Università degli Studi di Milano
Facoltà di Scienze e Tecnologie
Corso di Laurea Magistrale in Informatica

DA UN MODELLO PREDITTIVO OPACO AD UN SURROGATO TRASPARENTE: APPLICAZIONE AI PROCESSI DI BUSINESS

Relatore: Prof. Gabriele Gianini

Correlatore:

Tesi di:
Piero Pastore
Matricola: 791511

Anno Accademico 2021-2022

Ciò che viene comunemente chiamato sacrificio, mia madre e mia nonna lo chiamano amore. Dedico questo traguardo a loro, pilastro della mia vita, fondamenta dei miei giorni.

Indice

Introduzione **1**

1 I processi di business **3**
1.1 Definizione . . . 3
1.2 Cos'è e come funziona . . . 4
1.3 Ruolo e attori . . . 5
1.4 Tipologie di processi e classificazioni . . . 6
1.5 Mappatura e misurazioni . . . 10
1.6 Ottimizzazioni . . . 11

2 Modelli neurali opachi **13**
2.1 Funzionamento . . . 14
2.2 Tipologie . . . 15
2.2.1 Reti Feedforward . . . 15
2.2.2 Reti ricorrenti (RNN) . . . 17
2.3 Modello LSTM . . . 18
2.3.1 La logica dietro LSTM . . . 18
2.3.2 Vantaggi e motivazioni . . . 20

3 Modelli trasparenti **21**
3.1 L'interpretabilita' . . . 21
3.2 XAI: Explainable Artificial Intelligence . . . 23
3.3 Albero decisionale . . . 24
3.4 DTR: Decision Tree Regressor . . . 26
3.4.1 Logica dietro il DTR . . . 27
3.4.2 Vantaggi e motivazioni . . . 29

4 Progettazione **30**
4.1 Obiettivo . . . 31
4.2 Strumenti utilizzati . . . 32
4.3 Procedimento . . . 33

4.3.1 Analisi e pulizia dataset 33
4.3.2 Fase di addestramento e ottimizzazione 35
4.3.3 Predizione 38
4.3.4 Confronto e analisi 40

Conclusioni **43**

Introduzione

La XAI (Explainable Artificial Intelligence) indica un ambito di ricerca che si occupa di capire l'interpretazione dei risultati da parte di modelli neurali opachi, cercando di analizzare le sue caratteristiche e la sua comprensibilità'.
Si avvale dell'utilizzo di modelli e metodi che consentono all'utente di fidarsi e comprendere al meglio i risultati ottenuti da un algoritmo di apprendimento automatico. In questo modo, un organizzazione crea fiducia e sicurezza quando mette in produzione modelli di intelligenza artificiale, adottando un approcio responsabile al suo sviluppo.
Sulla base di questo studio e' nata l'idea di un analisi tra un modello neurale opaco ed un modello surrogato semplice, entrambi applicati ad un dataset di processi di business. Lo scopo della tesi era quello di verificare se, un modello neurale opaco, potesse potenzialmente essere approssimato attraverso l'uso di un surrogato semplice tramite lo studio e le analisi delle predizioni ottenute.

L'esposizione del lavoro è articolata come segue:

Il capitolo iniziale introduce il lettore nei processi di business, analizzando cosa sono e come funzionano, il ruolo e gli attori che la compongono, le diverse tipologie e classificazioni e infine come vengono misurati e ottimizzati.

Il secondo capitolo descrive in linea di principio i modelli neurali opachi, il loro funzionamento e alcune tipologie, il capitolo si conclude andando ad esaminare il modello neurale implementato per il nostro progetto.

Il capitolo tre analizza l'interpretabilita' dei modelli neurali opachi andando ad introdurre in seguito i modelli trasparenti, il capitolo termina introducendo il modello surrogato scelto per la nostra analisi.

L'ultimo capitolo descrive in modo pratico il progetto svolto, andando poi ad analizzare i i risultati ottenuti.

Infine, nelle conclusioni, vengono riassunti i risultati ottenuti e vengono proposti possibili sviluppi futuri.

Capitolo 1

I processi di business

1.1 Definizione

Si puo' definire un processo di business un insieme di attività' svolte con lo scopo di trasformare input in output. Il processo di business quindi e' costituito da un insieme di attivita' interconnesse tra di loro e strutturate in modo tale da generare un servizio o un prodotto specifico richiesto per il cliente finale.
L'output generato dunque rappresenta l'obiettivo finale delle attività' che costituiscono il processo di business.
"I business process devono quindi cercare di dare valore al cliente limitando al massimo le attivita' che non portano valore"

1.2 Cos'è e come funziona

Nei contesti aziendali, i processi di business rappresentano un elemento vitale ai fini della crescita aziendale, in quanto attraverso ciò' l'azienda aumenta il valore delle risorse trasformandole in prodotti finali utili a migliorare la soddisfazioni dei clienti. In genere, i processi di business vengono rappresentati attraverso un diagramma di flusso come una sequenza di attività' legate da snodi decisionali oppure tramite una matrice di processo formata da attività' da ordinare e regole basate sui dati del processo.
Possiamo suddividere i processi di business in base alla loro importanza e utilita' in tre categorie:

- **Processi operativi**: che rappresentano il core business e si occupano di creare il flusso del valore primario, come ad esempio acquisiti, produzione, marketing.
- **Processi di gestione**: permettono di regolare il funzionamento di un sistema, come ad esempio la 'gestione strategica'
- **Processi di supporto**: vengono utilizzati per sostenere il core business, come ad esempio contabilità', supporto tecnico,

1.3 Ruolo e attori

Un business process e' un sistema fondamentale e vantaggioso all'interno di un azienda il quale permette di completare un progetto e raggiungere l'obiettivo attraverso una stabilita pianificazione.
Inoltre ogni singolo processo all'interno della pianificazione crea un valore, dato che si otterrà' un prodotto o un servizio necessario all' attività' d'impresa, oppure un bene che sarà' a disposizione dell' azienda stessa o di un cliente esterno.
All'interno di un business process esistono diverse figure con ruoli attivi e distinti tra le singole fasi dei processi:

- **Responsabile del processo**: colui che si occupa della programmazione, supervisionando tutte le figure che fanno parte integrante dell'attivita' e le fasi per il raggiungimento dell'obiettivo, e' la figura professionale che all'interno di un azienda segue il processo e ha totale responsabilità' dei vari processi

- **Responsabile operativo**: colui che si occupa di eseguire un azione diretta di controllo e di gestione del processo aziendale, intervendo in caso di ritardo o di attriti tra i vari reparti

- **Process worker**: figura che agisce direttamente all'interno del processo, operando al fine di realizzare le singole fasi, non segue il processo aziendale, ma svolge un importante attivita'di routine che e' parte integrante dell'intero processo.

