Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea Magistrale in Informatica

Tecniche di explainability applicate a modelli predittivi in ambito didattico

Relatore:
Chiar.mo Prof.
Maurizio Gabbrielli

Presentata da:
Alessandra Boccuto

Correlatori:
Dott. Stefano Pio Zingaro
Dott. Andrea Zanellati

III Sessione
Anno Accademico 2021/2022

Introduzione

Le moderne soluzioni di intelligenza artificiale sono sempre più in grado di risolvere problematiche reali, fornendo un efficiente supporto in numerose attività come quelle decisionali e di automazione dei processi. Il guadagno derivante dall'utilizzo di tali soluzioni, sia in termini di performance che di facilità nella risoluzione di task complessi, ha portato nel tempo ad un incremento esponenziale della loro adozione da parte di aziende e organizzazioni operanti in numerosi campi, come ad esempio quello industriale, ludico ed educativo. Data la generale complessità dei problemi da affrontare, solitamente questi sono efficacemente modellati da modelli di machine learning flessibili e altrettanto complessi, come ad esempio le reti neurali, capaci di riconoscere pattern non triviali nei dati. Tali modelli sono però solitamente caratterizzati da un certo livello di opacità, ovvero il loro comportamento interno non è di facile comprensione e di conseguenza non risulta chiaro il processo decisionale dietro le previsioni prodotte. Questa opacità, propria dei cosiddetti modelli black-box, non è una caratteristica desiderabile in numerosi contesti di utilizzo, come ad esempio in campi critici, come quello medico o giudiziario, o durante il coinvolgimento in fasi decisionali, in quanto è fondamentale che le decisioni si basino su risultati di modelli comprensibili e trasparenti per tutte le parti interessate, facendo sì che la fiducia nelle decisioni comunicate non venga meno. In questi casi risulta quindi fondamentale cercare un modo per spiegare l'iter decisionale seguito da un modello, anche se questo non risulta essere interpretabile per sua natura. Ciò è possibile applicando dei metodi specifici, di explainability (spiegabilità), che hanno come

obbiettivo proprio quello di fornire spiegazioni in differenti modalità al fine di migliorare la comprensione di modelli di machine learning complessi e difficili da interpretare. La spiegabilità, ovvero la capacità di spiegare il comportamento di un modello complesso, come esposto nel lavoro [1], può consentire di valutare la validità delle decisioni basate sui modelli e di prevenire eventuali errori mitigando i rischi associati all'automazione delle decisioni, come la presenza di pregiudizi ingiusti. Inoltre, la spiegabilità si pone alla base di modelli IA trasparenti e appropriati e concorre, insieme ad altre caratteristiche come la robustezza e l'assenza di bias, all'aumento della fiducia da parte degli stakeholders coinvolti nella loro creazione, valutazione e utilizzo. Infine, l'explainability può essere anche utilizzata per fornire ulteriori metriche a supporto della scelta dei migliori modelli di machine learning prodotti per la risoluzione di un problema. Infatti, per confrontare diverse soluzioni, la comprensione del ragionamento del modello può essere fondamentale in modo da non essere guidati solo da metriche relative alle performance ma anche dalla robustezza e sensatezza dell'iter decisionale seguito, come mostrato in [14] (pag.2, fig 2).

Le tecniche fini al rendere un modello spiegabile possono essere raggruppate nel campo XAI (eXplainable Artificial Intelligence), il quale include numerosi metodi che si differenziano in base a diversi elementi come il destinatario delle spiegazioni (con uno specifico livello di tecnicismo associato) e la fase di vita del modello in cui si applicano (se prima, durante la progettazione o dopo l'addestramento). In generale, comunque, tutte le metodologie condividono l'obbiettivo comune di consentire agli stakeholders di comprendere e considerare attendibili i risultati comunicati dalle soluzioni di Intelligenza artificiale. L'importanza della spiegabilità dei modelli è evidente anche nei campi del Learning Analytics (LA) e Educational Data Mining (EDM). LA e EDM condividono tecnologie per comprendere al meglio i processi di apprendimento degli studenti e migliorare il campo educativo promuovendo un approccio data-driven a livello organizzativo. Tuttavia, mentre EDM si concentra su tecniche di data mining per analizzare i dati educativi, LA ha

un maggiore focus sulla raccolta e l'analisi dei dati relativi alle esperienze di apprendimento degli studenti. In entrambi i casi, l'utilizzo di modelli spiegabili che forniscono previsioni comprensibili alle parti interessate è cruciale per permettere a educatori esperti del dominio di valutare e correggere eventuali errori evitando che decisioni errate possano avere delle conseguenze dannose per gli studenti coinvolti o a livello organizzativo [15]. Inoltre, tali modelli possono anche essere utilizzati per comprendere meglio i processi di apprendimento riconosciuti, generando nuova conoscenza nell'ambito [1].

Nel presente lavoro di tesi è proposto il confronto di due tecniche di Explainability applicate a modelli di machine learning sviluppati in uno studio di EDM. Tale studio è stato presentato nel lavoro di Zanellati, Zingaro e Gabbrielli (2022) [19] e consiste nell'addestramento di classificatori binari per prevedere se uno studente avrà un basso rendimento scolastico in seconda superiore, sulla base dei risultati dei test INVALSI di quinta elementare ed altri fattori culturali, socio-economici e demografici. I modelli proposti per la risoluzione di tale problema sono di tre tipologie differenti. Il primo modello è un random forest, algoritmo di apprendimento di insieme che coordina più alberi decisionali nella predizione, il secondo è una categorical embeddings neural network, una rete neurale che esegue il mapping degli attributi categorici tramite un embedding layer parte della rete, e l'ultimo è un feature tokenizer transformer, un adattamento dell'architettura transformer per dati tabulari. Mentre il random forest risulta essere un modello interpretabile per natura, dato che le feature originali sono direttamente utilizzate nell'iter decisionale che definisce l'output, gli ultimi due modelli risultano invece essere opachi, in quanto nel processo decisionale sono utilizzate delle feature interne estratte a partire da quelle originali in seguito a numerose trasformazioni lineari e non. Quindi, per il random forest risulta facile comprendere l'importanza di ogni feature rispetto l'output finale, mentre per gli altri due modelli non è un'attività triviale. Il lavoro di tesi presentato è stato formulato per affrontare questa problematica, applicando a tali modelli tecniche capaci di definire l'importanza di ogni feature originale nel calcolo dell'output finale.

Il confronto risulta utile per valutare la robustezza delle soluzioni, osservare se i modelli si concentrano sulle stesse caratteristiche e per fornire in generale una spiegazione intuitiva agli esperti del settore per comprendere il comportamento dei modelli proposti e aumentare la fiducia nelle predizioni fornite. Al contempo, tali analisi permettono anche di osservare ed eventualmente scoprire nuovi pattern riconosciuti.

Le due tecniche XAI scelte per questo fine sono la SHapley Additive exPlanations (SHAP) [11] e Local Interpretable Model-Agnostic Explanations (LIME) [14], le quali permettono di comprendere l'influenza delle diverse feature sulle singole predizioni dei modelli. Entrambi sono metodi additivi e locali appartenenti all'ambito di post-modelling, in quanto possono essere applicati a modelli già definiti e addestrati. La scelta di SHAP e LIME come tecniche di explainability nel presente lavoro è stata motivata principalmente dalla necessità di fornire spiegazioni semplici e comprensibili per stakeholders non tecnici ma esperti del dominio educativo. Inoltre, entrambe le metodologie forniscono spiegazioni dello stesso stipo, ovvero sotto forma di effetti additivi delle feature coinvolte, permettendo un confronto tra i modelli presi in considerazione nello studio.

Lo sviluppo del lavoro proposto ha quindi richiesto l'esecuzione di esperimenti per determinare l'importanza globale delle variabili per ogni modello analizzato. A tal fine, sono stati calcolati, sia con la tecnica SHAP che LIME, i singoli effetti delle feature per ogni predizione relativa alle istanze di un dataset di riferimento. Successivamente, per ogni attributo, i valori sono stati aggregati per ottenere la loro importanza generale. A seguito degli esperimenti è stata inoltre ricercata una coerenza tra i risultati ottenuti per i diversi modelli, osservando se le feature più influenti dei modelli fossero le stesse e in che magnitudo.

Il primo capitolo della tesi presenta una panoramica generale delle tecniche di explainability di modelling e post-modelling, con un particolare accento sulle applicazione delle ultime in formato locale e agnostico nel campo dell'Educational Data Mining. Il secondo, invece, espone nello specifico le

tecniche SHAP e LIME e la loro modalità di utilizzo nello studio presentato. Il terzo capitolo descrive il setup e i dettagli degli esperimenti condotti, i quali risultati saranno poi analizzati nel quarto. Infine, nel quinto capitolo sono discusse le implicazioni pratiche e le conclusioni della tesi, con uno sguardo anche a possibili futuri sviluppi.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Revisione della letteratura

Il campo dell'explainability dei modelli black-box è ancora in forte fase di sviluppo all'interno della comunità del machine learning. Tuttavia, negli ultimi anni si è assistito ad un forte aumento dell'interesse da parte della ricerca e dell'industria su questo argomento, in parte osservato anche dagli aumenti correlati delle ricerche del termine "Explainable AI" su Google e gli studi pubblicati a riguardo nello stesso periodo, come illustrato in [9]. Le tecniche di spiegazione dei modelli black-box possono essere suddivise in tre macro-categorie: di pre-modelling, modelling e post-modelling, come definito in [6]. Nel primo caso, si adottano tecniche per analizzare i dati in modo da favorire l'esplicazione dei risultati del modello; nel secondo, invece, si modifica la struttura del modello stesso per renderlo più interpretabile, come ad esempio sfruttando i modelli ibridi DKNN [12], CEN [2] e TED [5]; nel terzo caso, si applicano tecniche di spiegazione al modello già addestrato; un esempio di tali metodi è DeepRED [20], per l'estrazione di un decision tree interpretabile da una rete neurale. Tra le tecniche di spiegazione di post-modelling ne figurano anche alcune agnostiche, utili soprattutto per effettuare confronti tra modelli diversi grazie alla loro versatilità: i metodi LIME [14] e Kernel SHAP [11] fanno parte di questo sottogruppo.

Nel campo dell'Education Data Mining (EDM), tali metodi di spiegazione sono sempre più diffusi e consolidati e, nell'ultimo anno, l'incremento

di studi relativi all'applicazione delle metodologie SHAP e LIME ha dato molti segnali positivi rispetto la loro efficacia. In particolare, lo studio [7] ha sperimentato modelli di machine learning tradizionali per la predizione del successo degli studenti e ha utilizzato LIME per fornire spiegazioni locali relative al modello migliore. Simile obbiettivo è stato conseguito e ricercato anche dallo studio [18], il quale ha esplorato l'utilizzo di LIME su metodi ensemble di machine learning per la predizione del rendimento degli studenti. Lo studio [4], invece, ha sfruttato le importanze SHAP per interpretare modelli di predizione dell'abbandono scolastico degli studenti. Similmente, il lavoro di Ramaswami et al. (2022) [13] ha utilizzato la medesima tipologia di valori per valutare l'importanza delle feature per un modello predittivo binario CatBoost sviluppato per identificare gli studenti a rischio di basso rendimento accademico, ciò sulla base di informazioni legate all'ambito didattico dei corsi di studio e ad altri dati culturali e demografici. Inoltre, lo studio di Alwarthan et al. (2022) [3] ha utilizzato entrambi i metodi LIME e SHAP per spiegare gli output dei modelli di random forest e SVM utilizzati per rilevare precocemente gli studenti a rischio di non superare l'anno di preparazione, evidenziando le ragioni decisionali dietro le previsioni prodotte. Altri studi hanno inoltre lavorato sull'integrazione in dashboard delle spiegazioni individuali fornite da LIME, confermando la loro efficacia, in combinazione con sistemi di simulazione, nel permettere una comprensione intuitiva da parte degli educatori [16].

Degno di nota è anche lo studio condotto da Swamy et al. (2022) [17], in cui sono state valutate e testate cinque tecniche di spiegazione confermate nello stato dell'arte (LIME, Permutation SHAP, Kernel SHAP, DiCE, CEM), applicandole a modelli LSTM per la predizione del successo degli studenti in un ambiente MOOC. I ricercatori hanno scoperto che le diverse famiglie di spiegazioni non sono generalmente concordi sulla stessa importanza delle variabili, evidenziando la necessità di utilizzare più di una tecnica di spiegazione per ottenere una comprensione completa del modello.

Sulla base di tali ricerche e del visibile incremento dell'applicazione delle

tecniche SHAP e LIME, specificatamente per la predizione del rendimento scolastico e abbandono, si è ritenuto che tali metodologie potessero essere rilevanti anche per la ricerca proposta nella presente tesi. Infatti, gli studi esaminati hanno dimostrato l'utilità di SHAP e LIME per interpretare modelli di machine learning e fornire spiegazioni intuitive e comprensibili agli educatori. Dato l'agnosticismo dei metodi e la loro conseguente applicazione, con successo, su differenti tipologie di modelli, si è ritenuto che l'impiego di SHAP e LIME nello studio proposto potesse essere altamente utile per fornire spiegazioni intuitive anche per il funzionamento dei modelli di machine learning considerati.

Capitolo 2

Descrizione del metodo

Le tecniche di spiegabilità applicate e confrontate sui tre modelli oggetto di analisi del presente lavoro di tesi sono LIME (Local Interpretable Model-Agnostic Explanations) e SHAP (SHapley Additive exPlanations), quest'ultima nelle sue due varianti Kernel SHAP e Tree SHAP.

Le due metodologie si presentano come tecniche locali che appartengono al sottogruppo dei metodi XAI di tipo post-modelling. Con la caratteristica di località si intende che tali metodi non hanno il fine di fornire spiegazioni relative al funzionamento dell'intero modello, bensì del suo comportamento locale per singole predizioni. Con il termine post-modelling, invece, si intende che forniscono spiegazioni per modelli già creati e addestrati, differenziandosi così dalle tecniche di modelling che lavorano sul miglioramento dell'interpretabilità di un modello durante la sua fase di progettazione. Inoltre, LIME e la variante SHAP di tipo Kernel sono tecniche agnostiche che possono essere applicate a qualsiasi modello di machine learning indipendentemente dalla sua tipologia o architettura. Tree SHAP, invece, è specifica e ottimizzata per l'esecuzione su modelli basati su alberi.

Le proprietà esposte rendono questi metodi particolarmente adatti all'analisi del comportamento di modelli già esistenti di cui non è possibile rivisitare l'architettura per migliorarne l'interpretabilità. Inoltre, l'agnosticismo che caratterizza LIME e Kernel SHAP facilita successive comparazioni

tra iter decisionali di modelli differenti.

SHAP e LIME hanno quindi molte caratteristiche comuni e sono anche classificate in egual modo secondo i criteri relativi alla *Four-Aspect Taxonomy of post-hoc Explainability* (FATE), la tassonomia esistente per la classificazione dei metodi XAI di tipo post-modelling. Entrambi i metodi, infatti, analizzano l'impatto delle feature (driver della spiegazione) coinvolte in singole predizioni (target). Le spiegazioni che forniscono sono sotto forma di importanze (famiglia di spiegazioni) associate agli attributi, ottenute lavorando su perturbazioni dell'input originale (processo computazionale).

