Università degli Studi di Salerno

Dipartimento di Informatica

Progetto Fondamenti
di
Fondamenti di Data Science e Machine Learning

Professore
Giuseppe Polese
Loredana Caruccio
Stefano Cirillo

Studenti
Antonio Cimino 0522500922
Vincenzo De Martino 0522500966
Diego Varriale 0522500910

Sommario

Abstract 2

1. Introduzione 3

2. Fondamenti Teorici 5

2.2 Dipendenze Funzionali 5

2.3 Classificatori 9

3. Esperimenti 12

3.1 Difficult Dataset Generation Algorithm 12

3.2 Dataset Reali 15

Conclusioni 20

Abstract

Il nostro studio si basa sul documento “Evaluating Classification Feasibility Using Functional Dependencies” dove si ipotizza il valore del G3 come limite superiore dell’accuratezza ottenibile dai vari modelli di classificazione. Abbiamo implementato l’algoritmo G3 per testarlo su diversi dataset prima generati casualmente e poi su dataset reali, successivamente abbiamo scelto quattro diversi classificatori tra i dieci usati nel documento applicandoli sugli stessi dataset del G3 così da comparare in un secondo momento le accuratezze dei modelli e il valore del G3, per avere una conferma della ipotesi dell’articolo.

1. Introduzione

Negli ultimi anni sono state sviluppate numerose tecniche e librerie per l'utilizzo di modelli predittivi, ma ciò che Marie Le Guilly, Jean-Marc Petit, Vasile-Marian Scuturici si sono chiesti è quanto sono affidabili i modelli di ML e quali tecniche o strumenti permettono di valutare l'affidabilità dei modelli predittivi, e ciò è stato documentato nell'articolo "Evaluating Classification Feasibility Using Functional Dependencies". Nell'articolo citato si dice che si può dare un limite superiore stretto dell'accuratezza del classificatore basato sulla misura del G3 per le dipendenze funzionali. Le Dipendenze funzionali e la classificazione supervisionata sembrano due problemi distanti e non avere punti in comune. Tuttavia, questi approcci risultano essere complementari, come mostrato nella seguente osservazione sempre ripresa dall'articolo:

> *"Dato un insieme di dati r su {A_1 ,...,A_n ,C} dove i valori C rappresentano la classe da prevedere, gli algoritmi di classificazione cercano di trovare una funzione per prevedere un output (valore C) basato su un dato input (i valori A_1,...,A_n), mentre la soddisfacibilità della dipendenza funzionale A_1,...,A_n → C in r non esprime altro che l'esistenza di quella funzione*."

Di conseguenza, in un contesto di classificazione, lo studio dell'esistenza di una dipendenza funzionale tra gli attributi e l'attributo da classificare potrebbe produrre informazioni significative sulle possibili prestazioni del classificatore. Può anche identificare controesempi rilevanti, cioè tuple nel set di dati che causeranno il fallimento di un classificatore. Questo lavoro di confronto avviene secondo metriche ben note, in modo da poter dare un limite superiore per l'accuratezza del classificatore. In particolare, viene utilizzato il G3 sull'algoritmo "Difficult dataset generation algorithm" per la generazione di un dataset, contenente una colonna che rappresenta la classe di una determinata tupla e i restanti attributi, i quali presi insieme, identificano la tupla [3].
Nel seguente documento sono stati creati diversi dataset utilizzando questo algoritmo, andando a cambiare diversi valori per stabilire la dimensione del dataset e il numero delle classi. Con questo algoritmo abbiamo generato set di dati con diversi parametri per validare quello che hanno dichiarato gli autori nell'articolo e abbiamo utilizzato quattro classificatori (Linear Support Vector, Random Forest, Decision Tree ed AdaBoost) per ottenere diverse accuratezze con cui poi poter valutare se il G3 rappresentasse un limite per queste ultime. I classificatori analizzati nello studio si basano sui parametri di default dalle librerie e noi ci siamo limitati ad analizzare quattro di questi poiché le accuracy di altri classificatori sono risultate pressoché identiche nel dataset creato dall'algoritmo. Terminati gli esperimenti sui dataset generati casualmente abbiamo poi mostrato, attraverso grafici, i risultati ottenuti dai

test, per mostrare proprio che il valore G3 nei vari dataset fosse sempre superiore alle accuratezze. Successivamente, utilizzando diversi dataset reali è stato applicato lo stesso procedimento descritto in precedenza per evidenziare i comportamenti dei test ed analizzare eventuali anomalie. Per ognuno di questi dataset, in un primo momento, è stato applicato un processo di data cleaning a causa del fatto che diversi valori degli attributi mancano oppure contengono come valori “?”, in un secondo momento, sono stati testati con i classificatori, descritti in precedenza, confrontandoli con il G3. Come da suggerimento poi, abbiamo scelto un’ulteriore misura per il calcolo della soddisfacibilità delle dipendenze per testare i risultati ottenuti, cioè quella dell’impurità.
Successivamente all’introduzione appena descritta, troveremo un capitolo incentrato sull’analisi delle tecnologie e delle metodologie utilizzate per produrre lo stesso studio sviluppato dagli autori del “Evaluating Classification Feasibility Using Functional Dependencies” al fine di trovare una motivazione sul perché usiamo questi tipi di algoritmi e di classificatori di machine learning. Nel terzo capitolo andremo ad illustrare i risultati ottenuti dagli esperimenti effettuati, mostrando attraverso l’utilizzo di grafici le differenze di accuratezze tra i classificatori e le diverse misure utilizzate.