1.4 Tipologie di processi e classificazioni

Ci sono tre tipi di processi di business:

- **i processi di gestione**: che regolano il funzionamento di un sistema.
  Sono tipici esempi di processi di gestione la "gestione strategica" e la "corporate governance"

- **i processi operativi**: che costituiscono il core business e si occupano di creare il flusso del valore primario. Acquisti, produzione, marketing e vendite sono inclusi in questa categoria

- **i processi di supporto**: che hanno l'obiettivo di sostenere il core business. Gli esempi relativi a questi processi includono contabilità, supporto tecnico e customer service

Un processo di business nasce a partire da un obiettivo aziendale e si conclude con il raggiungimento dell'obiettivo stesso. Le organizzazioni orientate al processo abbattono le barriere rappresentate dal concetto di reparto aziendale ed evitano i rischi derivanti dalla mancanza di correlazione funzionale al loro interno.
Naturalmente ogni processo di business può essere scomposto in diversi sotto-processi dotati ciascuno di caratteristiche proprie ed ognuno dei quali contribuisce al raggiungimento dell'obiettivo del processo principale.
Di conseguenza in genere l'analisi dei processi aziendali comprende la mappatura di processi e sotto-processi fino ad arrivare alla descrizione dettagliata delle singole attività. I processi di business devono avere l'obiettivo di dare valore al cliente limitando al massimo le attività che non portano valore.

Parliamo di un processo di business ben progettato nel momento che:

- **efficace**: è in grado di fornire un elevato valore al cliente
- **efficiente**: è in grado di fornire il valore con ridotti costi per la società

Le proprietà' fondamentali per cui un processo di business e' caratterizzato sono:

- esistenza di clienti interni od esterni all'organizzazione di cui il processo fa parte
- esistenza di una persona responsabile del processo che abbia un livello sufficiente di autorità e di competenza
- ripetibilità del processo intesa come ripetibilità delle attività correlate che formano il processo stesso
- esistenza di una condizione che inneschi il processo stesso (trigger condition)
- misurabilità del processo ed in particolare delle principali variabili che ne determinano la performance (KPI)
- indipendenza strutturale del processo

Se proviamo ad analizzare un processo di business notiamo come si possano individuare due criteri distinti per determinare una classificazione, il primo criterio si basa sul destinatario del processo e relativamente a cio' si possono individuare:

- **processi rivolti a clienti esterni**: Gli output di tali processi sono direttamente rivolti al mercato esterno all'azienda
- **processi rivolti a clienti interni**: Questi ultimi sono un'insieme di attività interconnesse tra loro che hanno lo scopo di creare outputs richiesti da altri dipartimenti interni all'azienda di appartenenza o dai dirigenti dell'organizzazione al fine di raggiungere gli obiettivi della stessa e rendere più efficiente il sistema di creazione del valore

Il secondo criterio si riferisce invece alla tipologia del processo basato sulle caratteristiche costitutive dell'attività' che lo compongono ed in particolare si possono distinguere:

- **processi base**: processi basati su una richiesta di un cliente esterno, sono processi principali che caratterizzano ciascuna organizzazione e ne definiscono il core business
- **processi manageriali**: processi fondamentali a definire gli obiettivi e a supportare organizzando e gestendo i processi di base per ottenere gli obiettivi che l'organizzazione si propone. Tali processi, come i processi di supporto sono definiti anche processi abilitanti, in quanto assicurano la realizzazione dei processi base di un'organizzazione.
- **processi di supporto**: Tali processi sono collaterali ai processi di base e ne garantiscono il corretto funzionamento. I processi di supporto, definiti anche

processi abilitanti, anche se non creano direttamente valore sono necessari per assicurare la realizzazione dei processi di core business. Fanno parte di questa categoria i processi di rifornimento degli stock, la manutenzione di strutture e macchinari e la pulizia di ambienti e beni

1.5 Mappatura e misurazioni

Dopo aver appresso cosa sono e le tipologie di un processo di business, e' importante comprendere come mappare un processo per ottimizzare le fasi di analisi, aiutare l'analista a comprendere un processo avendo un analisi completa, relativa all'interno percorso da seguire in modo tale da riuscire a consentire l'individuazione di criticità' di singole sotto fasi del processo.

Altri vantaggi molto utili possono essere la facilita' di comprensione durante riunioni aziendali, la facile individuazione di ruoli e mansioni e del corretto carico di lavoro per ogni individuo.

La mappatura viene eseguita tramite diagrammi di flusso, il quale prevede una rappresentazione temporale delle attività' di un processo attraverso una raffigurazione grafica oppure utilizzando colori e rappresentazioni differenti a seconda del tipo di attività'.

Le misurazioni di un processo di business vengono eseguite attraverso indicatori di performance o sistemi di rilevazione, essi permettono di misurare le prestazioni aziendali riguardo un singolo processo e questi indicatori devono essere di semplice interpretabilità', avere una cadenza breve ed essere analizzati da specifiche figure in modo tale da evitare che l'interpretazione dei dati possa generare conflitti.

Una figura preposta attraverso queste misurazioni ha la possibilità' di misurare gli scostamenti tra le attività', verificare la distanza tra l'obiettivo atteso e i risultati ottenuti e permettere di pianificare e programmare le attività' aziendali permettendo correzioni in corso d'opera.

1.6 Ottimizzazioni

Se la mappatura permette un analisi in modo semplice, l'ottimizzazione permette di chiudere al meglio l'analisi effettuata in modo tale da porre rimedio a problematiche evidenziate nella fase precedente. Attraverso l'ottimizzazione dei processi di business un azienda aumenta la sua crescita e produttività', aumentando l'efficienza organizzativa e migliorando i processi.
Ottimizzando i processi conseguentemente vengono ottimizzati gli obiettivi aziendali, altri vantaggi che possono aiutare le organizzazioni sono:

- ottimizzazione delle operazioni
- riduzione dei rischi
- ottimizzazione nell'uso delle risorse
- consistenza
- garanzia di qualità'

Per eseguire una efficiente ottimizzazione e' ottenere da essa il massimo profitto e' opportuno svolgere una pianificazione corretta, attraverso diverse fasi:

- **Pianificazione**: scegliere il processo problematico che si desidera ottimizzare e quindi definire lo scopo e gli obiettivi
- **Analisi**: analizzare il processo e verificare se sta raggiungendo gli obiettivi desiderati analizzando inoltre se esiste uno spreco in eccesso da ridurre
- **Implementazione** Dopo aver eliminato gli elementi conflittuali, si automatizza il processo rivisto nella sua nuova forma
- **Monitoraggio** tenere monitorati i fattori sulle prestazioni del processo e ottimizzarli fino ad ottenere gli obiettivi prefissati

Per eseguire una corretta ottimizzazione dei processi di business possono esserci molte modalità' distinte, esso dipende dal processo in questione, non esiste una soluzione adatta per ogni scenario, nella maggior parte dei casi tuttavia, l'ottimizzazione avviene attraverso uno dei seguenti metodi:

- **Ristrutturazione** viene identificato un processo o passaggio che rappresenta uno spreco o un efficienza, una volta individuati i processi e' sufficiente migliorarli per raggiungere l'efficienza.
- **Automazione** l'automazione di processi di business permette di migliorare il carico di lavoro dei dipendenti, il che porta a un aumento della produttività e ad un incremento del morale.