Nello specifico, LIME e SHAP si basano sull'idea che la miglior spiegazione per un modello sia il modello stesso, ma nel caso in cui questo sia troppo complesso per essere compreso direttamente, è possibile utilizzarne uno più interpretabile che ne approssimi il comportamento.

Sulla base di questo approccio, i due metodi creano, a partire da una singola predizione, un modello interpretabile più semplice, chiamato explanation model (modello di spiegazione), che approssima localmente il comportamento del modello black-box complesso. Il modello di spiegazione locale generato permette di ottenere la misura esatta con cui gli attributi dell'input originale del modello contribuiscono al calcolo della predizione restituita. Infatti, i pesi relativi alle variabili del modello di spiegazione possono essere considerati come le effettive contribuzioni che essi apportano all'output finale del modello originale.

E' possibile fare tale considerazione in quanto LIME e SHAP rientrano nel gruppo degli additive feature attribution methods, ovvero dei metodi che generano modelli di spiegazione come funzioni lineari di variabili binarie, le quali sono una versione semplificata delle feature dell'input originale. I modelli in questo formato prendono il nome di additive explanation models (modelli additivi di spiegazione) ed il loro dominio binario permette di poter definire la predizione in output come somma dei pesi associati alle variabili non nulle. La spiegazione fornita sotto forma di contribuzioni effettive delle feature risulta essere intuitiva e facilmente rappresentabile graficamente,

presentandosi particolarmente adatta per l'analisi da parte di esperti del dominio di applicazione dei modelli, in questo caso quello didattico, non tecnici dell'ambito del machine learning.

Inoltre, tali valori, se ottenuti per una quantità significativa di istanze, possono essere combinati in modo da ottenere una spiegazione globale del modello, che assegna un'importanza generale ad ogni feature. Il concetto di importanza globale di un attributo rispetto ad un modello è specificato dal framework SHAP ma è facilmente estendibile anche a LIME, in quanto, per ottenere la misura di importanza globale di un attributo è sufficiente calcolare la media dei suoi valori SHAP o LIME assoluti (equazione 2.1).

$$I_j = \frac{1}{m} \sum_{i=1}^{n} |\phi_j^{(i)}| \tag{2.1}$$

L'utilizzo del valore assoluto inibisce il concetto di additività ma risulta una scelta sensata considerando un valore LIME o SHAP importante unicamente in base alla sua magnitudo, a prescindere da se apporta un aumento o decremento dell'output finale mentre concorre al suo calcolo.

Nella sottocapitolo 2.1 sono esposte le caratteristiche generali degli additive feature attribution methods, di cui LIME e SHAP fanno parte, e dei modelli di spiegazione che generano. Le sezioni 2.2 e 2.3, invece, approfondiranno, rispettivamente, le due tecniche utilizzate. Infine, seguirà il sottocapitolo 2.4 relativo all'applicazione dei metodi al caso di studio dei modelli di predizione del basso rendimento scolastico.

2.1 Additive feature attribution methods

La classe degli additive feature attribution methods, di cui fanno parte le due tecniche LIME e SHAP, è stata presentata per la prima volta nel 2017 da Lundberg e Lee [11], nello stesso lavoro in cui è stato proposto anche il metodo SHAP.

I metodi appartenenti a questa classe generano spiegazioni per singole predizioni sotto forma di modelli interpretabili, più semplici, che approssi-

mano il comportamento locale di un modello complesso di cui risulta difficile interpretare direttamente le importanze delle feature rispetto l'output.

Entrando più nello specifico, considerando un modello originale black-box f, le tecniche di generazione di spiegazioni locali sono progettate per spiegare una specifica predizione $f(x)$ basata sull'input x creando un modello interpretabile g che approssima il comportamento di f sull'input x. Una caratteristica importante di tali modelli di approssimazione è l'utilizzo di una versione semplificata x' dell'input x, che può essere facilmente trasposta nel dominio originale tramite una funzione di mapping definita ad-hoc $h_x(x') = x$. Gli additive feature attribution methods rientrano nell'insieme di questi metodi locali e generano dei modelli che soddisfano specificatamente l'equazione 2.2, la quale li vincola ad essere funzioni lineari di variabili binarie.

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i \tag{2.2}$$

dove $z' \in \{0,1\}^M$, M è il numero di feature per l'input semplificato e $\phi_i \in \mathbb{R}$.

La caratteristica interessante dell'equazione 2.2 è che, essendo il dominio di g binario, i pesi ϕ_i assegnati alle variabili possono essere considerati come dei contributi effettivi, che, se sommati, corrispondono esattamente al risultato di $g(x')$ che approssima l'output $f(x)$ del modello originale. In riferimento a ciò, è inoltre possibile considerare ϕ_0 come l'output base di partenza del modello.

Nella pubblicazione di Lundberg, metodi locali già esistenti come LIME, DeepLIFT, Shapley Regression Values e Shapley sampling values (entrambi basati sul teorema dei valori Shapley dalla teoria cooperativa dei giochi) sono stati analizzati e assegnati a questa classe, in quanto i modelli di spiegazione da essi generati aderiscono all'equazione 2.2. Inoltre, sempre nel lavoro in oggetto, è stata introdotta la tecnica SHAP, anch'essa parte degli additive feature attribution methods, proposta per migliorare questi metodi esistenti al fine di estendere loro il rispetto di alcune proprietà desiderabili. Tali proprietà da considerare e ricercare in modelli locali che fungono da spiegazione

per un modello complesso sono la *local accuracy* (accuratezza locale), la *missingness* (mancanza delle feature corrispondente a peso nullo) e la *consistency* (consistenza).

La proprietà relativa alla *local accuracy* definisce la capacità del modello di spiegazione di replicare fedelmente l'output del modello originale black-box per l'input specifico di cui si vuole spiegare la predizione. Tale proprietà, rappresentata dall'equazione 2.3, risulta rispettata se il modello interpretabile g, generato per il modello originale f sull'input x, fornisce in output un valore uguale alla funzione f almeno per l'input x' (ovvero la versione semplificata di x).

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{2.3}$$

con $x = h_x(x')$.

La seconda proprietà ricercata è invece la *missingness* (equazione 2.4), la quale stabilisce che nel caso in cui x' rappresenti la presenza o assenza delle feature nell'input originale, allora l'effetto ϕ_i associato nel modello interpretabile a $z'_i = 0$ sarà nullo e non avrà quindi impatto sull'output finale.

$$x'_i = 0 \implies \phi_i = 0 \tag{2.4}$$

A differenza delle altre due proprietà, la *missingness* è l'unica soddisfatta da tutti gli additive feature attribution methods esistenti.

L'ultima proprietà desiderabile è invece la *consistency*, secondo cui, se il modello originale viene modificato in modo tale che il contributo di un attributo aumenti o rimanga lo stesso, allora l'effetto associato alla feature nel modello di spiegazione non deve diminuire. Nello specifico, dato $f_x(z') = f(h_x(z'))$, definendo che $z' \backslash i$ specifichi che $z'_i = 0$ e dati due modelli qualsiasi f e f', se per ogni input $z' \in \{0,1\}^M$

$$f'_x(z') - f'_x(z' \backslash i) \geq f_x(z') - f_x(z' \backslash i) \tag{2.5}$$

allora

$$\phi_i(f', x) \geq \phi_i(f, x) \tag{2.6}$$

Esiste un teorema secondo cui l'unica soluzione che permette ad un modello di spiegazione di soddisfare queste tre proprietà prevede l'utilizzo di valori Shapley come pesi per le variabili. I valori Shapley, già precedentemente citati, sono un concetto appartenente alla teoria dei giochi e rappresentano le contribuzioni marginali di un giocatore (in questo caso di una feature) rispetto il risultato finale del gioco (la predizione). I valori Shapley possono essere calcolati osservando le differenze nell'output rispetto diverse coalizioni di feature coinvolte, al fine di poter valutare quanto la presenza di una feature impatti nella predizione finale. Nello specifico, il teorema in questione (definizione 2.7), stabilisce che solo un possibile modello di spiegazione g può risultare conforme all'equazione 2.2 e soddisfare allo stesso tempo le tre proprietà di *local accuracy*, *missingness* e *consistency*. Il modello g deve avere i pesi ϕ calcolati nella seguente modalità:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z' \setminus i)] \tag{2.7}$$

dove $|z'|$ è il numero degli attributi non nulli in z', e $z' \subseteq x'$ rappresenta tutti i vettori z' i quali elementi diversi da 0 contenuti sono un sottoinsieme degli elementi diversi da 0 in x'. Tale equazione, così definita, fa sì che ϕ_i sia calcolato come valore Shapley, prendendo il nome, in questa specifica metodologia, di valore SHAP. Nel 1985, Young dimostrò che solo i valori Shapley soddisfano due assiomi simili alle proprietà di *local accuracy* e *consistency*, mentre la *missingness* è stata successivamente dimostrata per adattarli all'utilizzo da parte degli additive feature attribution methods. SHAP si presenta come un approccio unificato proposto al fine di estendere il soddisfacimento della prima e terza proprietà anche agli altri metodi già esistenti, basandosi sull'utilizzo e l'approssimazione dei valori Shapley come effetti delle feature sull'output.

Come anticipato nell'introduzione del capitolo, esistono diverse tecniche basate su SHAP, in quanto questo si presenta come un framework, e possono essere sia agnostiche che legate al modello. In generale l'obbiettivo comune di tutte le implementazioni è quello di approssimare i valori Shapley per

utilizzarli come importanze delle feature in una predizione. In questo specifico lavoro sono state prese in considerazione le due tecniche Kernel SHAP e Tree SHAP. La prima propone un'integrazione dei valori SHAP come pesi di modelli locali surrogati creati seguendo la logica della tecnica LIME. Esiste quindi una forte correlazione tra LIME e Kernel SHAP, dato che quest'ultimo si propone come un miglioramento del primo metodo. Risulta comunque interessante l'utilizzo di entrambe, in quanto si differenziano principalmente nella modalità in cui i pesi per i modelli di spiegazione sono computati. Tree SHAP, invece, definisce un approccio differente al problema e non lavora alla creazione di modelli surrogati bensì sfrutta la conoscenza dell'architettura sottostante basata su alberi per approssimare direttamente i valori Shapley.

2.2 LIME

La tecnica LIME (Local Interpretable Model Agnostic Explanations), è stata proposta per la prima volta nel 2016 da Ribeiro et al. [14]. Come introdotto anche nelle sezioni precedenti, LIME è un additive feature attribution method agnostico e locale, che genera spiegazioni sotto forma di modelli di spiegazione interpretabili, anche chiamati modelli surrogati, che approssimano il comportamento locale di qualsiasi modello black-box in relazione ad una singola predizione. L'approccio adottato della tecnica LIME è piuttosto intuitivo e consiste nell'utilizzo del modello complesso da spiegare come se fosse una scatola nera, da cui è unicamente possibile ottenere previsioni relative a input inseriti (è interessante notare che è tale comportamento che abilita l'agnosticismo del metodo). L'obbiettivo finale di LIME è quello di comprendere le motivazioni che portano il modello black-box a restituire una determinata previsione, calcolando le importanze delle variabili rispetto l'output del modello in termini di contributi effettivi, come spiegato precedentemente per gli additive feature attribution methods. A tal fine LIME crea, per una singola predizione, un modello di spiegazione interpretabile addestrato su un dataset contenente l'input originale e altre sue versioni

perturbate, tutte in una forma semplificata, con le corrispondenti previsioni ottenute somministrando i campioni al modello originale black-box. Durante l'addestramento del modello surrogato, i campioni perturbati sono pesati rispetto alla vicinanza all'input originale, in modo da incentivare e soddisfare la proprietà di *local accuracy*. La *local accuracy*, in questo caso, definisce che l'accuratezza del modello surrogato dev'essere alta nell'approssimazione locale della predizione effettuata dal modello black-box e non necessariamente soddisfacente nell'approssimazione del suo comportamento globale. LIME valuta quindi l'effetto di perturbazioni dell'input originale sulle previsioni del modello black-box da spiegare, permettendo di utilizzare i pesi associati alle variabili nel modello surrogato come misure di importanza delle variabili rispetto la predizione dell'output originale.

Formalmente il concetto può essere espresso come segue: dato il modello black-box originale $f : \mathbb{R}^d \to \mathbb{R}$, il surrogato generato da LIME per una predizione $f(x)$ può essere definito come un modello g appartenente ad una classe G di modelli interpretabili, come ad esempio algoritmi basati su alberi e modelli lineari. Il dominio del modello interpretabile g è $\{0,1\}^{d'}$, in quanto esso non si basa sull'input originale x bensì su una sua versione binaria più semplice e interpretabile, da cui è sempre possibile riottenere la versione nello spazio originale tramite una funzione di mapping h_x. Tale forma risulta in linea con l'equazione che caratterizza i modelli di spiegazione generati dagli additive feature attribution methods, e permette al modello di lavorare sulla presenza e assenza delle feature piuttosto che sul loro valore specifico. Ciò fa sì che i pesi possano essere valutati come contributi effettivi all'output.

Nel dettaglio, LIME genera un modello di spiegazione per una predizione $f(x)$ risolvendo il seguente problema di ottimizzazione:

$$\xi(x) = \arg\min_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{2.8}$$

in cui $\Omega(g)$ è una misura di complessità che vincola il modello a mantenere un alto livello di interpretabilità e $\pi_x(z)$ è una misura di prossimità tra un'istanza z e x. $\mathcal{L}(f, g, \pi_x)$ misura, invece, il livello di fedeltà al modello originale, permettendo la valutazione della qualità dell'approssimazione di g rispetto

il modello originale f in uno spazio locale definito da π_x. Per avere un modello sia interpretabile che localmente fedele, è necessario minimizzare $\mathcal{L}(f, g, \pi_x)$ mantenendo basso anche $\Omega(g)$. Le tre misure π, $\mathcal{L}$ e Ω possono essere liberamente configurate in base al comportamento ricercato.

La fase di sampling locale per la generazione del dataset Z per il training del modello di spiegazione per una predizione $f(x)$, è effettuata attraverso un'attività di campionamento randomico e uniforme di istanze intorno x' (input originale semplificato), dove i nuovi campioni perturbati $z' \in \{0, 1\}^{d'}$ sono frazioni di elementi diversi da 0 del campione originale semplificato x'.

Per i campioni $z' \in Z$ è possibile ottenere la corrispondente label $f(z)$, dove $z \in R^d$ è la trasposizione del campione z' nello spazio originale dell'input tramite una funzione di mapping h_x . I campioni perturbati z' possono essere a distanza arbitraria dall'input originale z, ma i più vicini ad essa avranno un peso maggiore durante il training del modello di spiegazione grazie alla misura di prossimità π_x.

