenuti. Per l'utilizzo di questi modelli, verrà suddiviso il

database per utilizzare il 75% circa dei dati come *Training Set* e il restante 25% come *Test Set*. In una prima fase si andrà, quindi, ad addestrare i modelli fornendo il Training Set in input in modo da apprendere le relazioni tra i features e gli output. Una volta che i modelli saranno addestrati si passerà alla fase di test in cui si provano i modelli sul dataset di test, che sono dati in input ignoti al momento dell'addestramento. Il confronto tra l'output vero e quello predetto dai vari consentirà la loro valutazione in base a diverse metriche. Ovviamente, bisogna precisare che più è vasto un set di dati di training più questo rappresenta meglio la realtà, quindi i risultati che si otterranno potrebbero cambiare con l'aumento delle dimensioni del database.

I parametri dei modelli che saranno analizzati per condurre un'analisi comparativa saranno:

- *l'Accuracy:* rappresenta la frazione delle previsioni che il modello ha ottenuto correttamente, indicando così la precisone del modello con un valore compreso tra 0 e 1

- *RMSE (Root Mean Square Error):* è un metodo standard usato anche da [15] per misurare l'imprecisione di un modello, rappresentando l'errore medio in una previsione di dati qualitativi. In altre parole, è la deviazione standard dei residui, che sono una misura della distanza dai punti della linea di regressione. È un valore che assume l'unità di misura del fenomeno che si sta analizzando.

Prima di passare al visone dei modelli, è bene fare le ultime considerazioni sui valori dei parametri che visioneremo, in modo da analizzarli e interpretarli al meglio. Un modello, per essere ritenuto valido e correttamente addestrato deve avere livelli di *Accuracy* simili per il training e il testing. Se il livello di *Accuracy* del testing fosse decisamente inferiore rispetto a quello del training significherebbe che l'algoritmo non è stato correttamente addestrato ma ha solo memorizzato modelli e rumore specifici del training set e che sia avvenuto un caso di *Overfitting.* Stesso ragionamento vale per l'indicatore RMSE.

4.4.1 Regressione lineare multipla

La regressione lineare è un metodo statistico di stima del valore atteso condizionato di una variabile dipendente Y, dati i valori di altre variabili indipendenti X (predittori). Nel caso di una variabile indipendente e un output, il risultato è una retta nel piano che interpola i punti costituiti dal dataset di addestramento. In altri termini, la regressione con più regressori consente, ovviamente se i dati sono disponibili, di misurare l'effetto di una specifica variabile x_i sulla variabile y, tenendo costanti le altre variabili indipendenti. Formalmente il modello di regressione lineare multipla include più regressori x_i ed associa a ciascun regressore un coefficiente β_i. Il coefficiente β_1, ad esempio, rappresenta la variazione attesa della variabile dipendente y associata ad una variazione unitaria di x_1, tenendo costanti gli altri regressori. Tutti i regressori si interpretano in questo modo. La formula della retta di regressione è così descritta:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_k X_{ki} + \epsilon$$

Dove ϵ rappresenta l'errore statistico associato al modello.

Nel nostro caso sono state identificate l'esperienza del gruppo, la complessità, il numero medio dei defects e la conoscenza del perimetro come variabili indipendenti.

I coefficienti β_i dell'equazione risultante dall'applicazione del modello al caso studio in esame sono i seguenti:

VARIABILE	COEFFICIENTE
Intercetta	657,765
Complessità	4.198,278
N. medio dei defects	1922,343
Exp. Gruppo	-4039,334
Conos. Perimetro	-1204,584

Figura 4.6 Coefficienti regressione lineare

I coefficienti associati che associano le variabili indipendenti con la variabile target mostrano un andamento in linea con le aspettative e con quanto analizzato precedentemente. L'aumento della complessità e dell'esperienza del gruppo impattano molto sulla variabile target Story Points anche se in senso contrario, invece, sembrano avere un impatto di misura inferiore le due variabili Numero medio dei defects e Conoscenza perimetro.

Per andare a misurare la bontà dell'adattamento del modello di regressione multipla, come fatto nel [15] si effettua il test t , che serve a valutare la significatività statistica di un predittore all'interno del modello. In questo caso, i p-value associato alla complessità e all'esperienza del gruppo sono minori di 0.05 e i p-value associati alle variabili numero medio dei defects e conoscenza del perimetro sono maggiori di 0.05. Ciò significa che la complessità e l'esperienza del gruppo spiegano una quota significativa della varianza di Y, mentre le altre due non sono statisticamente significative.