Capitolo 2

Modelli neurali opachi

Una rete neurale e' una serie di algoritmi che tenta di riconoscere le relazioni sottostanti in un insieme di dati attraverso un processo che imita il modo in cui opera il cervello umano. In questo senso, le reti neurali si riferiscono a sistemi di neuroni, di natura artificiale che interpretano i dati sensoriali attraverso un raggruppamento di input grezzi, contenuti in vettori, che possono essere immagini, suoni, testo o serie temporali.

Le reti neurali possono adattarsi al cambiamento di input, quindi la rete genera il miglior risultato possibile senza la necessita' di riprogettare i criteri di output. Il concetto di reti neurali, che nasce dall'intelligenza artificiale, sta rapidamente guadagnando popolarità' soprattutto all'interno della comunità di analisi dei dati.

2.1 Funzionamento

I modelli neurali si basano sul concetto di neurone, il quale indica un corpo cellulare che riceve un segnale sotto forma di un impulso elettrico mediante neuro trasmettitori. Dopo aver ricevuto il segnale, esso viene elaborato e se supera una certa soglia, viene prodotto un output. In modo analogo, un neurone artificiale, riceve dei segnali tramite un vettore di input numerici moltiplicato scalarmene con un vettore di pesi.
Uno di questi pesi nello specifico viene chiamato bias ed e' connesso ad un input fittizio di valore 1, viene eseguita una somma pesata ed applicata una specifica funzione di attivazione producendo un singolo e unico output.

$$\text{y} = \text{activation}(\sum_{i=1}^{N} w_i x_i + b)$$

- **y**: output
- **activation**: funzione di attivazione
- **N**: numero totale di input
- **b**: bias
- **w** : input i-esimo
- **x** : peso i-esimo

Un neurone quindi apprende informazioni in base al ribilanciamento dei suoi vettori pesi in modo tale da riuscire ad approssimare al meglio la funzione desiderata. Andando poi a collegare altri neuroni tra di loro la rete neurale diventa un grafo formato da dei vertici che sono i neuroni e da archi chiamati connessioni. Ogni connessione e' caratterizzata da un peso che indica la rilevanza dell'output di quel neurone per il neurone che segue, in questo modo riusciamo a scalare l'output ad un intervallo di uscita desiderato.

2.2 Tipologie

Nonostante il fatto che il numero di tipologie e' in forte crescita col passare del tempo, tendenzialmente esse possono essere suddivide in due categorie principali.

2.2.1 Reti Feedforward

Le reti neurali feed-forward sono un tipo di rete neurale nel quale il grafo che la caratterizza è aciclico, il calcolo procede dai neuroni di input verso i neuroni di output progressivamente seguendo l'ordine topologico dei neuroni nella rete. I neuroni quindi propagano il loro output ai neuroni connessi e si propagano fino al successivo neurone, fino a che l'output esterno viene generato.
Questo non significa che la rete possiede una struttura regolare, infatti i neuroni possono essere connessi in qualsiasi modo tra di loro con l'unico vincolo il fatto che il grafo che le descrive sia aciclico, quindi una sola direzione dei dati attraverso la rete. Se invece nella rete e' presente un qualsiasi tipo di ciclo (su un solo neurone o gruppi di neuroni) la rete non e' piu' feed forward ma ricorrente.

Uno degli aspetti più' interessanti delle reti neurali e' la possibilità' di allenarle con l'ausilio di dati di esempio.
La fase di learning avviene aggiornando i pesi delle connessioni e altri possibili parametri, come il bias, per approssimare al meglio una possibile funzione non nota, o più' generalmente risolvere un problema oggettivo. Ci sono due tipologie di apprendimento:

- **Supervisionato** : in questa tipologia e' richiesto un dataset il quale possiede per ogni input, l'output desiderato, in modo tale da riuscire a far apprendere alla rete neurale e generalizzare un algoritmo per classificare ogni input, cioè' riuscire a far riconoscere quelle piccole variazioni non note degli input per classificarle nel modo corretto.

  Per determinare il livello di correttezza di una rete neurale per un apprendimento supervisionato, si usa una funzione di errore per misurare quanto l'output prodotto coincide con quello corretto. La funzione di errore e' definita dalla somma della differenza al quadrato del output desiderato e quello attuale.

- **Non supervisionato**: in questa tipologia invece, non si conoscono gli output corretti e si ha a disposizione un set di training pattern.

  L'output viene scelto direttamente durante il training. L'output che viene generato riconduce ad un clustering in modo tale che i vettori di input simili producano lo stesso output nello stesso cluster.

2.2.2 Reti ricorrenti (RNN)

Una rete neurale ricorrente e' una rete neurale che possiede dei cicli al suo interno tra i neuroni. I cicli possono coinvolgere o un singolo neurone o un gruppo di neuroni. Gli output in uscita di determinati neuroni vengono usati come input per neuroni precedenti, questa interconnessione tra neuroni di posizioni differenti permette l'utilizzo di alcuni neoroni come memoria di stato e consente, inviando in ingresso una sequenza temporale di valori diversi, di avere un comportamento dinamico nel tempo attraverso le informazioni ricevute negli istanti di tempo precedenti.

L'output viene generato quando viene raggiunta una stabilita'. Questa tipologia di reti sono adatte per la rappresentazione di equazioni differenziali e per risolverle (approssimativamente) in modo numerico. Se il tipo di equazione differenziale è noto che descrive un dato sistema, ma i valori dei parametri che compaiono in esso sono sconosciuti, si può anche provare a addestrare una rete ricorrente adatta con l'aiuto di modelli di esempio al fine di determinare i parametri del sistema.

Considerando che i cicli si propagano nel tempo, non e' possibile utilizzare una funzione di errore per misurare l'output prodotto in quanto ad ogni iterazione del ciclo vengono propagati gli errori, una rete ricorrente quindi deve essere dispiegata nel tempo tra due pattern di training. La ricalibrazione dei pesi viene calcolata tramite la rete spiegata e questa particolare forma viene chiamata error backpropagation through time [2].