Le perturbazioni applicate ai dati risultano essere differenti in base al tipo di dato su cui il modello originale lavora. Nel caso di dati tabulari, come quelli utilizzati nel presente lavoro, i nuovi campioni z' sono creati perturbando individualmente ogni variabile e assegnando valori ottenuti da una distribuzione normale con media e deviazione standard relativi alla variabile stessa nel dataset.

Inoltre, risulta interessante notare che LIME, nel caso in cui G sia una classe di modelli lineari, approssima il comportamento di un modello black-box solitamente non lineare basandosi sull'assunzione che localmente il modello possa avere un comportamento lineare.

La figura 2.1 definisce concretamente tale concetto, presentando la differenza tra un modello di spiegazione locale lineare, rappresentato dalla linea nera tratteggiata, e il comportamento visibilmente non lineare adottato dal classificatore binario originale, osservabile tramite i decision boundaries colorati sullo sfondo. I campioni z del dataset di background Z sono identificati da simboli differenti in base alla classe di appartenenza ed è possibile con-

siderare la croce più spessa come il campione di riferimento di cui si vuole spiegare la predizione. E' facilmente osservabile che il modello di spiegazione risulta essere fedele localmente al comportamento del modello black-box rispetto al campione di riferimento ma non approssima correttamente i confini decisionali globali del modello.

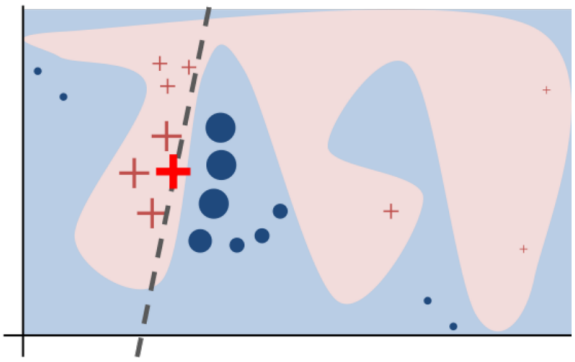

Figura 2.1: Intuizione sul rapporto tra modello di spiegazione lineare e modello originale (Ribeiro et al., 2016, p.4)

2.3 SHAP

SHAP (SHapley Additive exPlanation) è un approccio unificato per la valutazione dell'importanza degli attributi di un modello, esposto per la prima volta nel 2017 da Lundberg e Lee [11]. Come introdotto precedentemente nella sezione 2.1, SHAP nasce per migliorare gli additive feature attribution methods esistenti, permettendo loro di soddisfare, oltre la proprietà di *missingness*, anche quelle di *local accuracy* e *consistency*.

A livello intuitivo e in modo analogo a quanto osservato per il metodo LIME, SHAP ha l'obbiettivo di spiegare previsioni effettuate da un modello black-box su istanze specifiche, fornendo i contributi effettivi delle variabili rispetto la predizione finale. Nello specifico, per SHAP tali contributi devono coincidere con i valori Shapley degli attributi. I valori Shapley sono un concetto derivante dalla teoria dei giochi di coalizione e rappresentano la

contribuzione marginale di un attributo all'output finale. Nella visione di SHAP, infatti, le feature possono essere considerate come giocatori di una coalizione che partecipa al calcolo dell'output finale del modello. Il valore Shapley per una feature può essere calcolato utilizzando la formula 2.7, la quale lo definisce come una somma pesata delle differenze tra l'output del modello data una coalizione di attributi in input comprendente la feature in esame ($f_x(z')$) e l'output del modello data la stessa coalizione senza la feature ($f_x(z' \setminus i)$), ciò per ogni possibile coalizione (o sottoinsieme) degli attributi utilizzati dal modello.

Fondamentalmente in questo modo risulta possibile ottenere la contribuzione marginale dell'attributo nel calcolo dell'output, rispetto ad ogni possibile coalizione di cui può far parte. Dato che la maggioranza dei modelli non risulta capace di gestire input arbitrari non comprendenti tutti gli attributi originali con cui il modello è stato addestrato, nella realtà non si calcolano i valori Shapley della funzione $f_x(z_s)$, dove S è l'insieme degli indici degli elementi diversi da 0 in z', ma di una sua approssimazione definita dall'*expectation* $E[f(z)|z_S]$ $(= f(h_x(z')) = f_x(z'))$ dove il mapping $h_x(z') = z_s$ e z_s ha valori mancanti per le feature che non sono in S.

E' quindi possibile sostituire la funzione originale nell'equazione 2.7 con tale *expectation function*.

Al fine di proporre un'ulteriore intuizione relativa ai valori SHAP, la figura 2.2 mostra graficamente che essi attribuiscono a ciascuna caratteristica la variazione nella previsione attesa del modello quando si condiziona su quella caratteristica. Sommando i valori SHAP, caratterizzati dalla proprietà di additività in quanto associati a feature semplificate binarie, è possibile osservare l'iter che, a partire dal valore base $E[f(z)]$, previsto se non si conoscesse alcuna caratteristica, definisce l'output finale $f(x)$.

Data la complessità computazionale molto alta richiesta per il calcolo dei valori Shapley esatti, dovuta alla necessità di ripetere i calcoli per ogni coalizione di feature esistente, SHAP propone dei metodi per approssimare tali valori sfruttando e combinando logiche proprie degli additive feature

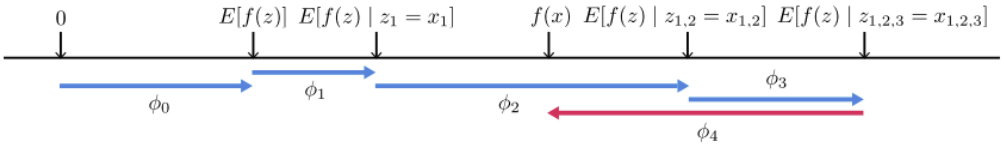

Figura 2.2: Additività dei valori SHAP ϕ_i (Lundberg et al., 2017, p.4)

attribution methods precedentemente esistenti.

Per il presente progetto di tesi sono stati utilizzati principalmente due dei diversi metodi proposti, **Kernel SHAP** (per il calcolo di spiegazioni in relazione alla categorical embeddings neural network e al feature tokenizer transformer) e **Tree SHAP** (per le spiegazioni relative al random forest da valutare).

Mentre Kernel SHAP si presenta come metodo agnostico di approssimazione, valido quindi per qualsiasi modello, Tree SHAP è invece un approccio di stima dei valori SHAP strettamente legato all'architettura dei modelli basati su alberi, risultando anche più efficiente computazionalmente.

2.3.1 Kernel SHAP

Kernel SHAP, come già introdotto nella sezione precedente, è un metodo locale e agnostico di approssimazione dei valori Shapley relativi agli attributi di un modello predittivo complesso per una singola predizione. Tale tecnica lavora sotto l'assunzione di indipendenza delle variabili, ovvero si assume che l'impatto dato dall'aggiunta di una feature ad una coalizione è sempre lo stesso a prescindere dalla composizione della coalizione stessa.

Kernel SHAP si presenta fondamentalmente come un adattamento del metodo LIME lineare, di cui vincola i pesi ϕ dei modelli di spiegazione generati ad essere dei valori Shapley approssimati. Come visto nelle sezioni precedenti, l'utilizzo di questi valori risulta essere l'unico modo per permettere a LIME di soddisfare le tre proprietà desiderabili di *missingness*, *local accuracy* e *consistency*.

Per ottenere i valori Shapley per ogni attributo di un modello di spiegazione generato da LIME, Kernel SHAP definisce in modo specifico la funzione di perdita $\mathcal{L}$, la misura di distanza $\pi_{x'}$ ed il termine di regolarizzazione Ω che devono essere utilizzati nella fase di ricerca e ottimizzazione mostrata nell'equazione 2.8.

Infatti, secondo il teorema dello Shapley Kernel alla base del metodo Kernel SHAP, per la tecnica LIME lineare le forme specifiche che $\pi_{x'}$, $\mathcal{L}$ e Ω devono assumere per permettere il soddisfacimento delle tre proprietà citate sono le seguenti:

$$\begin{aligned} \Omega(g) &= 0, \\ \pi_{x'}(z') &= \frac{(M-1)}{(M \; choose \; |z'|)|z'|(M-|z'|)}, \\ \mathcal{L}(f,g,\pi_{x'}) &= \sum_{z' \in \; Z} [f(h_x^{-1}(z')) - g(z')]^2 \pi_{x'}(z'), \end{aligned} \tag{2.9}$$

dove $|z'|$ è il numero di elementi non nulli in z'. Fondamentalmente, configurando il problema di ottimizzazione in tale modo, i pesi ϕ generati per il modello di spiegazione coincidono con un'approssimazione dei valori Shapley.

Entrando più nello specifico, data un'istanza x e un modello black-box f, Kernel SHAP genera un modello interpretabile g per la predizione $f(x)$ seguendo quattro passi principali:

1. Creazione del dataset di training per il modello di spiegazione lineare, composto da perturbazioni della versione semplificata z' dell'input originale. Per Kernel SHAP, le versioni perturbate dell'input possono essere considerate come coalizioni di feature, dove ogni coalizione $z'_k \in \{0,1\}^M$ con M equivalente alla grandezza massima della coalizione e K equivalente al numero di campioni perturbati richiesti per popolare il dataset. Come per LIME, i valori in tali coalizioni uguali a 1 rappresentano la presenza della feature, mentre quelli uguali a 0 rappresentano l'assenza;

2. Ottenimento della predizione del modello originale black-box per ogni coalizione z'_k, previa conversione del campione semplificato nello spazio dell'input originale $f(h_x(z'_k))$;

3. Calcolo del peso associato ad ogni coalizione z'_k tramite $\pi_{x'}(z')$;

4. Addestramento del modello di spiegazione lineare pesato rispetto i pesi definiti nel punto 3.

Gli step illustrati mostrano che l'iter di Kernel SHAP presenta delle forti analogie con il metodo LIME, il quale si basa sulla perturbazione dell'input originale e la successiva generazione di un dataset di training composto da diverse istanze perturbate. Tuttavia, le principali differenze tra i due metodi riguardano anche il processo di *sampling* delle coalizioni, ovvero le combinazioni di feature che vengono selezionate come input perturbati, e la misura π che assegna un peso differente alle coalizioni durante la fase di training del modello di spiegazione. In particolare, il dataset generato per il training del modello di spiegazione comprende coalizioni che rappresentano sottogruppi di attributi. Per ottenere la predizioni del modello originale associate alle coalizioni $z'_k \in \{0,1\}^M$, è necessario utilizzare una funzione di mapping che trasforma il campione perturbato nello spazio originale dell'input. Nel caso in cui una feature non sia presente nella coalizione, la funzione di mapping assegna ad essa un valore specifico o tutti i possibili valori che tale variabile può assumere in un dataset di background rappresentante il set di training originale. Tale comportamento risulta fondamentale per rappresentare una feature mancante senza eliminarla effettivamente dall'input, dato che la maggior parte dei modelli non può gestire l'assenza di attributi attesi.

Nel caso in cui siano disponibili più valori per gli attributi mancanti in una coalizione, Kernel SHAP, al momento del calcolo della predizione per quel campione specifico, restituirà la media delle predizioni ottenute dal campione in cui il valore delle variabili mancanti è sostituito con tutti gli altri possibili valori, mantenendo le altre variabili fisse.

Riguardo, invece, la differenza tra le misure di peso π utilizzate da LIME e da Kernel SHAP, la prima tecnica assegna un peso maggiore ai campioni più vicini all'istanza di interesse, portando le coalizioni con un alto numero di feature assenti ad avere un peso minore. Al contrario, Kernel SHAP utilizza una formula specifica che calcola il peso dei campioni sulla base del peso che la coalizione avrebbe durante la stima dei valori Shapley. In particolare, coalizioni piccole, che hanno quindi poche variabili presenti, e coalizioni grandi avranno pesi maggiori, in quanto è possibile apprendere maggiormente sull'effetto delle feature individuali se queste sono viste in isolamento (come nel caso delle coalizioni con una sola feature), e allo stesso modo, l'esclusione di una feature in una coalizione composta da tutte le altre permette di apprendere molto sull'effetto dell'output senza quella feature.

Di conseguenza, il *sampling* delle coalizioni che entrano a far parte del dataset di training è effettuato adottando una strategia che tiene conto del fatto che le coalizioni più piccole e più grandi rappresentano la maggior parte del peso e che la loro inclusione migliora l'accuratezza delle stime dei valori Shapley. Quindi, invece di campionare casualmente, una porzione del budget di campionamento K viene utilizzata per includere queste coalizioni ad alto peso. Si inizia quindi con tutte le possibili coalizioni con 1 e $M-1$ feature, che corrispondono a un totale di $2*M$ coalizioni. Nel caso in cui il budget non sia esaurito (budget attuale $= K - 2M$), si procede includendo anche coalizioni con 2 feature, M-2 feature e così via.

2.3.2 Tree SHAP

L'algoritmo Tree SHAP è un'altra delle tecniche proposte da Lundberg e Lee [10] come implementazione del framework SHAP. Nello specifico, si presenta come un metodo non agnostico per l'approssimazione dei valori Shapley per le feature coinvolte nel calcolo delle predizioni di modelli basati su alberi. Sfruttando la conoscenza dell'architettura del modello, l'algoritmo è ottimizzato al fine di migliorare l'efficienza e la precisione delle stime dei valori. Inoltre, Tree SHAP ha una complessità temporale polinomiale di solo

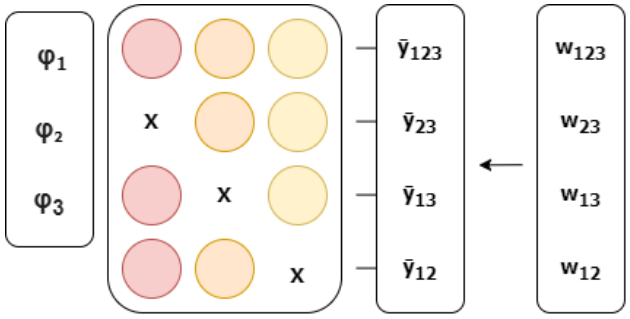

Figura 2.3: Esempio semplificato che rappresenta le correlazioni tra dataset di coalizioni (di massimo tre feature), le predizioni associate, i pesi e le importanze. A sinistra sono definiti i pesi ϕ dell'Explanation Model, i quali coincidono con gli Shapley values delle variabili. L'insieme più ampio rappresenta il dataset contenente le coalizioni, alcune con delle feature assenti, e per ciascuna di esse è calcolata la relativa previsione $\overline{y}$ come media di tutte le previsioni relative alla coalizione, tenendo conto delle diverse combinazioni delle caratteristiche mancanti. Durante la fase di addestramento, ogni coalizione avrà un suo peso associato w.

$O(TLD^2)$, dove T è il numero di alberi nel modello, L è il numero massimo di foglie in ogni albero e D è la profondità massima dell'albero. Il parametro M rappresenta invece il numero di feature dell'input.

Al fine di stimare i valori SHAP delle feature di un input x tramite la formula per i valori shapley, Tree SHAP sfrutta le funzioni di *conditional expectation* $E[X_j|X\_j](\hat{f}(x)|x_j)$ relative all'algoritmo da spiegare.