Di seguito viene invece riportata la tabella che mostra i valori di Accuracy e RMSE del training set e del testing set.

METRICA	RISULTATO
Training Accuracy	0,9237
Testing Accuracy	0,7489
Training RMSE	0.8160
Testing RMSE	0.8430

4.7 Parametri regressione lineare

I valori di Accuracy sembrano essere alti per il Training set, al contrario non sono molto elevati per il Testing set, segno di una non corretta corrispondenza tra i dati utilizzati per l'addestramento e quelli da testare e di una possibile presenza del fenomeno di overfitting. Per quanto riguarda i valori di Training RMSE e Testing RMSE risultano essere simili tra loro, indicando un buon addestramento, ma l'errore associato di circa 0,8 Story Points non risulta particolarmente basso.

4.4.2 Regressione polinomiale multipla

La regressione polinomiale multipla è una forma di regressione in cui la relazione tra la variabile indipendente x e la variabile dipendente y è modellata con un polinomio di grado n in x. Siamo in presenza di un modello che è lineare nei parametri, ma non lineare nella relazione tra le variabili dipendenti e quelle indipendenti. Per decidere il grado del polinomio si parte con un r massimo e si stima il modello di regressione polinomiale. Si può utilizzare il test t e se H_0 viene rigettata si procede con la stima di un'altra forma polinomiale di grado inferiore. Nel caso in esame il grado del polinomio è tre. La formula della curva di regressione è così descritta:

$$Y = \beta_0 + \beta_1 X + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_n X_n + \epsilon$$

Dove ϵ rappresenta l'errore statistico associato al modello.

Di seguito viene riportata la tabella che mostra i valori di Accuracy e RMSE del training set e del testing set.

METRICA	RISULTATO
Training Accuracy	0,8321
Testing Accuracy	0,8124
Training RMSE	0.0212
Testing RMSE	0.0091

4.8 Parametri regressione polinomiale

Dai dati ottenuti si può osservare come i valori di Accuracy e RMSE siano molto simili tra di loro nella fase di Training e nella fase di Testing, segno di una valida corrispondenza tra i dati utilizzati nelle due fasi. Inoltre, i valori di RMSE ottenuti sono molto buoni in quanto riportano un errore piuttosto basso. Nonostante ciò, dai livelli di Accuracy ottenuti, il modello non risulta essere molto preciso, in quanto un 83% circa di precisione nell'eseguire previsioni non risulta molto elevato.

4.4.3 Albero decisionale

Un albero decisionale è un algoritmo di apprendimento supervisionato, utilizzato sia per attività di classificazione che di regressione. Si compone di una struttura ad albero gerarchica, che consiste di un nodo radice, di rami, nodi interni e nodi foglia. Sono algoritmi che separano il dataset in più regioni e ad ogni input associano la corrispondente regione (o foglia) di appartenenza. Non richiedono operazioni di normalizzazione del dataset, inoltre, la presenza di *outliers* non impatta sul risultato. La loro rappresentazione è molto intuitiva ed è molto facile comprenderne i risultati. Il risultato è un grafico ad albero simile a un flow chart dove in ogni foglia sono contenuti i dati di una sola classe. L'algoritmo procede ad ogni passo separando i dati in base alla caratteristica che produce il miglior risultato e procede ricorsivamente fino alla classificazione di tutti i dati. Per l'applicazione del modello al caso studio in esame sono state impartite al modello regole decisionali semplici sotto forma di dichiarazioni se-allora-altro, dedotte dalle caratteristiche dei dati analizzati. Uno dei limiti di questo metodo è che gli alberi decisionali complessi tendono all'overfitting e non si generalizzano bene con i nuovi dati. Questo scenario può essere evitato attraverso i processi di pre-potatura o post-potatura. La pre-potatura interrompe la crescita degli alberi quando ci sono dati insufficienti mentre la post-potatura, dopo la creazione dell'albero, rimuove gli alberi secondari contenenti dati inadeguati. Per ovviare a questo problema è stata una profondità massima pari a 5.