Le ricalibrazioni dello stesso peso vengono combinate per generare il valore della rete ricorrente. Questa particolare tipologia di rete neurale viene adottata specialmente nel riconoscimento vocale o della grafia.

2.3 Modello LSTM

Le reti LSTM ('Long Short-term memory' ovvero una memoria a lungo-breve termine), sono una tipologia speciale di reti opache ricorrenti nate da Sepp Hochreiter e Jurgen Schmidhuber nel 1997 [1] e vengono utilizzate nel campo del Deep Learning. Si tratta di una varietà' di reti neurali ricorrenti (RNN) in grado di apprendere dipendenze a lungo termine, specialmente nei problemi di previsione delle sequenze, viene utilizzata frequentemente nel riconoscimento vocale o nella traduzione automatica ma si tratta di un modello che mostra prestazioni molto efficienti su un ampia varietà' di problemi.

2.3.1 La logica dietro LSTM

La particolarita' delle reti neurali LSTM e' il fatto che risolvono il problema piu' serio delle RNN, cioe' la scomparsa del gradiente [3], le celle LSTM permettono di elaborare i dati in modo sequenziale e mantenere il loro stato nascosto al trascorrere del tempo. Questo e' possibile grazie ad un fattore chiave delle LSTM, la cella di memoria, la quale ad ogni iterazione mantiene l'output in uscita al di fuori del normale flusso della rete.

Ogni cella di una rete LSTM e' divisa in due vettori che sono rispettivamente:

- **STM**: short term memory, una memoria a breve termine che contiene l'output della cella
- **LTM**: long term memory, una memoria a lungo termine che mantiene le informazioni utili nelle iterazioni successive

Ogni output di una rete LSTM e' formato dalla somma tra l'output precedente e l'input attuale, i quali vengono elaborati tramite gate di varie tipologie:

- **learn gate**: permette di determinare quali dati in output bisogna mantenere nella cella di memoria.
- **forget gate**: e' il gate utilizzato per rimuovere le informazioni che non hanno bisogno di essere mantenute a lungo termine.
- **remember gate**: permette di calcolare la nuova memoria a lungo termine andando a recuperare le informazioni del forget gate e del learn gate
- **output gate**: il suo compito e' quello di filtrare le informazioni in uscita.

Nella figura 1 viene evidenziata la struttura di una rete neurale LSTM, x indica il dato in input, h e c rappresentano rispettivamente la memoria a breve e lungo termine, infine le operazioni "+" e "x" indicano prodotti e somme elemento per elemento.

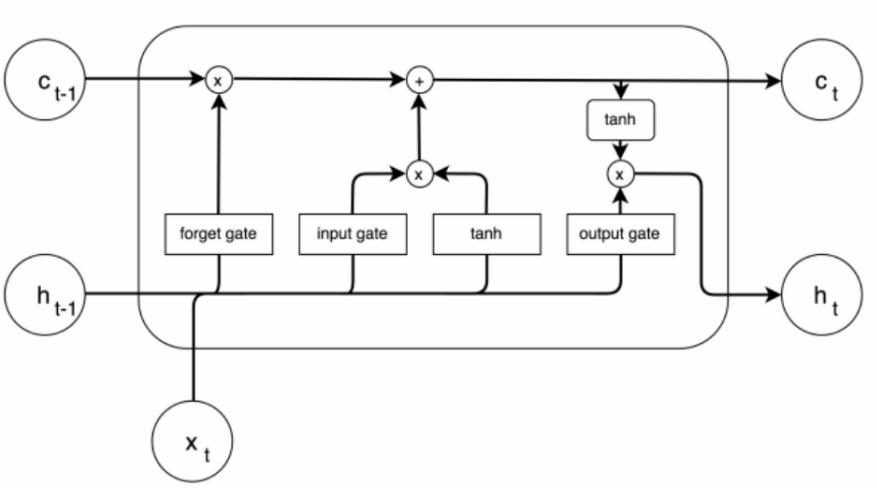

Figura 1 - Struttura di una rete LSTM

2.3.2 Vantaggi e motivazioni

Nell'ambiente odierno, la previsione di dati su larga scala e' molto complessa e i dati necessari per una previsione precisa e accurata su scala non sono sempre semplici e di facile reperibilita'. Utlizzando le reti neurali LSTM, i modelli di previsione di serie temporali generati possono prevedere i valori futuri tramite una base di dati di input che puo' essere formata da variabili categoriche o numeriche.

Tramite questo modello, si ha la possibilità' di prevedere i dati futuri contando su una maggiore precisione, il quale si traduce in un miglior processo decisionale per l'organo che utilizza questa tecnologia.

Capitolo 3

Modelli trasparenti

3.1 L'interpretabilita'

Uno degli aspetti piu' importanti riguardo i modelli neurali e' la loro interpretazione. Piu' una rete neurale e' profonda e complessa meno sara' una sua possibile interpretazione.

Anche se questo tipo di reti neurali ottengono numerosi risultati eccellenti in molteplici ambiti come la modellazione del linguaggio o la visione artificiale, esse non possono essere utilizzate in particolare quando si trattano argomenti umani, giurici o di etica, come applicazioni legali, finanziarie o mediche, dove si deve verificare con mano i risultati o l'attinenza di un modello.

Le reti neurali dunque sono ambigue e trasversali e vengono definite:

- **black box** le quali contengono strutture intricate di apprendimento automatico da parte dell'intelligenza artificiale, arrivando al punto di non conoscere neanche quale variabile o parametro passato in input abbia avuto un ruolo rilevante nel risultato generato.

  Pertanto, le reti neurali profonde riescono a produrre e generare un risultato con grande accuratezza ma risulta estremamente difficile riuscire a dare un senso ai milioni di neuroni che vengono sollecitati durante la fase di apprendimento. Un esempio e' la rete neurale LSTM analizzata in precedenza.

- **white box** si tratta della definizione delle reti neurali trasparenti, esse permettono la loro interpretazione in modo chiaro e univoco, riuscendo a stabilire una relazione lineare tra input e output .

  Esempi di modelli trasparenti, sono gli alberi decisionali che vedremo in seguito.

.