Alla base di tale metodo c'è l'intuizione secondo cui, per calcolare una predizione attesa per un'istanza di input x e un sottoinsieme di feature S utilizzando un singolo albero decisionale, se si condiziona l'*expectation* della predizione rispetto tutte le feature (quindi il subset S contiene tutte le possibili variabili) allora la predizione attesa è quella contenuta nella foglia in cui l'istanza x cade a seguito del corretto percorso decisionale. Se invece si condiziona la predizione sull'assenza di tutte le feature, quindi con $S = \emptyset$, allora è possibile utilizzare la media ponderata delle predizioni di tutte le foglie

dell'albero. Nel caso in cui S contenga un effettivo sottoinsieme delle caratteristiche, si possono ignorare le predizioni delle foglie irraggiungibili, ovvero quelle il quale il percorso decisionale tramite cui possono essere raggiunte non soddisfa i valori in x_s. Dai nodi terminali rimanenti, è possibile calcolare la media delle predizioni contenute, ponderata per il numero di esempi di addestramento in quel nodo e tale media fungerà da predizione attesa per l'input x dato S. La procedura esposta va applicata per ogni possibile sottoinsieme S delle feature dell'input x.

Il metodo Tree SHAP è strutturato in modo da calcolare questi valori in tempo polinomiale invece che esponenziale. Per fare ciò, intuitivamente, fa scorrere simultaneamente tutti i possibili sottoinsiemi di S nell'albero decisionale di cui spiegare la predizione su x. Per ogni nodo decisionale si tiene traccia del numero di sottoinsiemi che riescono ad arrivarvi tramite un percorso decisionale precedente. Ad esempio, se la prima suddivisione in un albero riguarda la caratteristica x_1, allora tutti i sottoinsiemi che contengono tale feature andranno in un nodo specifico. I sottoinsiemi che non contengono la caratteristica x_1 sono ripartiti invece in tutti i nodi del livello con un peso ridotto.

Tale computazione può essere estesa a più alberi grazie alla proprietà di additività dei valori shapley, la qualr fa sì che il contributo additivo di una feature in un algoritmo ensemble sia la media ponderata dei valori di Shapley in relazione ai singoli alberi.

E' interessante notare che Tree SHAP si differenzia da Kernel SHAP in quanto calcola direttamente i valori Shapley piuttosto che estrarli da un modello lineare addestrato. I valori Shapley calcolati possono comunque essere inclusi come pesi di un modello lineare, il quale risulterebbe conforme a quello caratteristico generato dagli additive feature attribution methods.

2.4 Modalità di applicazione

Il lavoro della presente tesi verte sull'applicazione e confronto dei risultati delle metodologie LIME e SHAP applicate ai classificatori binari presentati da Zanellati et al. [19]. I tre modelli proposti sono di tipo random forest, categorical embeddings neural network (CE) e feature tokenizer transformer (FTT), ed il loro obbiettivo è quello di prevedere se uno studente avrà un basso rendimento scolastico in seconda superiore sulla base dei risultati dei test INVALSI ottenuti in quinta elementare e di altri fattori culturali, socio-economici e demografici. Nello specifico, per ogni modello, sono state generate spiegazioni utilizzando metodo LIME lineare e le tecniche Kernel SHAP (per la categorical embeddings neural network ed il transformer) e Tree SHAP (per il random forest). La teoria e le intuizioni di questi metodi utilizzati sono state esposte nelle sezioni precedenti 2.2 e 2.3.

I metodi in oggetto sono stati applicati ad ogni istanza dell'insieme dei dati di test preso in considerazione, e per ognuna di esse sono stati generati i valori SHAP e LIME di ogni feature utilizzata. Successivamente, tali valori sono stati aggregati al fine di ottenere le importanze globali delle feature in relazione ai modelli.

Nelle seguenti sottosezioni saranno spiegati nel dettaglio il dataset su cui son state generate le spiegazioni e le modalità di applicazione di LIME e SHAP ai tre modelli presi in analisi.

2.4.1 Dataset

Gli esperimenti sono stati effettuati sullo stesso dataset di test definito nella pubblicazione di riferimento di Zanellati et al. [19]. Il dataset è composto da 232.033 campioni, rappresentanti gli studenti italiani della coorte 2013/2014 che hanno effettuato i test INVALSI in quinta elementare e in seconda superiore. Le feature del dataset sono 34, sia numeriche che categoriche e riguardano il contesto socio-economico, demografico e l'andamento scolastico dello studente, quest'ultimo dedotto principalmente dal voto medio

di italiano e di matematica, da un encoding dei risultati dei test INVALSI di matematica di quinta elementare e dal punteggio totale ottenuto. Nello specifico, l'estrazione delle feature dai dati relativi agli INVALSI è stata effettuata classificando le domande del test in termini di processi, macro processi e aree (tabella 2.1), assegnando ad ognuno di questi gruppi la percentuale di risposte corrette riscontrate.

Aree	
NU	Numeri
SF	Spazi e figure
DF	Dati e previsioni
RF	Relazioni e funzioni
Processi	
P1	Conoscere e padroneggiare contenuti specifici matematici
P2	Conoscere e utilizzare algoritmi e procedure
P3	Conoscere le diverse forme di rappresentazione e passare agilmente da una all'altra
P4	Risolvere problemi utilizzando strategie in campi diversi
P5	Riconoscere la natura misurabile di oggetti e fenomeni in diversi contesti e misurare quantità
P6	Acquisire progressivamente le forme tipiche del pensiero matematico
P7	Utilizzare strumenti, modelli e rappresentazioni per il trattamento quantitativo dell'informazione in ambito scientifico, tecnologico , economico e sociale
P8	Riconoscere forme nello spazio e utilizzarle per la risoluzione di problemi
Macroprocessi	
MP1	Formulare
MP2	Interpretare
MP3	Utilizzare

Tabella 2.1: Encoding dei quesiti nel test INVALSI di matematica

Le altre variabili, di taglio culturale, demografico e sociale, sono tutte categoriche e includono informazioni come il genere, il luogo di nascita, la cittadinanza, la regione di appartenenza e il titolo di studio e l'occupazione dei genitori. Inoltre, tra le feature è inclusa anche la misura numerica ESC, specifica per le prove INVALSI, che definisce lo status sociale, economico e culturale delle famiglie degli studenti che partecipano alle prove. Ad ogni campione del dataset è associata la classe target binaria relativa al basso rendimento scolastico, dove il valore 1 è assegnato agli studenti che, in seconda superiore, hanno ottenuto un punteggio minore o uguale di 2 su 5 ai test INVALSI di matematica.

2.4.2 LIME

Dato l'agnosticismo del metodo, LIME è stato applicato allo stesso modo per la generazione di spiegazioni per i tre modelli presi in esame. Come esposto precedentemente, le spiegazioni che LIME fornisce per una singola predizione sono sotto forma di effetti additivi degli attributi coinvolti nel calcolo dell'output. Tutti i modelli di riferimento lavorano sullo stesso set di dati che sono di tipo tabulare, il che richiede una gestione specifica relativa all'utilizzo degli input stessi e delle perturbazioni generate per popolare il dataset di training dei modelli di spiegazione.

In particolare, sono stati configurati in modo specifico gli elementi partecipanti al problema di ottimizzazione 2.8, risolto da LIME per la generazione di un modello g che approssima il comportamento del modello originale f rispetto un input x. Riguardo la classe G dei modelli interpretabili, è stata scelta la regressione di tipo Ridge appartenente all'insieme dei modelli lineari sparsi. Riguardo la scelta di $\pi_x(z)$, ovvero la misura di prossimità che assegna il peso dei campioni perturbati durante l'addestramento del modello interpretabile, questa è calcolata da un kernel esponenziale (equazione 2.10) applicato alle distanze euclidee tra i campioni e l'istanza di interesse

da spiegare.

$$\sqrt{\exp\left(\frac{-d^2}{\mathrm{k}^2}\right)} \tag{2.10}$$

dove k è grandezza del kernel che determina la dimensione del vicinato dell'istanza di interesse. Più il valore di k è alto, maggiore sarà l'influenza associata anche a istanze perturbate distanti. Per questo studio, la grandezza scelta per il kernel è stata di $\sqrt{|M|} + 0.75$, con $|M|$ equivalente al numero di feature del dataset.

Riguardo il sampling per la definizione del dataset delle perturbazioni dell'istanza di interesse di cui si intende spiegare la predizione, gli attributi numerici sono perturbati campionando da una distribuzione normale standard e applicando successivamente l'operazione inversa di centratura e ridimensionamento rispetto alle medie e alle deviazioni standard presenti nei dati di training. Per le feature categoriche, invece, la perturbazione avviene campionando secondo la distribuzione di addestramento. Un altro aspetto importante del metodo LIME utilizzato nei modelli oggetto di studio è che le spiegazioni generate non attribuiscono un effetto a una singola feature nella sua accezione più generale, ma piuttosto alla coppia feature-valore. Inoltre, le feature continue sono state discretizzate in quartili al fine di semplificarne la comprensione e la leggibilità nella spiegazione in output. Dato che i modelli originali restituiscono le predizioni come due probabilità, una per la classe relativa al basso rendimento e l'altra per la classe complementare, per ogni predizione e per ognuno dei due risultati è addestrato un modello di spiegazione separato.

Sulla base di queste scelte implementative e configurazioni per l'applicazione della tecnica LIME, sono state successivamente calcolate le spiegazioni relative alle predizione di ogni istanza del dataset di test. Infine, per ottenere una misura delle importanze globali delle variabili nelle scelte dei modelli, sono stati aggregati, per ogni attributo, gli effetti associati rispetto ogni predizione calcolata, seguendo la formula 2.1.

L'unica differenza relativa all'applicazione di tale metodo sui modelli presi

in esame riguarda la modalità con cui le variabili categoriche sono state gestite per la generazione di spiegazioni. La categorical embeddings neural network e il feature tokenizer transformer sono stati addestrati su variabili categoriche originali, di cui non è stato effettuato nessun tipo di encoding in quanto tali modelli sono caratterizzati proprio dalla capacità di gestire le categorie senza un'elaborazione iniziale. In questo caso, anche le spiegazioni son state generate considerando lo stesso input del modello, senza nessuna elaborazione aggiuntivo delle categorie.

Per quanto riguarda il modello random forest, invece, è stato necessario seguire una procedura differente. Le variabili categoriche utilizzate durante l'addestramento di questo modello sono infatti rappresentate in formato one-hot encoded, mentre le spiegazioni LIME necessitano di essere generate su una versione delle istanze in cui le variabili categoriche lo siano in formato standard. Questo perchè, durante l'attività di sampling per la creazione del dataset di training per il modello di spiegazione, nel caso in cui le variabili categoriche one-hot encoded fossero considerate come variabili numeriche, verrebbe calcolata la loro media e deviazione standard, mentre nel caso in cui, invece, fossero considerate come variabili categoriche, verrebbe effettuato il loro sampling mantenendo la distribuzione presente nel dataset originale. In qualsiasi caso, si perderebbe la rappresentazione delle feature come array binario, con un solo valore non nullo, che rappresenta la presenza o l'assenza di una specifica categoria.

Pertanto, per il random forest, le spiegazioni LIME sono generate sulle feature nella versione originale non one-hot encoded. È importante sottolineare che le predizioni relative ai campioni del dataset di supporto sono state comunque eseguite su una versione dei campioni one-hot-encoded, poiché è l'unico formato accettato dal modello random forest da spiegare.

2.4.3 SHAP

In relazione alla metodologia SHAP applicata ai casi di studio presi in considerazione, è stata utilizzata la tecnica Kernel SHAP per generare spiega-

zioni per i modelli categorical embeddings neural network e feature tokenizer transformer, e la tecnica Tree SHAP per il random forest. Per Kernel SHAP, è stato seguito l'approccio teorico descritto nella sezione 2.3.1: la spiegazione di una predizione è generata come una regressione lineare pesata, i cui pesi assegnati alle coalizioni del dataset di appoggio sono calcolati come indicato nella teoria. Anche la gestione del sampling per il dataset delle coalizioni è eseguita seguendo le specifiche teoriche. Per simulare le feature mancanti è stato considerato un dataset di background di 100 campioni estratti randomicamente dal dataset di training. Lo stesso dataset di background è stato utilizzato sia per generare spiegazioni per la categorical embeddings neural network e il feature tokenizer transformer.

Per quanto riguarda l'analisi effettuata sul random forest, è stata applicata la tecnica Tree SHAP. Anche in questo caso, si è seguito l'approccio teorico, ovvero l'uso di una distribuzione di background basata sui campioni memorizzati nel modello stesso. È importante definire che, a differenza degli altri due modelli CE e FTT, i quali hanno meccanismi interni per la gestione delle variabili categoriche, il random forest è stato addestrato con variabili categoriche codificate tramite one-hot encoding. In questo caso, Tree SHAP calcola separatamente i valori SHAP per ogni variabile *dummy* relativa ad una feature e successivamente questi sono stati sommati per ottenere l'effetto complessivo della variabile variabile categorica originale. Questo è possibile grazie alle proprietà dei valori Shapley.

Per tutti e tre i modelli, dopo il calcolo dei valori SHAP per le feature coinvolte nelle predizioni di tutte le istanze del test set, si è proceduto con l'aggregazione dei risultati per ottenere le importanze globali delle variabili, sotto forma di media dei valori assoluti degli effetti associati (formula 2.1).

Capitolo 3

Setup degli esperimenti

Dato che gli esperimenti condotti consistevano nella generazione massiva di valori SHAP e LIME su ogni istanza del dataset di test e per ognuno dei tre modelli da analizzare, l'impegno temporale richiesto non risultava indifferente. Pertanto le computazioni sono state eseguite su tre diverse piattaforme al fine di parallelizzare la loro esecuzione e ridurre i tempi necessari per l'ottenimento dei risultati. Nello specifico, la generazione di spiegazioni LIME e SHAP per il random forest è stata svolta su un pc Asus con un processore Intel i7 di 7^a generazione, 16 giga di RAM e una GPU Nvidia Geforce GTX1050 con `CUDA 11.6.0`. L'applicazione del metodo LIME sulla category embeddings neural network (CE) e sul feature tokenizer transformer (FTT), invece, è stata eseguita utilizzando un Google colab notebook in un ambiente con GPU dedicata. Infine, i valori SHAP, sempre su CE e FTT, sono stati generati su un cluster di GPU con Nvidia Driver 470 e `CUDA 11.40`, reso disponibile dal dipartimento di informatica dell'Università di Bologna.

Il lavoro presentato è stato interamente sviluppato in `Python 3.8.10` e sono state utilizzate diverse librerie a supporto. L'addestramento del random forest e le attività di processamento dei dati sono state sviluppate tramite le funzionalità di `Scikit-learn 1.12.1`, mentre per l'addestramento e la gestione dei modelli CE e FTT è stata utilizzata `pytorch tabular 0.7`, libreria focalizzata sul deep learning per dati tabulari ereditata dal lavoro di

Zanellati et al. [19]. I metodi SHAP e LIME sono invece stati applicati tramite le implementazioni degli autori stessi [14, 11, 10] proposte nelle librerie `lime 0.2.0` e `shap 0.41.0`.