Di seguito viene riportata la tabella che mostra i valori di Accuracy e RMSE del training set e del testing set.

METRICA	RISULTATO
Training Accuracy	0,9667
Testing Accuracy	0,9201
Training RMSE	0,2290
Testing RMSE	0,2431

4.9 Parametri albero decisionale

Dai risultati ottenuti è possibile vedere come il modello spieghi i dati ed effettui previsioni con un ottimo livello di precisione, di circa il 96% in fase di training e 92% circa in fase di testing. Inoltre, non si è verificato il temuto fenomeno dell'overfitting, ottenendo così risultati molto simili in fase di testing e di training. Anche i risultati del RMSE sono positivi, sia il training che il testing riportano errori accettabili e simili tra loro, a conferma che il modello in questione risulta tangibile e potenzialmente applicabile.

4.4.4 Foresta casuale

Quello della foresta casuale è un algoritmo che utilizza il metodo del *Bagging* basato sugli alberi decisionali. Il *Bagging* consiste nel campionare N volte il dataset iniziale e addestrare N volte lo stesso algoritmo, alla fine si aggregano i risultati facendo la media degli output in caso di regressione, la categoria con maggior frequenza nel caso di classificazione. È un metodo aumenta i dati di addestramento per diminuire la varianza nelle previsioni. Pertanto, il modello della foresta casuale è un modello complesso che costruisce più alberi su sottoinsiemi del dataset di addestramento selezionati casualmente, aggrega le previsioni di ciascun albero e sceglie la previsione migliore. Inoltre, la foresta casuale permette di valutare l'importanza delle caratteristiche definendo così una classifica degli input, dal più importante al meno importante nella definizione dell'output. Per il seguente caso di studio si è deciso di campionare 500 volte.

Di seguito viene riportata la tabella che mostra i valori di Accuracy e RMSE del training set e del testing set.

METRICA	RISULTATO
Training Accuracy	0,9572
Testing Accuracy	0,9478
Training RMSE	0,2304
Testing RMSE	0,2372

4.10 Parametri foresta casuale

Dai risultati ottenuti è possibile vedere come il modello effettua previsioni con un ottimo livello di precisione, ottenendo un livello di Accuracy del 95 % circa sia in fase di training sia in fase di testing. Anche i risultati del RMSE risultano ottimi, sia il training che il testing riportano errori bassi e simili tra loro, ottenendo un errore di circa 0,2 Story Points, a conferma della tangibilità metodologica e di un'eventuale applicazione al caso esaminato.

4.5 Feedforward neural network

Come già descritto nella parte iniziale dell'elaborato, la feedforward neural network è un particolare tipo di rete neurale, il cui funzionamento è abbastanza semplice. Basandosi su quando detta da [20] ''Una rete neurale artificiale feedforward con due strati di neuroni e una funzione di attivazione non costante e non decrescente in ciascun neurone nascosto può approssimare qualsiasi funzione continua a tratti da un sottoinsieme chiuso e limitato dello spazio N dimensionale euclideo." e da Hornik et al [21] secondo cui le architetture con uno strato nascosto possono approssimare qualsiasi funzione non lineare, per il caso di studio specifico è stata costruita una semplice struttura formata da 4 neuroni nell'input layer, ovvero le 4 variabili del caso di studio, un solo neurone nell'hidden layer e un solo neurone nell'output layer, ovvero la nostra variabile target story points. Di seguito viene riportata un'immagine della rete creata e il suo modello matematico.

4.11 Feedforward neural network utilizzata

Il cui modello matematico è:

$$A_1 = x1*w1 + x2*w2 + x3*w3 + x4*w4 + b$$

$$Y = F(A_1 w_1)$$

Di seguito viene riportata la tabella che mostra i valori di Accuracy e RMSE del training set e del testing set.

METRICA	RISULTATO
Training Accuracy	0,9663
Testing Accuracy	0,9528
Training RMSE	0,2401
Testing RMSE	0,2412

4.12 Parametri feedforward neurale network

Il modello effettua previsioni con un ottimo livello di precisione, ottenendo un livello di Accuracy del 96 % circa sia in fase di training sia in fase di testing. Anche i risultati del RMSE risultano ottimi, sia il training che il testing riportano errori bassi e simili tra loro, ottenendo un errore massimo di circa 0,2 Story Points. Per i risultati ottenuti, risulta essere un potenziale modello di applicazione per il caso in esame.