3.2 XAI: Explainable Artificial Intelligence

A seguito delle necessità' riscontrate nel capitolo precedente, nasce la XAI.
Si tratta di quell'ambito di ricerca il quale si occupa di capire le relazioni che ci sono tra i dati in ingresso di un modello e i risultati ottenuti, soprattutto nei modelli più' opachi, dove i numerosi parametri e classificatori non consentono una corretta interpretazione da parte dell'uomo.
I pilastri su cui si basa questo settore sono:

- **Conoscenza**: caratteristiche di un modello che permettano la comprensione e il funzionamento
- **Comprensibilità**: caratteristiche di un modello che comprendono i risultati ottenuti durante il training
- **Interpretabilità**: la capacità' di spiegazione e chiarimento da parte di un essere umano
- **Spiegabilità**: il risultato del modello deve essere comprensibile e spiegabile da parte dell'utente
- **Trasparenza**: un modello di semplice comprensione

Se andiamo ad analizzare i concetti chiave su cui si basa la XAI, notiamo come il concetto piu' essenziale sia la comprensibilita' che lega quasi tutti i pilastri, che possiamo suddividere in comprensibilita' del modello e comprensibilita' umana.

3.3 Albero decisionale

L'albero decisionale e' un modello predittivo trasparente che puo' essere usato sia in casi di classificazione sia in casi regressione. Sono stati proposti per la prima volta da Leo Breiman [4] all'universita' di California. L'idea di base e' quella di rappresentare i dati in ingresso come un albero dove ogni nodo interno denota un test su un attributo, ogni ramo rappresenta il risultato della condizione e la foglia (nodo finale non interno) contiene una classe di appartenenza.
Ai giorni d'oggi gli alberi decisonali sono ampiamente utilizzati nei modelli di predizione, nelle classificazioni o regressioni. Si tratta di modelli supervisionati formati da diverse regole decisionali su ogni nodo dell'albero, per regola decisionale si intende qualsiasi condizione su uno o più' attributi del dato in input.
Possiamo classificare gli alberi decisionali in due categorie in base alla variabili su cui vengono applicati:

- **Categoriche**: ovvero analizzando i dati in input si va a classificare un determinato dato tramite variabili categoriche, un esempio puo' essere andare a classificare se il prezzo di una macchina e' basso, medio o alto.

- **Continue**: in questo caso le variabili che vengono analizzate dall'albero andranno a prevedere un output continuo (ad esempio il prezzo di una macchina)

Gli alberi decisionali vengono ampiamente utilizzati nella branca del machine learning, in particolare in diverse aree della classificazione e della modellazione di regressione. Grazie alla sua capacita' di rappresentare in modo trasparente l'output, e' possibile trarre semplicemente informazioni dal flusso del processo di modellazione, infatti sono i primi modelli su cui la XAI si sta basando per cercare di approssimare i risultati di

modelli neurali piu' profondi.
Alcuni esempi dove gli alberi decisonali vengono applicati maggiormente sono:

- Gestione aziendale
- Gestione della sanita'
- Diagnosi di malfunzionamenti
- Consumo di energia

Avendo analizzato cos'è' e come funziona un albero decisionale, passiamo ora ad analizzare quali sono i vantaggi e svantaggi di utilizzare questo modello.
Vantaggi

- Facile interpretazione
- Non richiede una grande mole di dati per essere utilizzato
- Ottiene buone performance indifferentemente dal numero di dati del dataset
- Non ha bisogno di un determinato tipo di dati, lavora bene sia su dati numerici sia su dati categorici

Svantaggi

- Overfitting
- Fare una piccola variazione durante la fase di addestramento puo' portare ad un notevole cambiamento nella predizione, non e' un modello robusto

3.4 DTR: Decision Tree Regressor

L'albero decisionale di regressione puo' essere considerato una variante dell'albero decisionale ed e' stato sviluppato per stimare funzioni a valori reali. Sono progettati per variabili dipendenti che accettano valori discreti, continui o ordinati, dove la somma della differenza al quadrato dei valori previsti e osservati viene utilizzata per misurare l'errore di previsione.

I dataset per il funzionamento degli alberi di regressione sono costituiti da variabili di risposta (input) e variabili predittive (output) e per quest'ultime si tratta generalmente di variabili numeriche. In generale, la metodologia impiegata nella costruzione degli alberi di regressione consente alle variabili di input di essere una combinazione di variabili continue e categoriali.

Ogni volta che ogni nodo decisionale nell'albero di regressione contiene una condizione sui valori di alcune variabili in input, viene sviluppato un albero decisionale e il nodo terminale dell'albero contiene i valori della variabile di output prevista.

Viene introdotto il processo di costruzione di un albero di regressione come applicazione di un metodo noto come partizionamento ricorsivo binario.

Prima di applicare questo processo, dividiamo il set di dati in addestramento e set di test. Il modello viene sviluppato e addestrato utilizzando il set di addestramento, in particolare il partizionamento ricorsivo binario prevede la separazione dell'insieme di dati in sottoinsiemi in base al valore di una caratteristica di input, il quale poi viene ripetuto nel sottoinsieme ottenuto ricorsivamente.

Il set di dati di test invece viene utilizzato per testare il modello e per visualizzarne l'accuratezza nella previsione.

3.4.1 Logica dietro il DTR

Un DTR puo' essere visto come un modello additivo nella forma,

$$\mathrm{m(x)} = (\sum_{i=1}^{l} k_i \cdot I(x \in D_i))$$

dove k sono constanti, I e' una funzione che restituisce 1 se il suo argomento e' vero, 0 altrimenti e D sono partizioni disgiunte dei dati di addrestramento D del modello tali che:

$$\bigcup_{i=1}^{l} D_i = D \qquad e \qquad \bigcap_{i=1}^{l} D_i = \emptyset$$

DTR e' costruito introducendo un algoritmo ricorsivo di partizionamento, questo algoritmo costruisce un albero dividendo ricorsivamente il campione di addestramento in sottoinsiemi più piccoli. L'algoritmo riceve in ingresso una set N di dati e se non viene soddisfatto un criterio di terminazione dell'albero, l'algoritmo genera un nodo di test t, i cui rami sono ottenuti applicando delle considizioni ai dati in ingresso. Questi sottoinsiemi sono costituiti dai casi in cui la condizione e' soddisfatta o meno il criterio ricevuti. Ad ogni nodo, viene scelto il miglior test in base ad un criterio locale.

```
Algorithm 3.1 - Recursive Partitioning Algorithm.

Input  : A set of n data points, { < x_i , y_i > }, i = 1,...,n
Output : A regression tree

IF termination criterion THEN
       Create Leaf Node and assign it a Constant Value
       Return Leaf Node
ELSE
       Find Best Splitting Test s*
       Create Node t with s*
       Left_branch(t) = RecursivePartitioningAlgorithm({ <x_i , y_i> : x_i → s* })
       Right_branch(t) = RecursivePartitioningAlgorithm({ <x_i , y_i> : x_i ↛ s* })
       Return Node t
ENDIF
```

Figura 2 - Preudocodice algoritmo ricorsivo di partizionamento

I 3 parametri principali di questo algoritmo sono:

- il criterio per sceglere il test di split
- la regola di terminazione dell'albero
- una regola per assegnare un valore per ogni nodo foglia (predizione)

Esistono diversi approcci per applicare questo algoritmo che si basano sull'assegnazione di questi tre criteri, alcuni esempi possono essere cercare di ridurre al minimo l'errore quadratico medio o la deviazione media assoluta dell'albero risultante.
Andando a selezionare i diversi parametri nel modello e' possibile durante la fase di addestramento utilizzare il modello che si considera piu' appropriato per la base di dati a nostra disposizione.