Il calcolo dei valori SHAP e LIME per il random forest è stato effettuato su un Jupyter notebook locale, mentre computazioni LIME per CE e FTT su un notebook remoto su Colab. L'esecuzione del metodo SHAP sulle reti neurali, invece, è stato gestito da uno script Python dedicato.

Tutti gli esperimenti sono stati condotti sul dataset originale di test utilizzato nel lavoro che ha proposto i modelli in esame [19]. Anche il dataset di training è stato considerato per fornire statistiche e sottoinsiemi utili ai fini della generazione di spiegazioni. Di tali dataset sono state considerate come numeriche tutte le feature codificate dai risultati dei test INVALSI, insieme alla misura ESC ed ai voti medi di italiano e matematica, mentre le restanti categoriche.

Prima di procedere con la generazione delle spiegazioni, è stato necessario addestrare i modelli random forest, CE e FTT in quanto questi non risultavano essere disponibili e già addestrati. Per fare ciò, è stato riprodotto il loro addestramento utilizzando le stesse configurazioni degli esperimenti definiti nella pubblicazione in cui sono stati proposti. Mentre, per la creazione del modello CE e FTT non è stato richiesto un processamento aggiuntivo dei dati originali, il training del random forest ha richiesto il one-hot encoding delle variabili categoriche e l'applicazione di un undersampling (in quanto la classe relativa al basso rendimento ha un numero minore di campioni rispetto l'altra).

Una volta addestrati e salvati i modelli è stato possibile procedere con la computazione delle spiegazioni LIME e SHAP, ovvero i valori rappresentanti gli effetti delle feature in una predizione, per ogni istanza del dataset di test e per ogni modello.

3.1 LIME

L'applicazione del metodo LIME per generare spiegazioni delle predizioni rispetto i tre modelli è stata gestita utilizzando la classe `LimeTabularExplainer` della libreria `lime`. Tale classe include funzioni utili per la generazioni di valori LIME per modelli che lavorano specificatamente con dati tabulari, come quelli dei dataset utilizzato per lo studio.

L'utilizzo della classe citata ha richiesto un processamento aggiuntivo dei dati di training e test. E' stata infatti necessaria l'applicazione del numerical encoding alle variabili categoriche per rappresentare numericamente le categorie testuali, in quanto il tipo di modello lineare da generare come spiegazione non gestiva input testuali.

Essendo LIME un metodo agnostico, la sua implementazione non accetta esplicitamente il modello da interpretare, bensì richiede la definizione di una funzione di predizione da poter utilizzare in modo black-box per ottenere le predizioni necessarie per il dataset dei campioni perturbati. In questo caso, per CE e FTT, la funzione definita, presi input un numero arbitrario di campioni, effettua il decoding delle variabili categoriche nella loro versione originale e utilizza il modello CE o FTT per ottenere le predizioni da restituire. Essendo il problema di classificazione, l'output restituito è composto dalle probabilità per ognuna delle possibili classi in output. La stessa funzione è definita per il random forest, con la differenza che, dato che il modello è addestrato su variabili categoriche in formato one-hot, le istanze originali input della funzione devono essere processate con il one-hot encoding prima di essere passate al modello per ottenere le predizioni sotto forma di probabilità delle classi target.

Una volta definite le funzioni di predizione per i tre modelli, è stato possibile procedere con la fase di computazione effettiva delle spiegazioni. I valori LIME sono stati calcolati tramite l'oggetto `LimeTabularexplainer` per ogni istanza del dataset di test e per ogni modello: data la grande mole di dati (232'033 istanze) e considerando che per ogni campione è generato un dataset di appoggio e addestrato un modello lineare pesato per ogni possibile classe

target, i calcoli sono stati eseguiti su gruppi di 100 istanze consecutive alla volta, in modo da memorizzare i risultati in singoli files e poterli caricarli successivamente al momento delle analisi. Ciò ha permesso di prevenire l'eventuale perdita dei valori calcolati nel caso di arresto involontario o voluto del processo attivo.

Per snellire il processo, sono state generate solo le spiegazioni relative alla classe del basso rendimento. Dato il problema di natura binaria, gli effetti degli attributi delle feature rispetto una predizione risultano essere complementari agli effetti sull'altra classe.

Al termine delle computazioni, sono stati caricati tutti i valori LIME per le predizioni su tutte le istanze del dataset e sono state calcolate le importanze globali come media dei valori assoluti dei pesi per ogni variabile.

La figura 3.1 presenta l'intuizione relativa alle relazioni tra le diverse componenti dell'esperimento, considerando il modello come elemento istanziabile dalla CE, FTT o random forest.

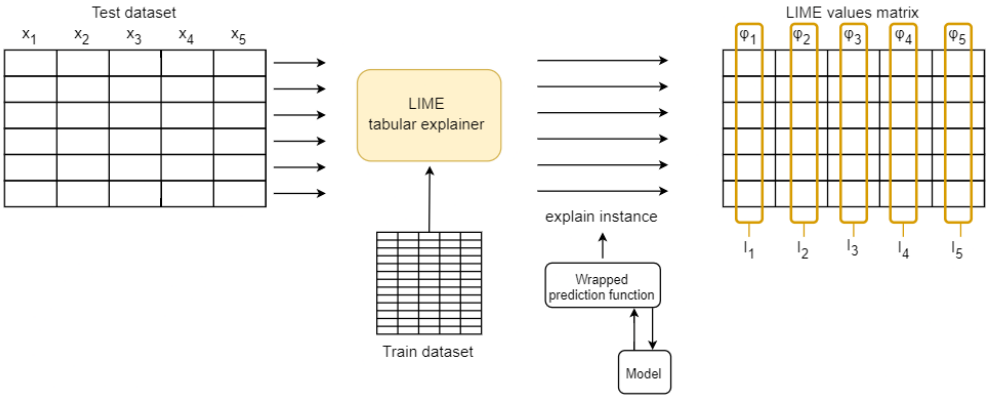

Figura 3.1: Intuizione esperimento metodo LIME

3.2 SHAP

Gli esperimenti relativi all'applicazione del metodo SHAP sui modelli oggetti di studio sono stati effettuati utilizzando le classi della libreria `shap`

`KernelExplainer`, per CE e FTT, e `TreeExplainer` per il random forest.

A differenza di quando esposto per LIME, le implementazioni presentate non richiedono che le variabili delle istanze in input siano necessariamente in formato numerico. Per i test relativi a CE e FTT, quindi, è stato possibile utilizzare la versione originale delle istanze, dato che tali modelli sono stati addestrati su variabili categoriche testuali.

Prima di procedere con la generazione dei valori SHAP per CE e FTT, è stato definito il dataset di background da utilizzare per integrare le feature mancanti nel metodo. Il dataset è stato ottenuto facendo un campionamento randomico di 100 istanze dal dataset di training originale.

Come per LIME, essendo Kernel SHAP agnostico, è stato necessario definire una funzione wrapper per ottenere le predizioni dei modelli, fondamentali per la generazione del dataset di appoggio su cui è eseguito il training dei modelli lineari di spiegazione. Anche in questo caso, la funzione è definita in modo tale da restituire, data una matrice di campioni, una matrice di predizioni ottenibili dai modelli stessi, composte dalle probabilità per ogni classe target del modello.

Una volta configurate le funzioni di predizione da passare all'oggetto `KernelSHAP`, sono stati calcolati i valori SHAP per ogni istanza del dataset di test. Anche in questo caso le istanze sono state suddivide in gruppi da 100 al fine di memorizzare i risultati intermedi.

Riguardo gli esperimenti relativi al calcolo dei valori SHAP per il random forest, invece, è stato eseguito un processamento iniziale del dataset di test in modo da ottenere le versioni one-hot encoded delle variabili categoriche. Ciò è risultato necessario data la scelta di calcolare i valori SHAP separatamente per ogni variabili *dummy* relativa ad una feature categorica.

Per il random forest, il metodo SHAP è stato applicato tramite la sua implementazione esposta dalla classe `TreeSHAP`. Tale explainer risulta essere strettamente legato al modello e non richiede dunque il passaggio di una funzione wrapper per l'ottenimento delle predizioni. In questo caso, infatti, è quindi possibile utilizzare direttamente il modello random forest. Una volta

configurato l'explainer, sono stati generati i valori SHAP relativi alle feature di ogni istanza del dataset di test. Data la presenta di variabili one-hot encoded per cui sono stati calcolati i valori SHAP, gli effetti relativi alle variabili dummy sono stati sommati al fine di ottenere il valore SHAP della variabile categorica originale.

Terminate le computazioni e le eventuali aggregazioni locali (per il random forest), per tutti i modelli sono stati caricati i valori SHAP calcolati e successivamente, per ogni feature, i valori SHAP rispetto tutte le istanze sono stati aggregati per ottenere la misura di importanza globale rispetto il modello. Tale importanza si mostra come media dei valori assoluti dei valori SHAP.

Le figure 3.2 e 3.3 presentano le intuizione relative alle relazioni tra le diverse componenti dell'esperimento con explainer di tipo SHAP. Nel grafico 3.2 è possibile considerare il componente "modello" come elemento istanziabile dalla CE o FTT.

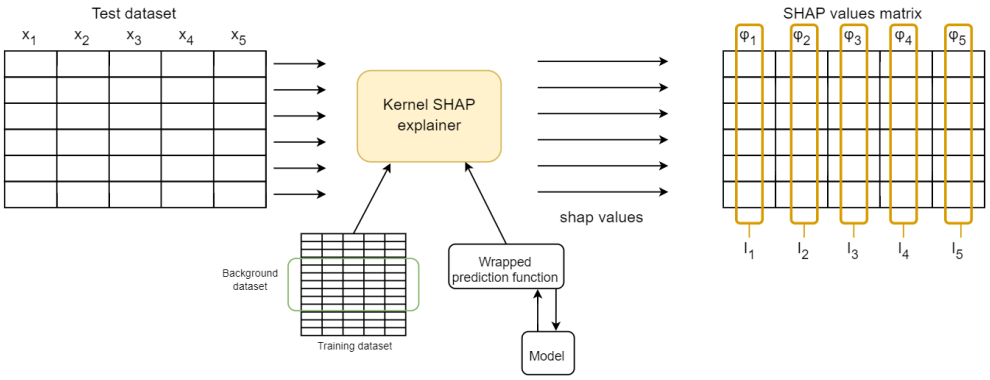

Figura 3.2: Intuizione esperimento metodo Kernel SHAP

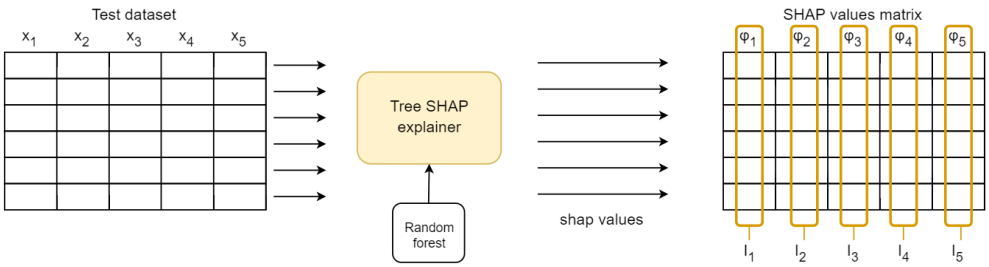

Figura 3.3: Intuizione esperimento metodo Tree SHAP

Capitolo 4

Risultati sperimentali

Gli esperimenti condotti hanno coinvolto, per ognuno dei tre modelli considerati, ovvero il random forest, categorical embeddings neural network e feature tokenizer transformer, la generazione massiva di valori SHAP e LIME per ciascuna istanza del dataset di test, costituito da 232.033 campioni. Quindi, nel complesso sono state effettuate separatamente sei generazioni di spiegazioni, coprendo le combinazioni LIME su Random Forest, LIME su CE, LIME su FTT e SHAP su random forest, SHAP su CE e SHAP su FTT. Per ognuna delle combinazioni sono stati successivamente aggregati tutti i valori calcolati per ogni istanza al fine di ottenere le importanze globali delle feature per ogni modello. Come esposto nella sezione 2.4, la metodologia di aggregazione scelta è la media dei valori, SHAP o LIME, assoluti. Tale metodo risulta sensato considerando come feature più importanti quelle che hanno effetto più forte, sia in positivo che negativo, sull'output.

Nelle seguenti sezioni saranno esposti i risultati degli esperimenti eseguiti. Le importanze globali, calcolate per ogni feature, saranno rappresentate graficamente tramite grafici a barre contenenti le 15 feature con le importanze globali più alte, colorate differentemente a seconda del tipo di informazione che comunicano sugli studenti. Riguardo ciò, è importante specificare che tali tipologie di informazioni sono le seguenti:

- anagrafiche: mese e luogo di nascita, cittadinanza;

- demografiche: genere;
- geografiche: area geografica, codice regionale e codice ISTAT per la provincia di residenza dello studente;
- socioeconomico - culturali: indicatore ESCS, luoghi di nascita, titoli di studio e professioni dei genitori;
- scolastiche: frequenza scuola materna, scuola iniziata regolarmente, medie dei voti di italiano e matematica;
- relative ai test INVALSI: punteggio totale del test e tutte le feature definite nella tabella 2.1.

Inoltre, saranno anche utilizzati dei grafici di tipo *beeswarm*, al fine di presentare un riassunto intuitivo di come le principali feature del dataset influenzino uno specifico output del modello, in questo caso quello relativo alla classe positiva per il basso rendimento scolastico, sfruttando l'importanza SHAP associata. Questo tipo di grafico mostra, per le feature più importanti, una riga di punti, ognuno rappresentante un'istanza del dataset, dove la posizione orizzontale del punto rappresenta il valore SHAP della feature corrispondente per quella particolare istanza. In questo modo, il grafico permette di individuare le feature più importanti per il modello, grazie alla loro posizione rispetto agli altri punti. Inoltre, per le variabili numeriche è anche possibile, in base al colore del punto rappresentante l'istanza, avere un'idea generale del valore associato.

La prima sezione 4.1 sarà relativa alle importanze globali calcolate a partire dai valori SHAP e LIME per il modello categorical embeddings neural network. Le sezioni 4.2 e 4.3, invece, esporranno i risultati rispettivamente per il feature tokenizer transformer e il random forest. L'ultima sezione, 4.4 conterrà invece una discussione complessiva dei risultati osservati nelle sezioni precedenti.

4.1 Categorical embeddings neural network

Per il modello in esame di tipo categorical embeddings neural network, le importanze globali ottenute aggregando i valori LIME calcolati per ogni feature sono visualizzate nel grafico 4.1.