4.6Analisi dei risultati

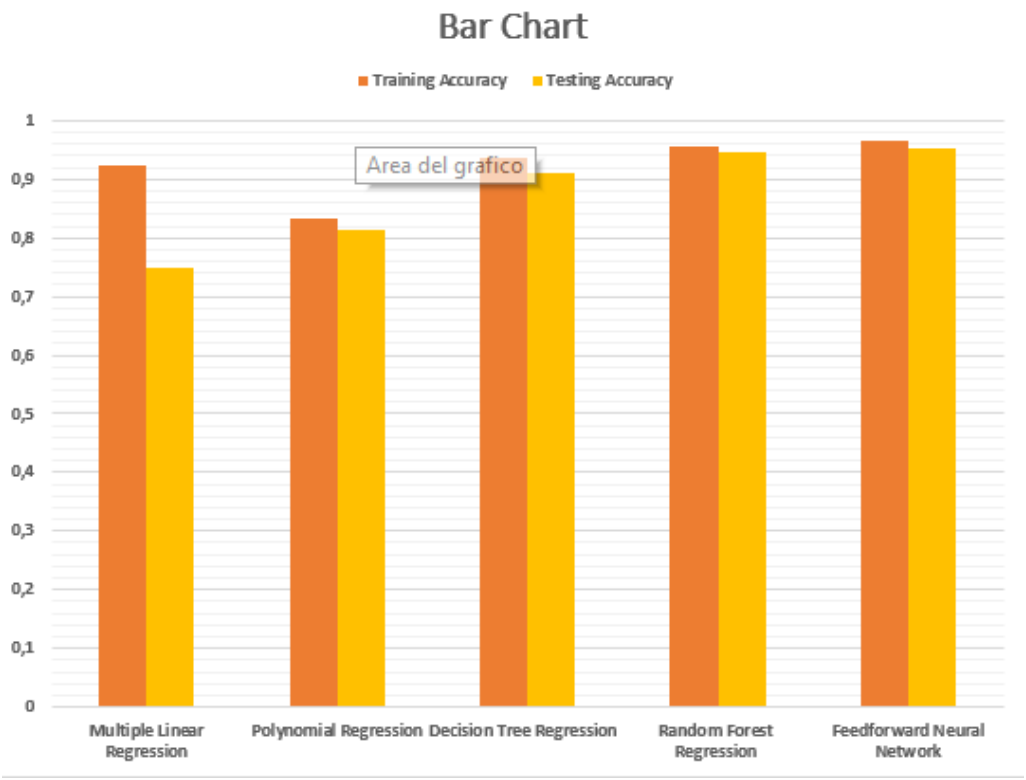
In questa parte dell'elaborato verrà presentato un riepilogo dei risultati ottenuti e sulla base di questi si faranno delle considerazioni su quello che risulta essere il modello più adatto per il caso di studio in esame. Come già anticipato precedentemente, i risultati ottenuti sono correlati al numero di dati a disposizione e potrebbero essere diversi al loro variare. Tuttavia, la procedura da seguire rimane la stessa.

Di seguito un riepilogo dei risultati ottenuti

	Training Accuracy	Testing Accuracy	Training RMSE	Testing RMSE
Multiple Linear Regression	0,9237	0,7489	0.8160	0.8430
Polynomial Regression	0,8321	0,8124	0.0212	0.0091
Decision Tree Regression	0,9667	0,9201	0,2290	0,2431
Random Forest Regression	0,9572	0,9478	0,2304	0,2372
Feedforward Neural Network	0,9663	0,9528	0,2401	0,2412