3.4.2 Vantaggi e motivazioni

Un DTR puo' essere utilizzato per risolvere problemi di regressione ed e' facile da interpretare, comprendere e visualizzare. Rispetto ad altri algoritmi, la preparazione dei dati durante la pre-elaborazione in un albero decisionale richiede uno sforzo minore e non richiede la normalizzazione dei dati, in piu' l'implementazione puo' essere effettuata senza scalare i dati.

Tutte queste pratiche invece sono fondamentali quando si parla di un modello neurale opaco, in modo tale da performare a meglio i risultati. Inoltre, e' possibile creare nuove caratteristiche per migliorare la previsione delle variabili target, essi non sono influenzati da valori errati o mancanti e possono gestire qualsiasi tipo di variabile (numerica o categorica).

Capitolo 4

Progettazione

La fase di progettazione definisce il sistema che dovrà utilizzare gli strumenti ricavati dai capitoli precedenti, può essere pensata come la creazione di soluzioni alle funzionalità da implementare e per questo motivo durante la fase di analisi bisogna sempre considerare lo scopo, l'analisi e gli strumenti che verranno poi adoperati. L'obiettivo di questa sezione, è quello di ottenere una descrizione concisa di tutti i componenti del sistema, partendo analizzando gli strumenti utilizzati, che possono essere ad esempio modelli o architetture software , analizzando gli utilizzi da parte dell'utente e il risultato finale.

4.1 Obiettivo

Lo scopo di questo lavoro di tesi è stato quello di affrontare lo studio della XAI, andando di seguito ad effettuare un analisi o confronto tra due modelli opposti tra loro. In particolare si e' cercato di verificare, se un modello opaco possa essere approssimato da un modello trasparente, andando ad analizzare ed effettuare un confronto tra le predizioni ottenute dai due modelli. Le conoscenze acquisite e maturate durante questo studio, sono state poi applicate nella fase di revisione, per analizzare al meglio gli output e tracciare un risultato. Il tutto e' nato partendo da una ricerca di un set di dati che potesse fare al caso nostro, in particolare e' stato scelto un dataset di serie temporali di processi di business.

4.2 Strumenti utilizzati

In base alle caratteristiche principali della tesi è stata scelta una implementazione attraverso Python, il quale possiede numerosi vantaggi per il tipo di progetto che e' stato sviluppato:

- **semplicità'**: a differenza di numerosi altri linguaggi che offrono algoritmi complessi e sfumature tecniche del linguaggio, python si concentra su flussi di codice versatile, leggibile e semplice da imparare, permettendo allo sviluppatore di concentrarsi esclusivamente sulla risoluzione di problemi di machine learning.
- **librerie**: I numerosi framework, librerie ed estensioni di python offrono una implementazione di diverse funzionalita' nell'ambito del machine learning, disponendo quindi di questo ambiente ben strutturato e testato si accorciano notevolmente i tempi di implementazione di algoritmi di ML.
- **popolarità'**: secondo la developer survey 2021[5] python e' stato uno dei 5 linguaggi di programmazione piu' utilizzati, il che implica il fatto che possieda una community molto piu' ampia per risolvere eventuali problematiche, in piu' nella maggior parte dei casi, viene utilizzato per ML e analisi dei dati.

4.3 Procedimento

4.3.1 Analisi e pulizia dataset

Il dataset scelto, presentava una serie di variabili numeriche e categoriche, inoltre erano presenti dati mancanti o dati in formati non corretti. Questa fase del procedimento quindi, prevedeva la pulizia del dataset, la correzzione dei formati da parte di alcune variabili e la trasformazione delle variabili categoriche in variabili numeriche, i quali potevano essere appresi dal modello neurale utilizzato in seguito. Inoltre, le variabili numeriche, vengono infine normalizzate per varie motivazioni:

- fase di addestramento piu' veloce e miglioramento delle performance
- possibilità' di utilizzare funzioni di attivazione per reti piu' profonde
- inizializzazione dei pesi meno rilevante

Per eseguire la rimozione,la formattazione e il salvataggio di alcune variabili categoriche del dataset ho applicato delle funzioni lambda, per la normalizzazione invece abbiamo utilizzato sklearn.preprocessing e in particolare MinMaxScaler e LabelEncoder. Di seguito e' evidenziato il loro utilizzo.

```
def labeltofloat(db):
  values = db.values
  encoder = LabelEncoder()
  db_cleaned = encoder.fit_trasform(values)
  db_cleaned = db_cleaned.astype('float32')
  return db_cleaned

def normalize(db):
  scaler = MinMaxScaler(feature_range=(0, 1))
  scaled = scaler.fit_transform(db)
  return scaled

def mappinglabel(label):
  mapping_label = {item:i for i, item in enumerate(label.unique())}
  return mapping_label
```

Figura 3 Implementazione metodi pulizia e normalizzazione

4.3.2 Fase di addestramento e ottimizzazione

L'addestramento e ottimizzazione del modello prevedeva varie fasi, inizialmente e' stato diviso il dataset ottenuto dalla fase precedente in dataset per l'addestramento e per il test.

In seguito e' stato allenato il modello, proseguendo poi con l'ottimizzazione attraverso la valutazione del settaggio di tre parametri: il numero di epoche, il numero di neuroni e il batch size, il quale indica il numero di campioni che verra' propagato attraverso la rete.

La ragione di ciò è dovuto alle condizioni iniziali casuali dei parametri di una rete LSTM, esse infatti possono produrre risultati molto diversi ogni volta che viene addestrata una determinata configurazione.