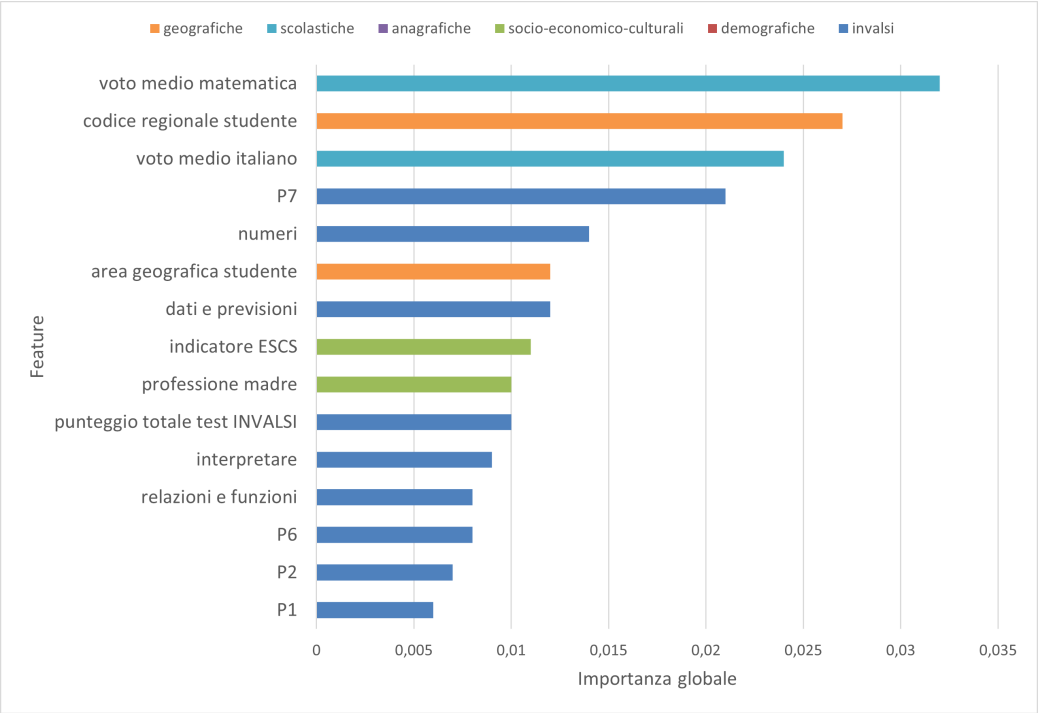

Figura 4.1: Importanze globali ottenute dai valori LIME per la categorical embeddings neural network

Analizzando il grafico si può notare che i valori complessivi delle importanze assegnate non sono molto elevati, poiché il valore massimo è solo dello 0,03, associato alla feature che rappresenta il voto medio di matematica dello studente. È inoltre interessante osservare che tra le 15 feature presentate nel grafico, le quali sono le migliori a livello di importanza, prevale la presenza di variabili relative al contesto didattico, sia generale che specifico per i test INVALSI presi in considerazione. Il grafico mostrato nella figura 4.2 rappresenta in dettaglio la distribuzione degli effetti SHAP associati a ciascuna feature per ogni istanza del dataset di test. Dai risultati ottenuti, è possibile

notare che per le variabili numeriche relative al contesto scolastico, si conferma una logica intuitiva che suggerisce come i valori bassi di tali variabili (che generalmente rappresentano voti e punteggi) abbiano un impatto positivo (in misura di punti in probabilità) e significativo sulla predizione della classe di basso rendimento, ovvero quella che indica che lo studente probabilmente non avrà un buon rendimento scolastico.

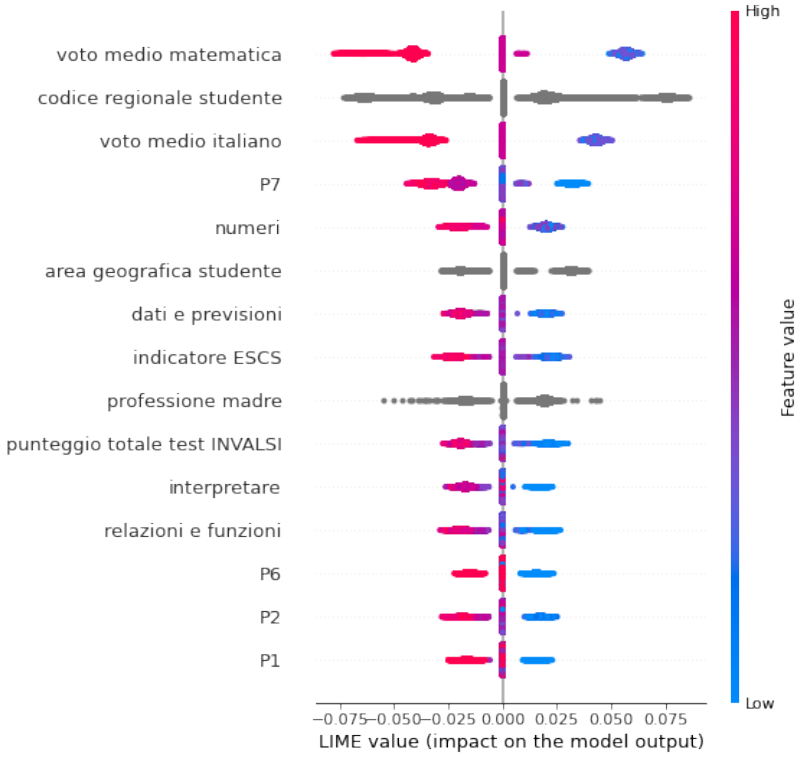

Figura 4.2: Distribuzione valori LIME per la predizione della classe positiva per il basso rendimento per la CE

Riguardo i risultati ottenuti applicando il metodo SHAP, invece, la figura 4.3 mostra le feature con le migliori importanze globali aggregate.

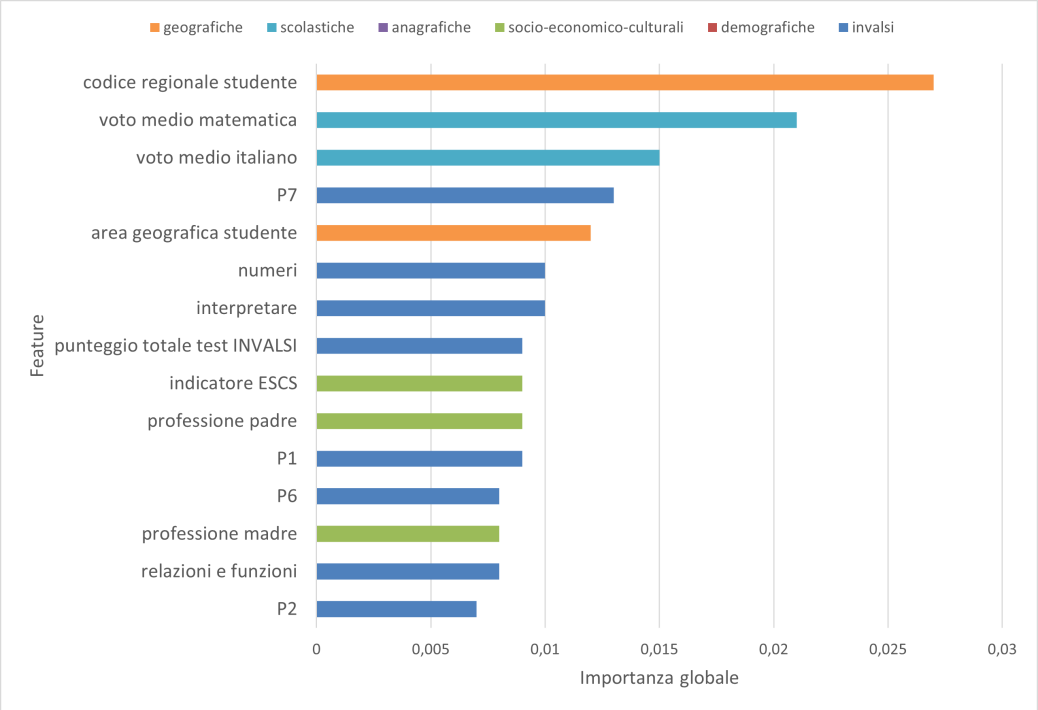

Figura 4.3: Importanze globali ottenute dai valori SHAP per la categorical embeddings neural network

Anche in questo caso, come per le importanze calcolate con LIME, queste non risultano avere dei valori particolarmente elevati. E' presente però una coerenza tra i due risultati, in quanto le tre feature più importanti sono in entrambi i casi il codice regionale dello studente ed i voti medi di italiano e matematica. Nel grafico si notano molti attributi relativi all'ambito didattico, ma anche tre associati al contesto socio-economico dello studente. Tuttavia, come evidenziato anche nel grafico 4.4, questi attributi non hanno un impatto significativo sulla predizione finale influenzando l'output di non più di cinque punti percentuali. Nonostante la presenza di questi attributi, si osserva comunque una netta differenza tra le prime tre feature e le altre, come osservabile anche nel grafico di tipo *beeswarm*.

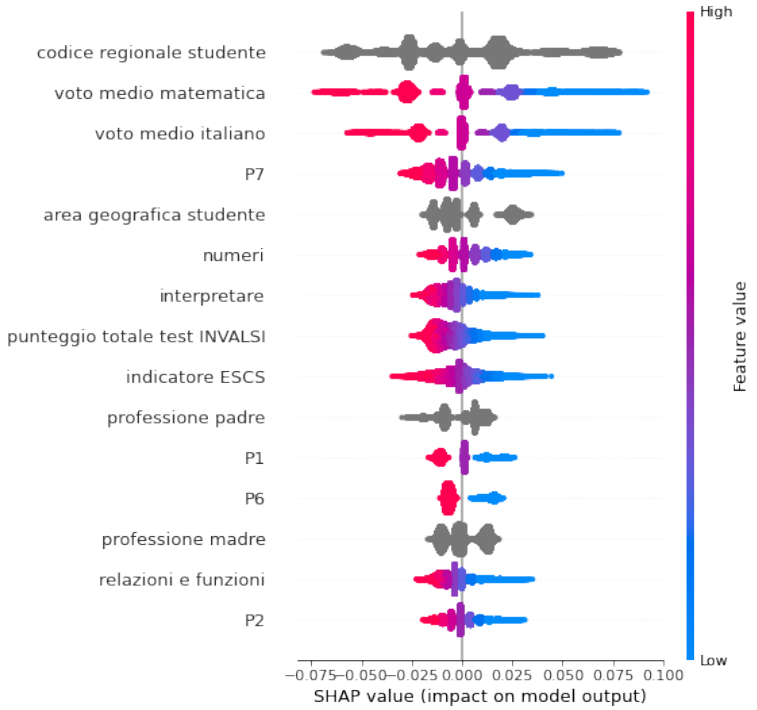

Figura 4.4: Distribuzione valori SHAP per la predizione della classe positiva per il basso rendimento per la CE

4.2 Feature tokenizer transformer

Relativamente al modello in esame di tipo feature tokenizer transformer, la figura 4.5 mostra le importanze globali ottenute aggregando i valori LIME calcolati per ogni feature su ogni istanza del dataset di riferimento. E' possibile osservare che, rispetto le importanze per la CE, la magnitudo risulta essere generalmente più alta, soprattutto per le tre variabili ritenute più importanti. Anche in questo caso tra esse risulta il voto di matematica e l'area geografica in cui lo studente vive, seguite dal genere dello studente, anch'esso con un'importanza mediamente alta associata. Come osservato anche per i risultati relativi al modello CE, in generale, il contesto scolastico e relativo agli INVALSI rimane il più significativo a livello di importanze predittive.

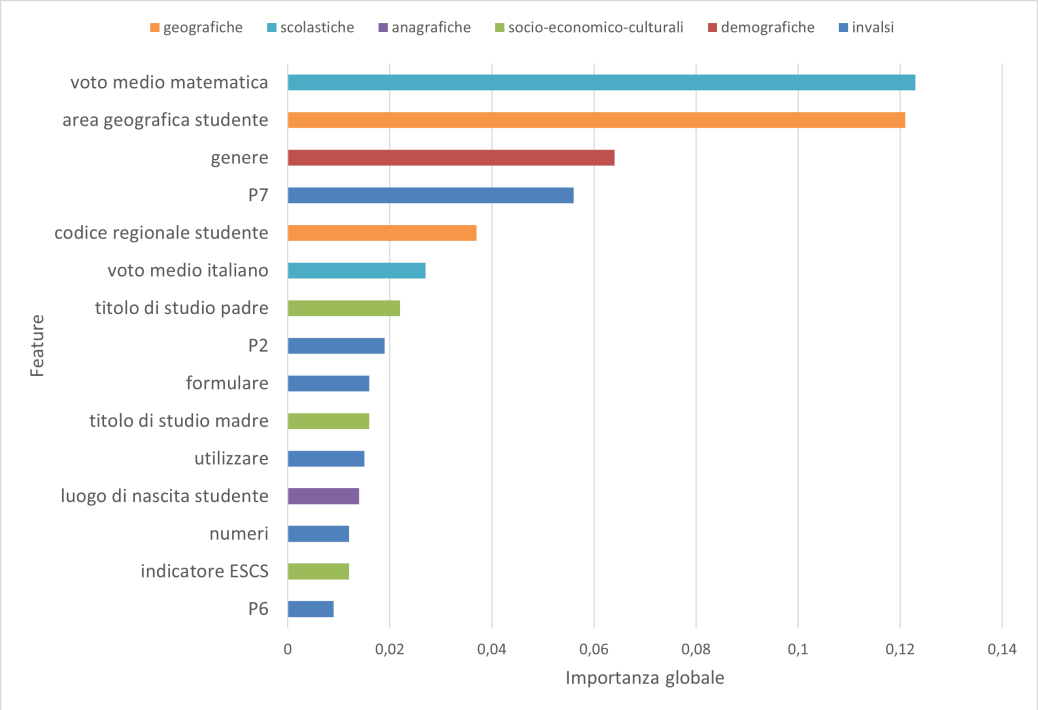

Figura 4.5: Importanze globali ottenute dai valori LIME per il Feature tokenizer transformer

Il grafico *beeswarm* 4.6 evidenzia una netta separazione tra gli effetti della variabile relativa al voto di matematica, ovvero la migliore in termini di importanza globale. Tale feature, infatti, risulta avere generalmente un effetto maggiore di 20 punti percentuali sull'output finale, con una chiara distinzione del segno dell'effetto in base al valore della variabile (voto alto o basso).