2. Fondamenti Teorici

Una dipendenza funzionale è una proposizione generale applicata alla realtà. Esse caratterizzano l'esistenza di una funzione che un algoritmo di classificazione cerca di definire. Piuttosto che concentrarsi sul modello stesso, ci proponiamo di spiegare la limitazione riguardante l'input di un problema di classificazione, ovvero il set di dati, per mostrare perché l'accuratezza di qualsiasi classificatore è limitato dai dati disponibili. A prima vista, la classificazione supervisionata e le dipendenze funzionali non appaiono essere due concetti correlati, infatti non si applicano agli stessi problemi. Entrambi condividono il concetto di funzione: *'una funzione f: X → Y è un mapping di ogni elemento x dell'insieme X (dominio) verso un unico elemento y di un altro insieme Y (codominio)'*. Gli algoritmi di rilevamento di FD sono progettati in modo tale da poter essere facilmente generalizzati anche ad altri tipi di dipendenze. Una dipendenza funzionale, in sintesi, esprime i valori degli attributi da determinare in modo univoco, in modo da esplicitare la struttura implicita delle relazioni [1].

2.2 Dipendenze Funzionali

Definiamo ora la sintassi e la semantica di una dipendenza funzionale FD:
Sia R uno schema di relazione, e X, Y ⊆ R. Una FD su R è un espressione della forma R : X → Y. Sia r una relazione su R e X → Y una dipendenza funzionale su R. X → Y è soddisfatto in r, indicato con r |= X → Y , se e solo se per tutti t1, t2 ∈ r, t1[X] = t2[X] allora t1[Y] = t2[Y]. E può essere verificata utilizzando questa proprietà: *Sia r una relazione su R e X → Y un FD su R. Allora: r |= X → Y ⇔ $|\pi_{XY}(r)| = |\pi_X(r)|$, dove π_{XY} e π_X rappresentano le proiezione su r.*
Le dipendenze funzionali sono state ampiamente utilizzate in operazioni avanzate sui database come nel Data cleaning, Query Relaxation e Record Matching. Tuttavia, c'è stata la necessità di rilassare alcuni vincoli della definizione di dipendenza funzionale canonica. A questo scopo sono state introdotte nuove definizioni di dipendenza funzionale: le cosiddette dipendenze funzionali rilassate (RFD). Informalmente, una RFD è una dipendenza funzionale che si rilassa sul confronto delle tuple, utilizzando vincoli sulla distanza o sulla somiglianza tra i valori degli attributi, e/o che si rilassa sull'estensione, utilizzando una misura di copertura per indicare il numero minimo o la percentuale di tuple su cui deve valere l'RFD e/o utilizzando condizioni che limitano il dominio di applicabilità dell'RFD.

Definizione (Dipendenza Funzionale Rilassata)
Considerato uno schema di database relazionale R e uno schema di relazione $R = (A_1,...,A_m)$ di R. Una RFD φ su R è indicato con $X_{\Phi 1} \rightarrow Y_{\Phi 2}$ dove

- $X = X_1,...,X_h$ e $Y = Y_1,...,Y_k$, con $X,Y \subseteq attr(R)$ e $X \cap Y = \varnothing$;

- $\Phi_1 = \wedge_{Xi \in X} \varphi_i[X_i]$ ($\Phi_2 = \wedge_{Yj \in Y} \varphi_j[Y_j]$, resp.), dove φ_i (φ_j, resp.) è una congiunzione di predicati su X_i (Y_j, resp.) con i = 1,...,h (j = 1,...,k , resp.). Per qualsiasi coppia di tuple (t_1, t_2) $\in$ dom(R), il vincolo Φ_1 (Φ_2, resp.) è vero se $t_1[X_i]$ e $t_2[X_i]$ ($t_1[Y_j]$ e $t_2[Y_j]$, resp.) soddisfano il vincolo φ_i (φ_j, resp.) $\forall$ i $\in$ [1,h] (j $\in$ [1,k], resp.).
- Ψ è una misura di copertura definita sul dom(R), che quantifica la quantità di tuple che violano o soddisfano φ. Tra le misure di copertura più comunemente utilizzate vi sono la confidenza, il G3error e la probabilità.
- ε è una soglia che indica il limite superiore (o il limite inferiore nel caso in cui l'operatore di confronto sia ≥) per il risultato della misura di copertura.

Dato r $\subseteq$ dom(R) un'istanza di relazione su R, r soddisfa la RFD φ, indicata con r $\models$ φ, se e solo se: $\forall$ $t_1,t_2 \in$ r, se Φ_1 indica vero, allora quasi sempre Φ_2 indica vero. In questo caso, quasi sempre è espresso dal vincolo $\Psi \leq \varepsilon$ [8].