Verra' utilizzato uno scenario di previsione continua, ad ogni passaggio temporale verra' analizzato l'errore quadratico medio (RMSE) in quanto misura quanto le previsioni sono diverse dai valori reali attraverso l'utilizzo della distanza euclidea.

```
for k in range(5):
  # define model
  print('epoch: ' , epo)
  LSTM_model = Sequential()
  LSTM_model.add(LSTM(100, return_sequences=False, activation='relu' ,input_shape=(n_steps, n_features)))
  #model.add(LSTM(100, activation='relu'))
  LSTM_model.add(Dense(n_features))
  #LSTM_model.add(Dense(n_features, activation="softmax"))
  LSTM_model.compile(optimizer='adam', loss='mse')

  # fit model
  LSTM_model.fit(xtrain, ytrain, verbose = 0, epochs=epo)
  #LSTM_model.summary()

  dfpred_LSTM = []
  dfpred_LSTM = pd.DataFrame(columns=['time', 'kind', 'task','sourceName','Value'])
  for pred  in scaler.inverse_transform(LSTM_model.predict(xtest, verbose=0)):
    dfpred_LSTM = dfpred_LSTM.append(pd.Series(pred, index=['time','kind', 'task','sourceName','Value']), ignore_index=True)

  dfpred_LSTM.time = (dfpred_LSTM.time).astype(int)
  dfpred_LSTM

  rmse = calc_rmse(real, pred)
  ep.append(epo)
  res.append(rmse)
  print('rmse: ' , rmse)
  epo = epo+100

#for i in range(5):
#  print(ep[i],res[i])
```

```
epoch:  100
rmse:  60.59359758056991
epoch:  200
rmse:  61.24944922550938
epoch:  300
rmse:  61.12543495379409
epoch:  400
rmse:  61.58347071610851
epoch:  500
rmse:  61.71114793887821
```

Figura 4 Implementazione ottimizzazione epoche

Per quanto riguarda l'esempio evidenziato, notiamo come il miglior numero di epoche da settare sia 200, i risultati possono variare in quanto si tratta di un algoritmo di natura stocastica quindi la procedura di valutazione conviene eseguirla piu' volte in modo tale da confrontare le differenza nella precisione numerica, nell'esempio sottostante vengono mostrate altre due esecuzioni.

```
Epoch 500/500
161/161 [==============================] - 2s 13ms/step -
Epoch500rmse62.055315822310796
epoch RMSE
100 60.489207743334305
200 59.68963417642868
300 60.97953958619488
400 63.09300864871076
```

```
print('ep  mse')

  #yhat = model.predict(X, verbose=0)
#(model.predict(xtest, verbose=0))[0]
```

```
ep  mse
100 66.44805339065617
200 64.39215271278572
300 65.30493874499128
400 64.56445491349363
500 65.25958441394543
```

Figura 5 Risultati ottimizzazione epoche

4.3.3 Predizione

Dopo aver eseguito l'addestramento di entrambi i modelli, il nostro obiettivo era quello di generare delle predizioni, in modo tale da effettuare un analisi e verificare se era possibile approssimare i risultati grazie all'uso di un modello trasparente.
Il primo passo e' stato quello di generare le predizioni dal modello LSTM, ottimizzando il modello al meglio per generare i migliori risultati, di conseguenza poi abbiamo allenato il modello surrogato(DTR) in modo tale da cercare di mimare i risultati del modello LSTM. Il modello surrogato è stato definito attraverso l'uso della varianza interna come parametro per determinare quale fosse la miglior condizione iniziale per splittare il nostro dataset, la media invece è stata utilizzata per attuare la regressione e la profindità massima di 5 livelli come criterio di terminazione dell'albero.

```
print('distribuzione attributo kind LSTM: \n',df['kind'].value_counts().sort_index() )
print('distribuzione attributo kind surrogato: \n',sur['kind'].value_counts().sort_index())
```

```
distribuzione attributo kind LSTM:
 0.0     240
1.0     1042
2.0     1060
3.0      751
5.0      222
6.0     1807
7.0     1056
8.0      701
Name: kind, dtype: int64
distribuzione attributo kind surrogato:
 -1.0      45
 0.0      303
 1.0      940
 2.0     1020
 3.0      661
 4.0      159
 5.0      437
 6.0     1451
 7.0     1122
 8.0      618
 9.0      123
Name: kind, dtype: int64
```

Figura 6 Distribuzione attributo kind

```
distribuzione attributo sourceName LSTM:
 0.0     240
1.0     957
2.0     480
3.0     948
4.0     237
5.0     237
6.0     750
7.0     696
8.0     924
9.0      68
10.0     68
11.0    326
12.0    635
13.0    159
14.0    154
Name: sourceName, dtype: int64
distribuzione attributo sourceName surrogato:
 -3.0       3
-2.0       13
-1.0      118
-0.0      355
 1.0      601
 2.0      692
 3.0      598
 4.0      447
 5.0      465
 6.0      592
 7.0      654
 8.0      653
 9.0      310
 10.0     220
 11.0     320
 12.0     359
 13.0     286
 14.0     133
 15.0      47
 16.0      13
```

Figura 7 Distribuzione attributo sourcename

Trattandosi di 2 attributi che devono ricoprire un certo range di valori, in quanto fanno riferimento a determinate istruzioni possibili nella fase del processo, notiamo come la distribuzione di entrambi i valori sia molto simile tra i 2 modelli, nel paragrafo successivo proviamo a proiettarle attraverso un grafico per verificare se sono approssimabili o meno.

4.3.4 Confronto e analisi

Infine, come ultima fase sono stati analizzati i risultati ottenuti, attraverso l'utilizzo dell'indice MSE e proiettando alcuni dei nostri attributi attraverso un grafico per verificare se la distribuzione dei dati potesse essere simile.

Abbiamo quindi calcolato l'indice MSE tra il dataset iniziale e le predizioni del modello LSTM, andando a verificare dopo varie iterazioni quale fosse la stima media di questo valore, ottimizzando il modello, il risultato è un errore del due per mille, precisamente intorno al 0.002.

Applicando invece l'indice MSE per il modello surrogato le predizioni sono peggiorate (situazione prevedibile trattandosi di un modello molto meno complesso rispetto LSTM) arrivando ad un massimo del tre per cento, precisamente intorno a 0.033, cambiando quindi ordine di grandezza ma non rivoluzionando il dataset.

Passiamo ora ad analizzare alcuni grafici delle predizioni per verificare se se la distribuzione viene approssimata o meno.