4.13 Riepilogo parametri dei vari modelli

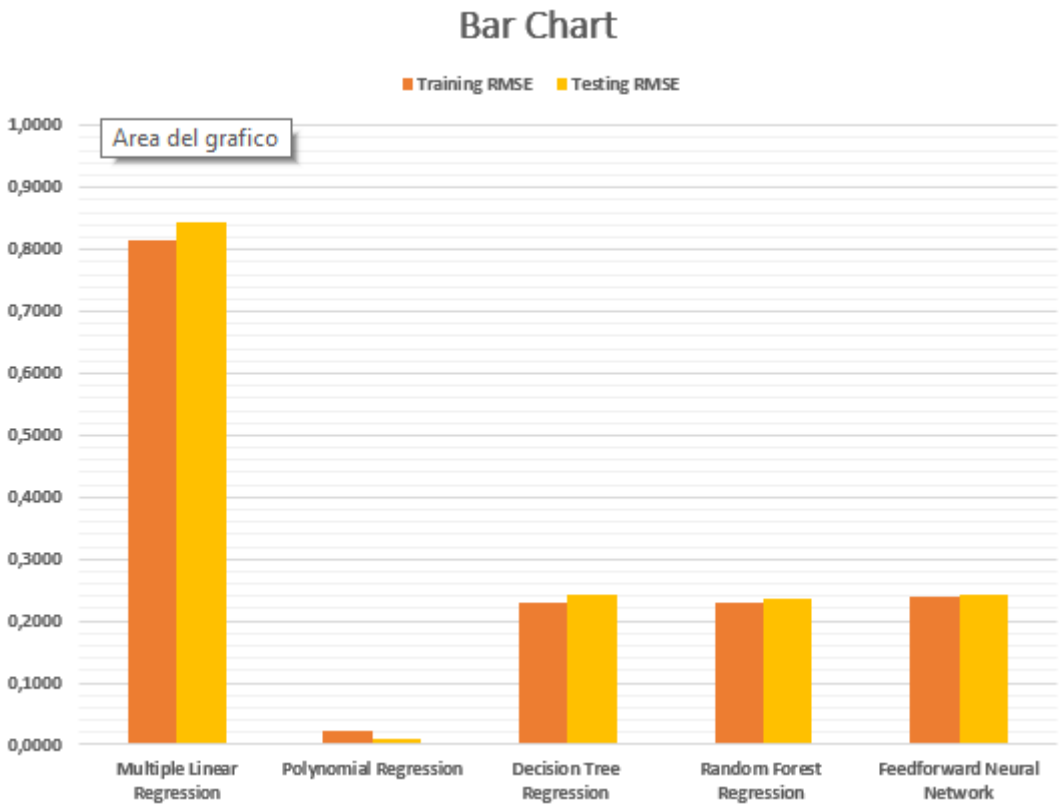
Visualizzazione grafica dell'Accuracy.



4.14 Training accuracy vs Testing accuracy

Dal grafico mostrato sopra, risulta come i modelli che subito appaiono più precisi sono l'albero decisionale, la foresta casuale e la feedforward neural network. Ognuno di essi mostra ottimi valori di Accuracy, superiori rispetto agli altri due modelli sia in fase di training si in fase di testing.

Visualizzazione grafica del RMSE.



4.15 Training RMSE vs Testing RMSE

Con quest'altro grafico, si osserva l'andamento del RMSE nei vari modelli. Si può osservare come il modello di regressione polinomiale abbia riportato i più bassi valori relativi all'errore di previsione, ma anche l'albero decisionale, la foresta casuale e la feedforward neural network hanno ottenuto valori sufficientemente bassi sia in fase di training che in quella di testing, confermando la loro tangibilità metodologica e una loro potenziale applicazione.

Le considerazioni effettuate vanno in linea con lo studio effettuato da *Berlin Stanislav, Raz Tzvi, Moshe Zviran,* [15] secondo cui le ANN e gli algoritmi, che non sono vincolati a funzioni lineari, possono trattare meglio le osservazioni che si trovano lontano dalla migliore linea retta. Si è deciso, pertanto, di scartare quelli che sono ritenuti i modelli meno adatti al caso di studio, ovvero, il modello di regressione lineare

multipla e il modello di regressione polinomiale multipla, e di concentrare l'analisi sui modelli rimanenti.

Di seguito viene riportata una simulazione dell'applicazione dei tre modelli e saranno messi a confronto i dati reali ottenuti e le previsioni da loro effettuate. Saranno riportarti i primi sette risultati ottenuti con l'applicazione dei tre modelli.

Risultati relativi all'applicazione dell'albero decisionale.

SP REALI	SP PREVISTI
3,50	3,27
7,00	7,19
4,00	4,33
1,50	1,55
1,50	1,42
2,50	2,41
5,00	5,21

4.16 Valori reali vs valori previsti albero decisionale

Risultati relativi all'applicazione della foresta casuale.