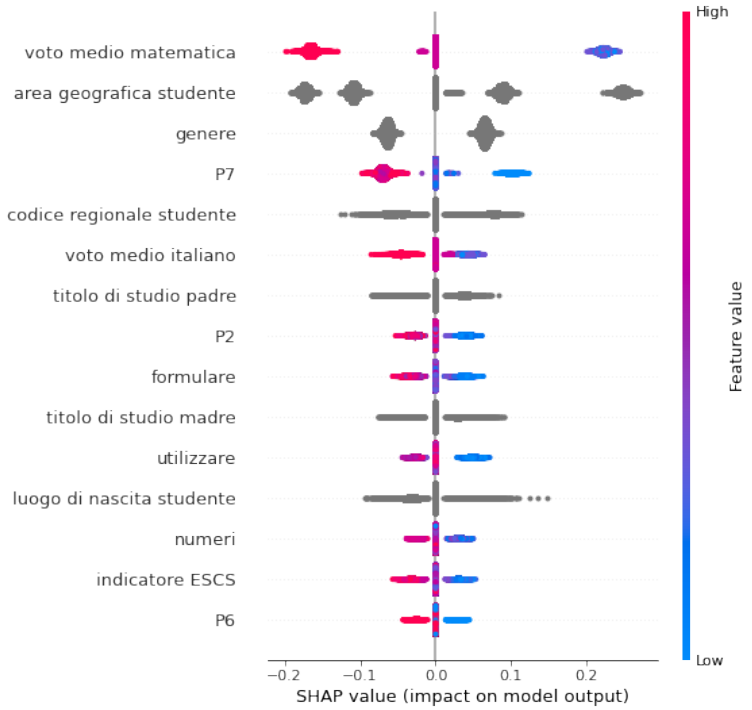

Figura 4.6: Distribuzione valori LIME rispetto le feature per la predizione della classe positiva per il basso rendimento per il Feature tokenizer transformer

I risultati derivanti dal calcolo dei valori SHAP sono osservabili nei grafici 4.7 e 4.8. Il grafico a barre mostra le importanze globali associate alle migliori variabili, tra le quali è possibile notare i già osservati attributi relativi alla locazione geografica, al contesto didattico e al genere dello studente. Anche in questo caso le feature relative al voto medio di matematica e all'area geografica dello studente figurano come le più significative, e le loro importanze sono fortemente distaccate dalle altre, come la successiva variabile P7 estratta dai risultati INVALSI e rappresentante la capacità di utilizzare strumenti, modelli e rappresentazioni per il trattamento quantitativo dell'informazione.

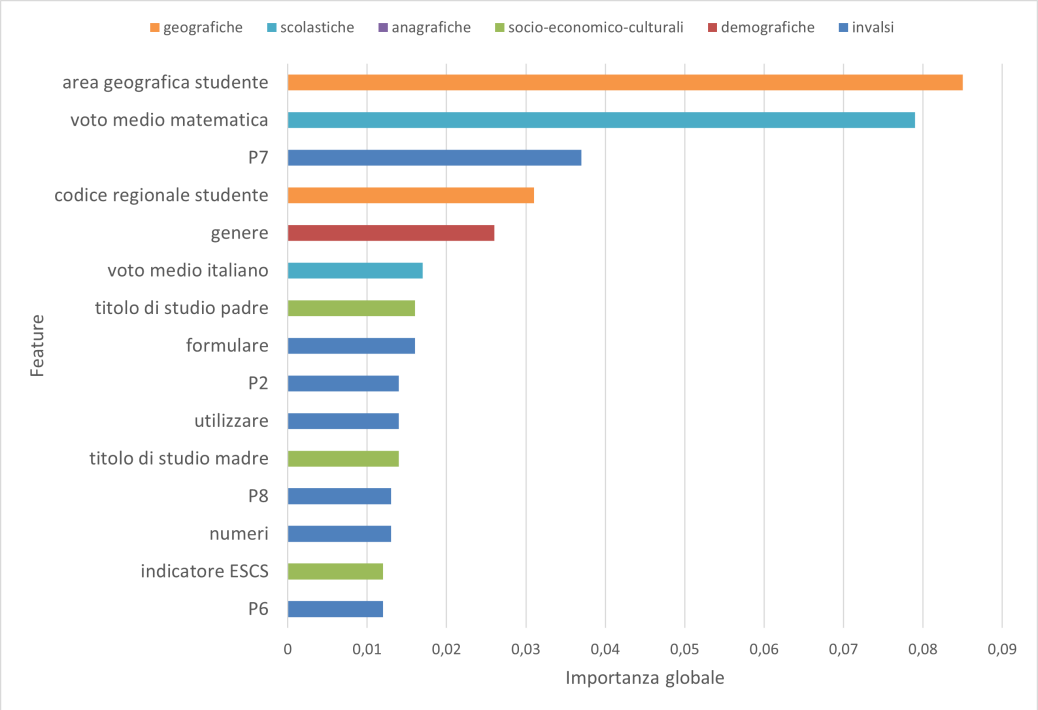

Figura 4.7: Importanze globali ottenute dai valori SHAP per il feature tokenizer transformer

Il grafico *beeswarm* mostra intuitivamente il modo in cui i valori SHAP sono distribuiti per le variabili ed è subito possibile notare come quelle che si concentrano maggiormente agli estremi riguardano proprio le due feature ritenute più importanti. Le importanze SHAP associate all'area geografica si distribuiscono infatti in un range da -0.2 a 0.3 circa, il che significa che tale informazione ha apportato per alcune predizioni un aumento o decremento percentuale dell'output fino a 30 punti. Anche i valori SHAP del voto di matematica, in questo caso, si distribuiscono in un range piuttosto ampio dal -0.2 allo 0.2. I valori per le altre feature risultano essere invece maggiormente dense sull'asse dello 0, specificando un impatto basso se non nullo sull'output finale.

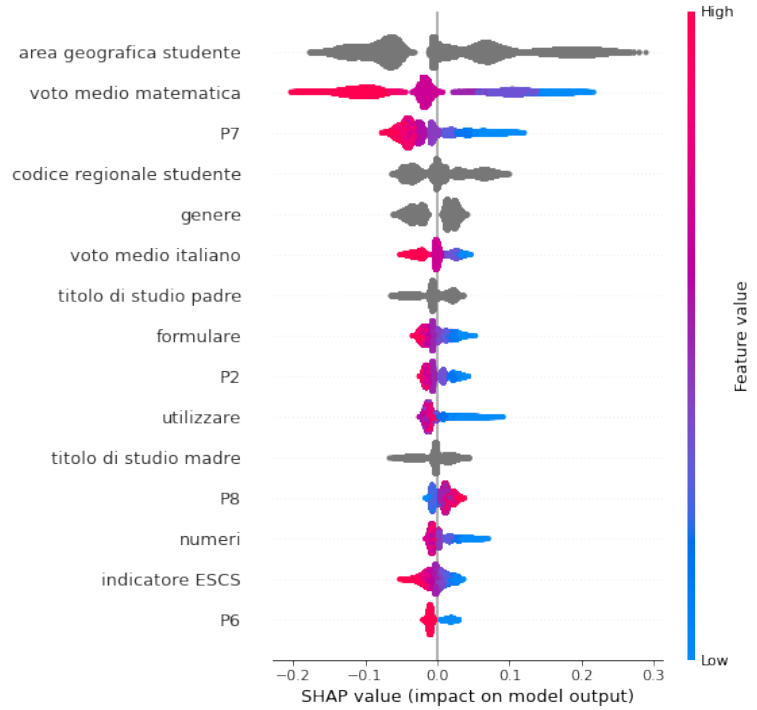

Figura 4.8: Distribuzione valori SHAP rispetto le feature per la predizione della classe positiva per il basso rendimento per il Feature tokenizer transformer

4.3 Random forest

I risultati relativi alle importanze delle variabili, calcolate sulla base dei valori LIME, per le predizioni del random forest sono presentate nel grafico 4.9. Anche in questo caso, le migliori 15 feature trovate sono indicativamente le stesse riscontrate per gli altri modelli. Tra le prime tre, come riscontrato anche per gli altri modelli, è possibile trovare il codice regionale dello studente ed il voto medio di matematica. In questo caso, però, è presente anche la variabile relativa al punteggio totale del test INVALSI, per ora osservato tra le migliori 15, anche se non con un impatto così alto, solo per le spiegazioni relative alla CE. A differenza, però, dalle altre due feature più importanti, è interessante notare, nel grafico 4.10 che la distribuzione dei valori SHAP

relativi risulta essere non particolarmente dipendente dal valore assegnato. Tale considerazione è dedotta dalla presenza di valori sia bassi che alti con effetti SHAP bassi (i quali hanno effetti negativi sulla predizione dell'output positivo).

Inoltre è presente un grande distacco nelle importanze tra le prime tre e la quarta feature, che anche in questo caso è la variabile P7, la quale risulta essere la migliore tra il gruppo delle feature con importanze medio-basse, coerentemente con quanto osservato anche per gli altri modelli.

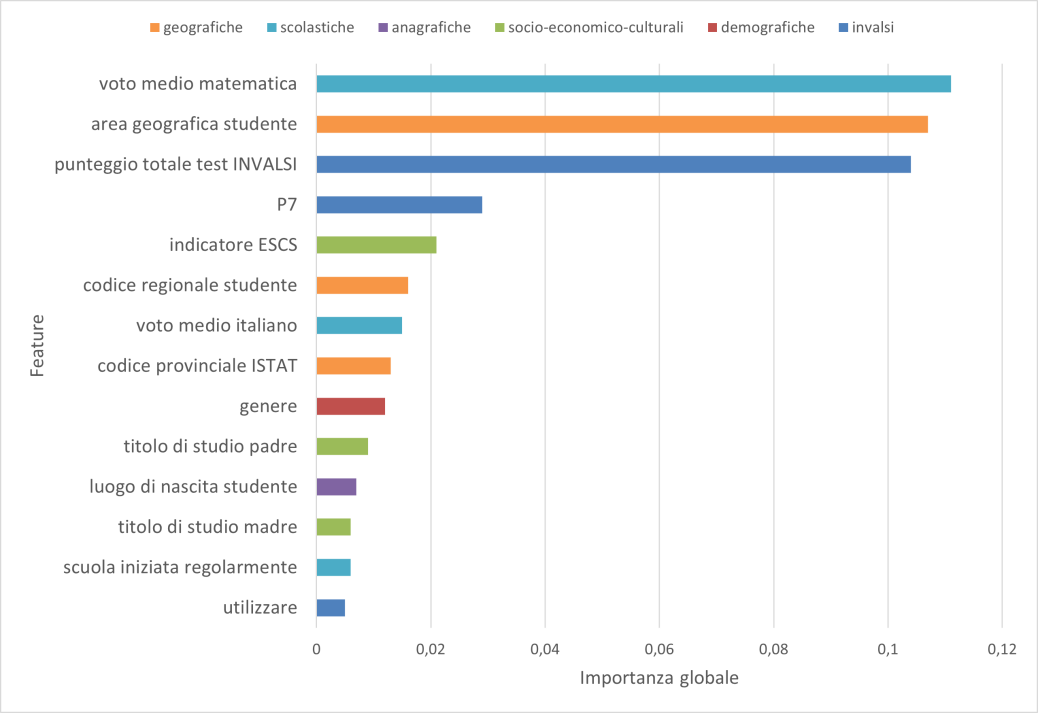

Figura 4.9: Importanze globali ottenute dai valori LIME per il random forest

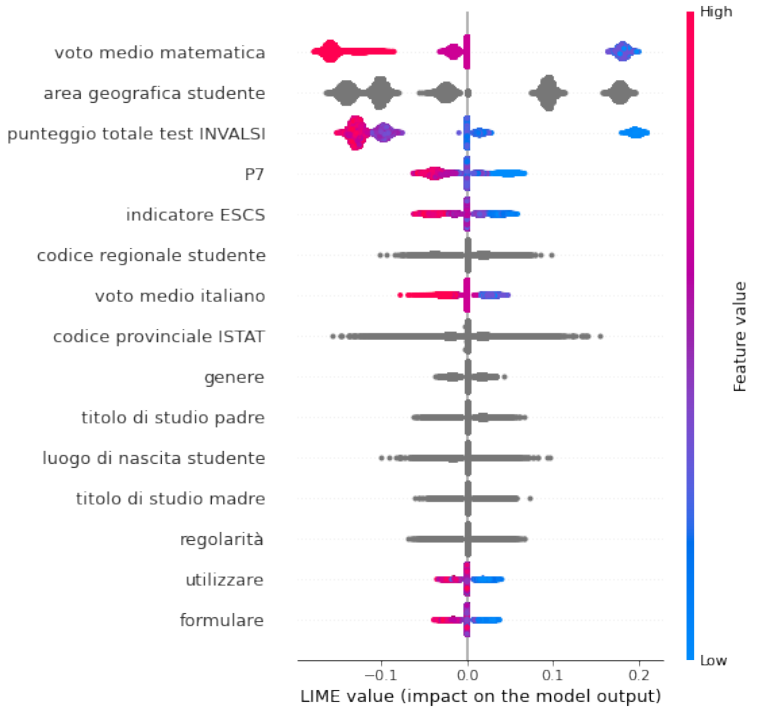

Figura 4.10: Distribuzione valori LIME per la predizione della classe positiva per il basso rendimento per la random forest

In relazione all'applicazione del metodo SHAP, invece, le importanze calcolate sono osservabili nel grafico 4.11. Anche in questo caso si nota una coerenza con il ranking delle migliori feature trovate sulla base delle importanze globali. Tra le prime tre, infatti, figurano il codice regionale e la media del voto di matematica, in aggiunta al punteggio totale del test INVALSI che è stato riconosciuto come di rilevante impatto anche secondo i risultati basati su LIME.

Il grafico *beeswarm* 4.12 accentua ulteriormente la correlazione tra valori bassi di punteggio e l'impatto positivo sull'output finale di basso rendimento, fino a un massimo di 30 punti percentuali.