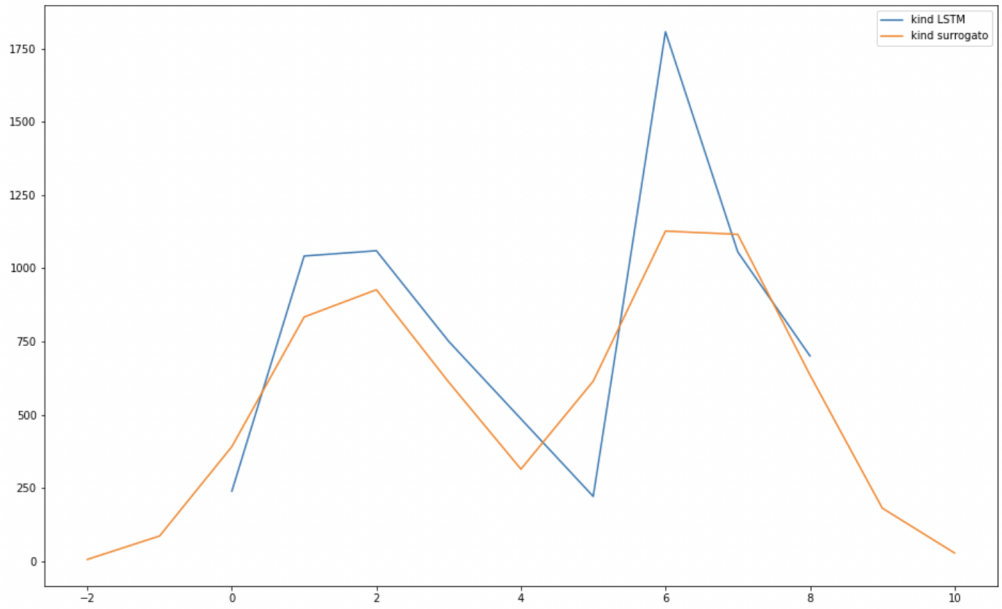

Figura 8 Distribuzione attributo kind

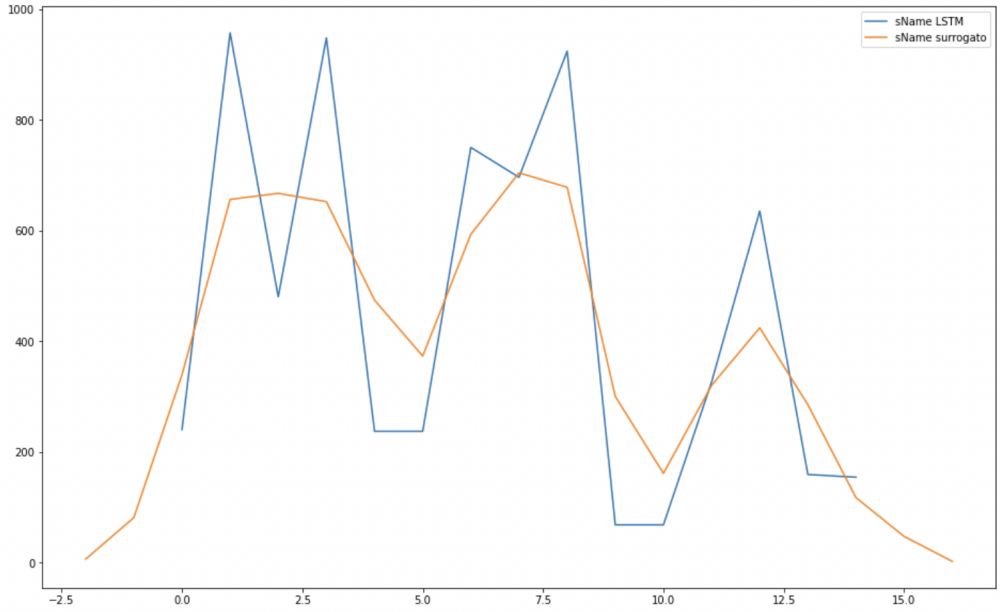

Figura 9 Distribuzione attributo sourcename

Analizzando i due ultimi grafici che rispecchiano alcune predizioni ottenute dai nostri modelli, possiamo notare come la distribuzione dei dati (senza considerare la minima generazione di alcuni outlier) venga approssimata notevolmente applicando un algoritmo surrogato anzichè utilizzare un algoritmo opaco come LSTM, i risultati dell'indice MSE vengono dunque confermati. Attraverso i dati analizzati deduciamo quindi che l'obiettivo della nostra tesi risulta portato a termine, ulteriori confronti attraverso l'uso di diversi dataset potrebbero portare ad una verifica ancora più attendibile dei nostri risultati anche se la loro reperibilità in questo determinato ambito come quello dei processi di business risulta molto complicato in quanto pochissime aziende preferiscono mettere a disposizione della comunità processi così importanti all'interno di una organizzazione.

Conclusioni

Lo scopo di questo lavoro di tesi è stato quello di analizzare se un modello neurale opaco possa essere approssimato attraverso l'uso di un modello trasparente, in modo tale da riuscire ad interpretare al meglio i risultati ottenuti dal modello.
Questa analisi nasce dallo studio della scienza della XAI, si tratta di quell'ambito di ricerca il quale si occupa di studiare l'interpretazione dei risultati da parte di modelli neurali opachi, cercando di analizzare le sue caratteristiche e la sua comprensibilità'. La XAI adotta l'uso di modelli e metodi per cercare di far conoscere all'utente il comportamento di un modello opaco e riuscire a interpretare i risultati ottenuti. Inoltre, le conoscenze acquisite e maturate sono state applicate per la progettazione di questi due modelli, applicati ad un dataset di processi di business.
Tramite questi strumenti è stato raggiunto l'obiettivo principale del progetto di tirocinio il quale prevedeva il fatto di approssimare il piu' possibile i risultati del modello neurale opaco attraverso l'uso di un modello neurale trasparente.
La tesi realizzata offre molti spunti per sviluppi futuri, come l'analisi con nuovi modelli sia opachi che trasparenti o l'analisi attraverso la comparazione di maggiori dataset visionati. Durante il tirocinio ho avuto la possibilità di applicare numerosi concetti appresi durante il percorso didattico come il miglioramento della conoscenza di python e delle sue librerie, inoltre, ho appreso la conoscenza di come vengono sviluppato,

ottimizzati e comparati tra di loro diversi modelli neurali complessi e semplici.

Bibliografia

[1] Hochreiter, Sepp Schmidhuber, Jürgen. (1997).
Long Short-term Memory. Neural computation.
9. 1735-80. 10.1162/neco.1997.9.8.1735.

[2] Jason Brownlee (2017)
A Gentle Introduction to Backpropagation Through Time
https://machinelearningmastery.com/gentle-introduction-backpropagation-time

[3] Chi-Feng Wang (2019)
The Vanishing Gradient Problem
https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484

[4] Leo Breiman (1984)
Classification And Regression Trees
https://www.taylorfrancis.com/books/mono/10.1201/9781315139470/
classification-regression-trees-leo-breiman

[5] Developer Survey 2021 by Stack Overflow, https://lp.jetbrains.com/python-developers-survey-2021/