SP REALI	SP PREVISTI
3,50	3,31
7,00	7,17
4,00	4,10
1,50	1,52
1,50	1,56
2,50	2,57
5,00	4,96

4.17 Valori reali vs valori previsti foresta casuale

Risultati relativi all'applicazione della feedforward neural network.

SP REALI	SP PREVISTI
3,50	3,54
7,00	6,89
4,00	4,12
1,50	1,46
1,50	1,53
2,50	2,52
5,00	4,89

4.18 Valori reali vs valori previsti feedforward neural network

Tutti e tre i modelli giungono a previsioni sufficientemente accurate e precise dei story points. Le previsioni sembrano accurate sia per story points bassi sia per story points più alti, segno che i modelli hanno avuto un training adeguato riescono ad essere sufficientemente precisi sia con valori bassi di sp (più numerosi all'interno del database) si con valori più alti di sp.

Sulla base dei risultati ottenuti dalla simulazione e delle metriche viste, il modello che risulta essere più adatto al caso di studio per le sue performance è il feedforward neural network. È infatti il primo modello per valore di Accuracy in fase di testing e il secondo per errore sia in fase di training sia in fase di testing.

L'analisi dei risultati ottenuti conduce, pertanto, alla scelta del feedforward neural network, che sulla base del database analizzato e delle valutazioni effettuate, sembra essere il modello che si adatta meglio al caso di studio.

5 Conclusioni

Circa il 30% di tutti i progetti software fallisce. Ciò significa che la durata del progetto sta diventando troppo lunga o il budget non viene mantenuto. Una delle criticità che contribuisce a tale percentuale è Il processo di stima dei story points per analogia, in quanto è un processo che può portare ad imprecisioni dettate da errori di analisi. Può capitare che due progetti presentino analogie solo in linea teorica, e dimostrarsi, solo in un secondo momento molto differenti tra loro. Inoltre, il settore tecnologico subisce continue innovazioni; sebbene ci si basi spesso sulle analogie con i progetti precedenti, in fase iniziale di stima progettuale è importante considerare le possibili trasformazioni dei tools e delle tecnologie valutando l'effettiva affidabilità della comparazione.

La credibilità di una stima si basa molto sulla trasparenza e la comprensione del metodo utilizzato, sui dati utilizzati, e sulle assunzioni fatte. Per questo motivo l'implementazione del machine learning all'interno del processo di stime economiche e temporali attraverso l'elaborazione di modelli predittivi rappresenta, sicuramente, una novità per gli esperti del mestiere e che sta avendo sempre più casi di applicazione. Ovviamente, per effettuare una stima predittiva dei tempi e dei costi in uno stadio iniziale del progetto attraverso il machine learning, è necessario identificare i descrittori di progetto che possono essere utilizzati come predittori e le relazioni che intercorrono tra di essi. Inoltre, è importante impostare un lavoro di analisi a monte al fine di identificare la metodologia di stima di tempi e costi da utilizzare all'interno dell'aziendale di ridurre il più possibile gli errori legati alla previsione. Tuttavia, bisogna tenere conto del fatto che le caratteristiche del prodotto nelle fasi iniziali del progetto tendono a cambiare nelle fasi successive e in particolare nei progetti di sviluppo software.

Al netto delle considerazioni da effettuare, lo studio presentato ha voluto essere un piccolo esempio di utilizzo di metodi intelligenti e moderni per investigare e trattare temi estremamente attuali e centrali nell'ambito del Project Management. Nello specifico, si è tentato di utilizzare strumenti di intelligenza artificiale per la previsione della stima dei story points di un progetto IT, processo che ad oggi rimane una delle fasi più complesse all'interno di un progetto. Il motivo è che non esistono due progetti uguali tra loro: ognuno è unico per obiettivi e parametri che lo compongono.

L'auspicio è che il presente lavoro, possa rappresentare un aiuto concreto per l'azienda e il team in cui è nata l'idea, e congiuntamente ad ulteriori approfondimenti possa servire per migliorare e innovare il processo di elaborazione delle stime progettuali. I futuri studi che si potranno condurre e le continue evoluzioni che emergeranno, potrebbero mettere in luce nuove tecniche, fattori, relazioni in grado di ridurre gli errori legati alle stime dei tempi e dei costi e il perfezionamento dell'implementazione del machine learning per essere utilizzato come strumento predittivo.