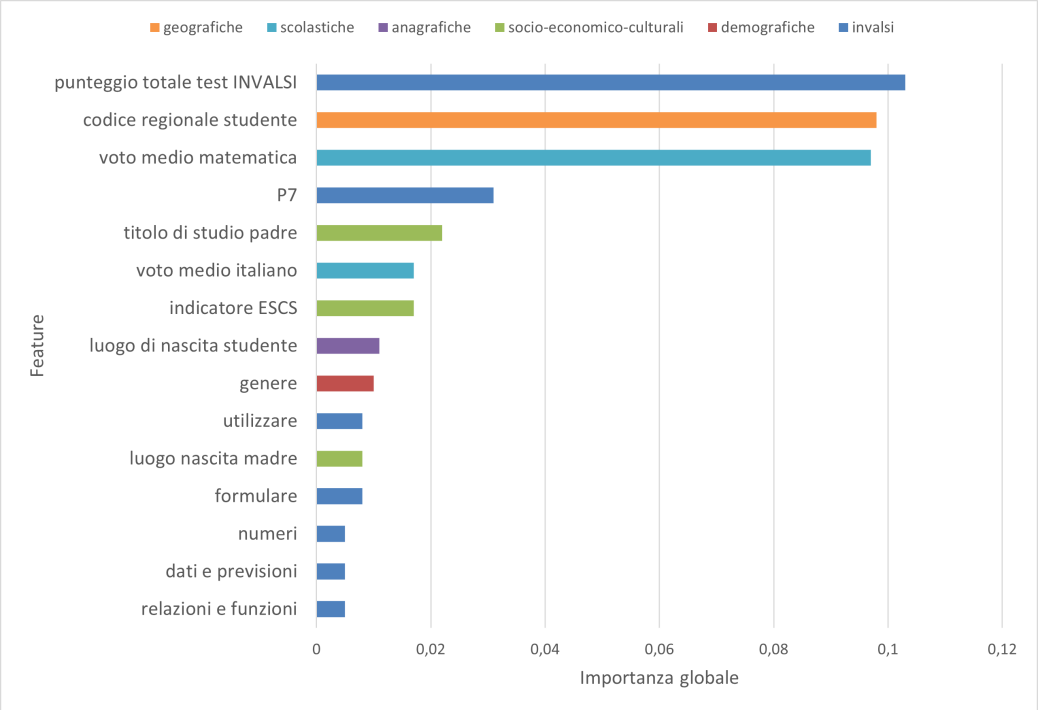

Figura 4.11: Importanze globali ottenute dai valori SHAP per il random forest

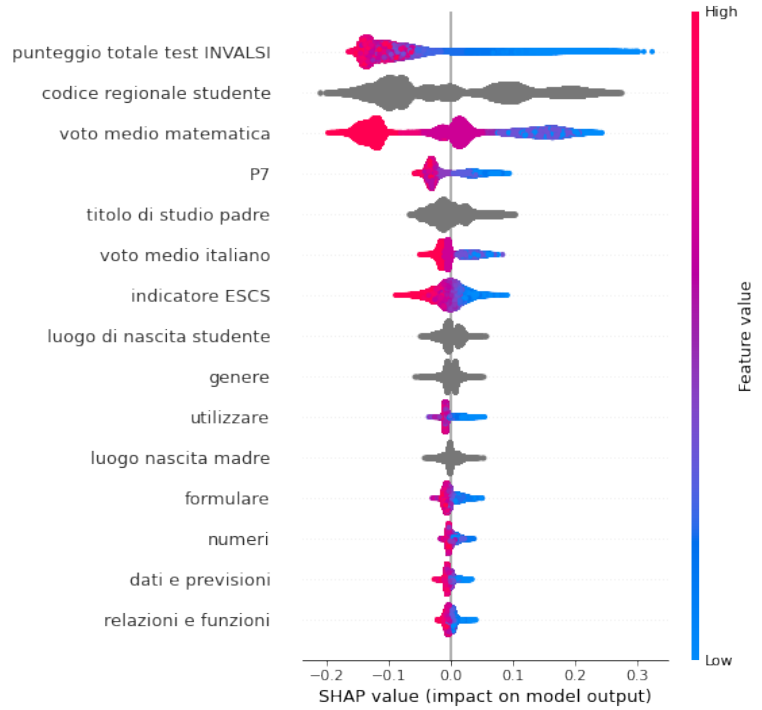

Figura 4.12: Distribuzione valori SHAP per la predizione della classe positiva per il basso rendimento per la random forest

4.4 Discussione risultati

Prima di procedere con la discussione dei risultati, è importante notare che vi sono diversi elementi che potrebbero aver influenzato lo svolgimento degli esperimenti e i relativi risultati ottenuti. Ad esempio, la scelta di utilizzare il metodo Tree SHAP, in quanto più efficiente, per ottenere i valori SHAP dell'albero invece del Kernel SHAP, potrebbe aver portato al calcolo di influenze differenti delle variabili. Tuttavia, va sottolineato che entrambi i metodi hanno lo stesso obiettivo di approssimare i valori SHAP, seppur con approcci differenti, e che pertanto i valori generati sono comunque comparabili. Inoltre, nonostante le leggere differenze di ranking, le variabili migliori identificate dall'albero risultano essere simili a quelle degli altri modelli considerati.

Un ulteriore elemento che potrebbe aver avuto un'influenza sul calcolo delle importanze globali, in questo caso basate su LIME, è la selezione delle feature effettuata durante il training dei modelli surrogati. Tale scelta semplifica il modello e lo vincola a considerare solo le variabili più significative. Di conseguenza, per poter effettuare il confronto con gli altri valori, è stato assegnato un peso LIME pari a 0 alle variabili non scelte. Per valutare ulteriormente l'importanza di tali variabili, sarebbe interessante ripetere i calcoli LIME vincolando i modelli surrogati a considerare tutte le feature, così da poter osservare i valori LIME associati ad esse, allo stesso modo di quanto fatto per i valori SHAP.

Dai risultati generali ottenuti, è possibile osservare che tutti i modelli analizzati riconoscono che i due cluster di informazioni con un maggior valore predittivo sono quello relativo ai dati scolastici-INVALSI e quello dei dati geografici. Solo in alcuni casi sono state considerate particolarmente importanti informazioni di tipo socio-economico, principalmente relative alle professioni dei genitori e all'indicatore ESCS, senza però mai avere delle importanze particolarmente significative. In generale, per tutte le combinazione metodo-modello, le tre variabili con maggior potere predittivo riconosciute, solitamente caratterizzate da un distacco elevato rispetto le importanze delle altre variabili, hanno sempre incluso il voto medio di matematica dello studente e l'area geografica in cui esso è residente. La terza variabile tra le migliori tre, invece, varia a seconda del modello utilizzato: per il modello CE è il voto medio di italiano, per il modello FTT sono il genere dello studente o la variabile P7, mentre per il modello RF è il punteggio totale ottenuto nella prova INVALSI.

È interessante notare che il random forest presenta un comportamento leggermente diverso rispetto agli altri modelli, in quanto, sia secondo i valori SHAP che LIME, sembra essere fortemente influenzato dal punteggio totale ottenuto nella prova INVALSI, variabile che non è particolarmente considerata dagli altri modelli.

Ciò che risulta interessante è che, per i singoli modelli, le importanze

globali associate alle feature sono simili, sia se calcolate tramite valori LIME che SHAP, e anche il ranking generale delle variabili risulta esserlo. Ciò permette di dedurre che i due metodi, applicati allo stesso modello, sono concordi su quali variabili considerare come più influenti nell'iter decisionale.

In relazione ai risultati ottenuti singolarmente per ogni tipo di modello analizzato, è stato osservato che il modello CE è quello che risulta avere dei valori relativi alle importanze globali generalmente bassi, i quali raggiungono un valore massimo di 0.03, sia se calcolati rispetto i valori LIME che rispetto SHAP. Questo permette di dedurre una robustezza minore del modello nel processo decisionale, in quanto non esistono delle variabili di particolare e forte importanza predittiva rispetto l'output, anche se le migliori che figurano tra esse sono comunque coerenti con quelle riconosciute dai metodi per gli altri modelli analizzati.

In base alle informazioni riportate, risulta evidente che, nonostante le variabili più importanti considerate siano solitamente le stesse, il modello FTT sembra essere il più robusto tra i tre. Infatti, le feature principali riconosciute sia da LIME che da SHAP presentano delle importanze globali più ampie e decisive rispetto quelle riscontrati negli altri modelli.

Conclusioni

Nel lavoro di tesi presentato è stato proposto il confronto delle due tecniche locali di explainability SHAP e LIME applicate a modelli di machine learning sviluppati nello studio appartenente all'ambito EDM di Zanellati, Zingaro e Gabbrielli (2022) [19]. Tale lavoro propone tre differenti tipologie di classificatori binari fini alla previsione del basso rendimento scolastico di uno studente in seconda superiore, sulla base dei risultati dei test INVALSI svolti di quinta elementare ed altri fattori culturali, socio-economici, demografici e scolastici. I modelli proposti per la risoluzione di tale problema sono di tipo random forest, categorical embeddings neural network e feature tokenizer transformer. Gli ultimi due modelli sono caratterizzati da un alto livello di opacità, ovvero la loro complessità rende di difficile comprensione l'iter decisionale seguito per fornire le predizioni. Nell'ottica dell'utilizzo di tali modelli in un contesto reale, risulta però importante che gli output forniti siano comprensibili per le parti interessate al fine permettere a esperti del dominio di valutare e correggere eventuali errori evitando che decisioni errate possano avere delle conseguenze dannose per gli studenti coinvolti o a livello organizzativo. Inoltre, tali modelli possono anche essere utilizzati per scoprire nuovi pattern nell'ambito e generare conoscenza.

Il presente lavoro di tesi è stato formulato per affrontare questa problematica, applicando ai modelli oggetto di studio le tecniche XAI SHAP e LIME capaci di definire, per singole predizioni, l'importanza delle feature originali nel calcolo dell'output finale sotto forma di valori additivi. Lo sviluppo della soluzione ha comportato l'esecuzione di esperimenti per determinare

l'importanza globale delle variabili per ogni modello analizzato. A tal fine, sono stati calcolati, sia con la tecnica SHAP che LIME, i singoli effetti delle feature per ogni predizione relativa alle istanze di un dataset di test riferimento e successivamente, per ogni attributo, i valori sono stati aggregati per ottenere la loro importanza globale.

Dopo aver effettuato un'analisi e comparazione dei valori di importanza calcolati per i diversi modelli, i risultati ottenuti hanno confermato l'ipotesi iniziale riguardo la coerenza delle feature riconosciute come le più importanti nei processi decisionali seguiti. Per tutti i modelli, infatti, i metodi SHAP e LIME definiscono due cluster principali di informazioni con un maggiore valore predittivo, ovvero quello relativo ai dati scolastici-invalsi e quello per le informazioni geografiche. Inoltre, in generale, per tutte le combinazioni metodo-modello, le tre variabili con maggior potere predittivo riconosciuto, sempre con un distacco elevato rispetto le importanze delle altre variabili, hanno sempre incluso il voto medio di matematica dello studente e l'area geografica in cui esso è residente. Nonostante l'utilizzo di tecniche e modelli diversi, le variabili che hanno maggior impatto sulle predizioni sono risultate essere quindi principalmente le stesse, seppur con magnitudo leggermente diverse. Tali risultati hanno confermato quindi che l'approccio intrapreso risulta essere corretto e significativo, permettendo di valutare la coerenza degli iter decisionali seguiti dai modelli, rafforzandola ulteriormente osservando i risultati concordi delle altre soluzioni.

Il presente lavoro rappresenta un punto di partenza per ulteriori sviluppi futuri che possono comprendere estensioni inerenti ai metodi SHAP e LIME utilizzati o il coinvolgimento di nuovi metodi XAI non considerati in questo studio. Relativamente ai metodi SHAP e LIME, potrebbe essere interessante analizzare i risultati ottenuti dagli stessi metodi esplorando configurazioni differenti rispetto quelle proposte. Ad esempio, si potrebbe modificare il dataset per l'integrazione delle feature per Kernel SHAP o definire una classe differente di modelli surrogati LIME da generare, diversi da quella attuale della Ridge regression. Inoltre, per completezza dello studio, si potrebbe

estendere l'applicazione del metodo Kernel SHAP anche al random forest analizzato, su cui invece è stato applicato unicamente il Tree SHAP. Tali esperimenti possono essere utili al fine di valutare ulteriormente la robustezza dei risultati proposti, ricercando conferme riguardo le features riconosciute come le più influenti.

Altri possibili sviluppi possono invece riguardare l'applicazione di tecniche differenti di post-modelling, agnostiche e locali, sempre rivolte ad una comprensione da parte degli educatori esperti del dominio. Un esempio potrebbe essere rappresentato dal metodo DiCE [8], il quale genera spiegazioni controfattuali. Le spiegazioni prodotte da questo metodo riguardano infatti i cambiamenti che dovrebbero essere apportati all'input al fine di avere un output diverso da quello attuale. Se visualizzate correttamente e in modo chiaro, tali spiegazioni, prettamente locali, possono mostrarsi molto intuitive anche per gli educatori. Potrebbe essere interessante far analizzare a esperti di dominio le spiegazioni fornite dalle tecniche SHAP, LIME e DiCE in modo da confrontare, in questo contesto, la facilità di comprensione dei metodi.

Infine, si potrebbe considerare l'integrazione dei metodi proposti in un sistema utilizzabile dai decision maker nel settore educativo. Oltre alla presentazione delle importanze e dell'influenza generale delle variabili, potrebbe essere utile fornire la rappresentazione dell'iter decisionale rispetto a singole predizioni, eventualmente tramite i grafici messi a disposizione dalle librerie SHAP e LIME, poiché forniscono spiegazioni molto intuitive per esperti non tecnici.

Bibliografia

[1] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[2] Maruan Al-Shedivat, Avinava Dubey, and Eric P. Xing. Contextual explanation networks. *CoRR*, abs/1705.10301, 2017.

[3] Sarah Alwarthan, Nida Aslam, and Irfan Ullah Khan. An explainable model for identifying at-risk student at higher education. *IEEE Access*, 10:107649–107668, 2022.

[4] Máté Baranyi, Marcell Nagy, and Roland Molontay. Interpretable deep learning for university dropout prediction. In *Proceedings of the 21st Annual Conference on Information Technology Education*, SIGITE '20, page 13–19, New York, NY, USA, 2020. Association for Computing Machinery.

[5] Noel Codella, Michael Hind, Karthikeyan Natesan Ramamurthy, Murray Campbell, Amit Dhurandhar, Ramazon Kush, Dennis Wei, and Aleksandra Mojsilovic. Ted: Teaching ai to explain its decisions, 11 2018.

[6] Pratiyush Guleria and Manu Sood. Explainable ai and machine learning: performance evaluation and explainability of classifiers on educational data mining inspired career counseling. *Education and Information Technologies*, 28:1081 – 1116, 2022.

[7] Khan Md. Hasib, Farhana Rahman, Rashik Hasnat, and Md. Golam Rabiul Alam. A machine learning and explainable ai approach for predicting secondary school student performance. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0399–0405, 2022.

[8] Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. *arXiv e-prints*, page arXiv:1905.07697, May 2019.

[9] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

[10] Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *CoRR*, abs/1802.03888, 2018.

[11] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.

[12] Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. 03 2018.

[13] Gomathy Ramaswami, Teo Susnjak, and Anuradha Mathrani. On developing generic models for predicting student outcomes in educational data mining. *Big Data and Cognitive Computing*, 6(1), 2022.

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[15] Carolyn Rosé, Elizabeth McLaughlin, Ran Liu, and Kenneth Koedinger. Explanatory learner models: Why machine learning (alone) is not the

answer: Explanatory learner models. *British Journal of Educational Technology*, 50, 08 2019.

[16] Hanne Scheers and Tinne De Laet. Interactive and explainable advising dashboard opens the black box of student success prediction. In *Technology-Enhanced Learning for a Free, Safe, and Sustainable World: 16th European Conference on Technology Enhanced Learning, EC-TEL 2021, Bolzano, Italy, September 20-24, 2021, Proceedings*, page 52–66, Berlin, Heidelberg, 2021. Springer-Verlag.

[17] Vinitra Swamy, Bahar Radmehr, Natasa Krco, Mirko Marras, and Tanja Käser. Evaluating the explainers: Black-box explainable machine learning for student success prediction in moocs, 07 2022.

[18] Alexandra Vultureanu-Albiși and Costin Bădică. Improving students' performance by interpretable explanations using ensemble tree-based approaches. EasyChair Preprint no. 5441, EasyChair, 2021.

[19] Andrea Zanellati, Stefano Pio Zingaro, and Maurizio Gabbrielli. Student low achievement prediction. page 737–742, Berlin, Heidelberg, 2022. Springer-Verlag.

[20] Jan Zilke, Eneldo Mencía, and Frederik Janssen. Deepred – rule extraction from deep neural networks. pages 457–473, 10 2016.