Bibliografia

1. Cha, Gi-Woo, Hyeun Jun Moon, and Young-Chan Kim. "A hybrid machine-learning model for predicting the waste generation rate of building demolition projects." *Journal of Cleaner Production* 375 (2022): 134096.
2. Soltani, Ali, et al. "Housing price prediction incorporating spatio-temporal dependency into machine learning algorithms." *Cities* 131 (2022): 103941.
3. Goswami, Ruchi, Mansi Jasuja, and Saru Dhir. "Impact of different estimation approaches in traditional and agile development." *2016 Second International Conference on Computational Intelligence & Communication Technology (CICT)*. IEEE, 2016.
4. Owais, Mohd, and R. Ramakishore. "Effort, duration and cost estimation in agile software development." *2016 Ninth International Conference on Contemporary Computing (IC3)*. IEEE, 2016.
5. Popli, Rashmi, and Naresh Chauahn. "Managing uncertainty of story-points in agile software." *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2015.
6. Cohn, Mike. *Agile estimating and planning*. Pearson Education, 2005.
7. Choetkiertikul, Morakot, et al. "A deep learning model for estimating story points." *IEEE Transactions on Software Engineering* 45.7 (2018): 637-656.
8. Müller, Matthias. "Agile challenges and chances for open source: lessons learned from managing a FLOSS project." *2018 IEEE Conference on Open Systems (ICOS)*. IEEE, 2018.
9. Keshta, Nesma, and Yasser Morgan. "Comparison between traditional plan-based and agile software processes according to team size & project domain (A systematic literature review)." *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 2017.
10. Scott, Ezequiel, Khaled Nimr Charkie, and Dietmar Pfahl. "Productivity, turnover, and team stability of agile teams in open-source software projects." *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2020.
11. Chatziasimidis, Fragkiskos, and Ioannis Stamelos. "Data collection and analysis of GitHub repositories and users." *2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA)*. IEEE, 2015.
12. Chongpakdee, Panut, and Wiwat Vatanawood. "Estimating user story points using document fingerprints." *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, 2017.
13. Porru, Simone, et al. "Estimating story points from issue reports." *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering*. 2016.
14. Scott, Ezequiel, and Dietmar Pfahl. "Using developers' features to estimate story points." *Proceedings of the 2018 International Conference on Software and System Process*. 2018.
15. *Berlin Stanislav, Raz Tzvi, Moshe Zviran, 2008, Comparison of estimation methods of cost and duration in IT projects*

16. *Benala Rao Tirimula, Dehuri Satchidanada, 2013, Software Effort Prediction Using Learning (Clustering) and Functional Link Artificial Neural Networks.*
17. *Charted Institute of Building (CIOB), Appalti e performance del progetto. Occasional Paper N45, CIOB, Ascot, Inghilterra, 1980.*
18. *D. Baccarini, Il concetto di complessità progettuale: una rassegna, ISBG Journal of Gestione del progetto 14 (1996) 201–204.*
19. *A. James Anderson, Introduzione alle reti neurali, Massachusetts Institute of Technology, MA, 1995.*
20. *k. Hornik, MB Stinchcombe, H. White, Le reti feedforward multistrato sono approssimatori universali, Neural Networks 2 (5) (1989) 359–366.*
21. *K. Hornik, MB Stinchcombe, H. White, Le reti feedforward multistrato sono approssimatori universali, Neural Networks 2 (5) (1989) 359–366.*

Sitografia

- https://www.focusmgmt.it/knowledge/che-cose-il-machine-learning-e-possibili-applicazioni/ Accesso [30 ottobre 2022]
- https://universeit.blog/machine-learning/ Accesso [30 ottobre 2022]
- Supervised learning, cos'è, apprendimento supervisionato - AI4BusinessAccesso [5 novembre]
- Regressione lineare multipla: definizione ed assuzioni OLS - Mathsly Research
- 451899 (unina.it)
- Reti neurali, cosa sono? - Applicazioni, limiti ed evoluzione (intelligenzaartificiale.it)
- Function Points nei progetti software | Project Management Center (humanwareonline.com)
- Introduzione alle reti neurali | MATEpristem (unibocconi.it)