Se si vuole mantenere il concetto di vincolo tra i dati, è importante ammettere eccezioni o condizioni che restringono l'insieme delle tuple per cui deve valere una dipendenza, o catturare vincoli esistenti sfruttando la similitudine e non l'esatta uguaglianza tra i valori degli attributi. Esistono due principali criteri di rilassamento ovvero il confronto tra i valori degli attributi (o attribute comparison) e il grado di soddisfacibilità (o extent). Nell'attribute comparison si vuole ammettere la possibilità che una RFD possa essere soddisfatta considerando anche coppie di tuple con valori simili (il grado di similitudine da rispettare può essere definito anche sui singoli attributi). In genere, si applica questo tipo di rilassamento in una delle due seguenti modalità: usando un confronto approssimato (approximate match) oppure usando un criterio di ordinamento. Esse, inoltre, possono essere generalizzate utilizzando il concetto di vincolo. Nel contesto delle RFD, un vincolo esprime la "vicinanza" di due valori in uno specifico dominio è rappresentato da una funzione ϕ. Dati due attributi A e B su un dominio D valuta la similarità/distanza di A e B a seguito di una possibile modifica dei loro valori. Invece, nel rilassamento sull'extent si vuole ammettere la possibilità che una RFD possa essere soddisfatta anche per un sottoinsieme di tuple di un'istanza di database (non necessariamente per tutte). In genere, si applica questo tipo di rilassamento in una delle due seguenti modalità: utilizzando una misura di copertura (coverage measure) oppure utilizzando una condizione.

In questo contesto, la misura G3 sembra essere di cruciale importanza per il problema di classificazione, in quanto rappresenta la porzione di tuple nel dataset, che soddisfano la dipendenza funzionale considerata. Seguendo la definizione di G3, affinché sia massimale, dovrebbe mantenere il maggior numero possibile di tuple, rimuovendo tutti i controesempi della dipendenza funzionale considerata. Di conseguenza, questo può essere fatto raggruppando tutte le tuple che condividono lo stesso lato sinistro, quindi selezionando tra di esse quelli che condividono lo

stesso lato destro. Questo permette di eliminare tutti i controesempi, togliendo il numero minimo di controesempi [3].

$$G3\ (X \rightarrow Y,r)=\frac{\sum_{xi,yi} max|\pi XY(\sigma X=xi \wedge Y=yi\ (r))|}{|r|}$$

Considerando la tabella sottostante, in cui abbiamo una dipendenza funzionale di A,B ->Label, se proviamo ad applicare il g3 verrà innanzitutto calcolato il numero di tuple uguali (quindi con lo stesso valore di A, B e Label) e poi verranno messe a confronto le tuple che presentano gli stessi attributi nel lato destro della dipendenza funzionale ma con valore diversi sul lato sinistro, confrontandole con il numero precedentemente calcolato e da questo confronto prendiamo il valore maggiore che corrisponderà al numero di tuple massimo che potremmo mantenere per avere una RFD, mentre se calcoliamo 1 - G3 avremo il numero minimo di tuple da eliminare. Per esempio, con A=X, B=Y e Label=Z avremo come numero di ripetizioni un valore pari a 2, invece con A=X, B=Y e Label=K un numero di ripetizioni pari a 1, siccome A e B sono uguali ma la Label è diversa la mettiamo a confronto e andiamo a prendere il numero maggiore calcolato, ovvero 2.

A	B	Label
X	Y	Z
X	Y	K
X	Y	Z
Y	Y	Z
L	Y	K

Proseguendo con gli esperimenti prefissati andremo poi a mostrare come questo parametro potrebbe rappresentare un limite superiore sull'accuratezza dei classificatori sui diversi set di dati.

Ora introduciamo anche un’altra metrica per le RFD che rilassano sull’extend , ovvero l’impurità, questo perchè come detto nell’introduzione oltre a confutare la tesi per cui il g3 è un limite superiore vorremmo provare anche se altre metriche possono rappresentare un limite. Quindi in merito all’impurità: Sia S un generatore, S sia un insieme e sia PART(S) l'insieme di tutte le partizioni di S. L'impurità di un sottoinsieme L di S rispetto a una partizione $\pi \in$ PART(S) e generata da f è la misura dell'impurezza monogenica indotta

$$\mathrm{IMP}_{\pi}^{f}(L) = |L| \left(f\left(\frac{|L \cap B_1|}{|L|}\right) + \cdots + f\left(\frac{|L \cap B_n|}{|L|}\right)\right)$$

dove π = {B 1 ,... ,B n }. L'impurità specifico di L rispetto a π e generato da f

$$\mathrm{imp}_{\pi}^{f}(L) = \frac{\mathrm{IMP}_{\pi}^{f}(L)}{|L|} = f\left(\frac{|L \cap B_1|}{|L|}\right) + \cdots + f\left(\frac{|L \cap B_n|}{|L|}\right)$$

Quando il sottoinsieme L è incluso in uno dei blocchi della partizione, IMPf π (L)=0 e L è detto insieme puro ; altrimenti, L è detto impuro [2]. Come esempio di impurità, riportiamo un dataset reale contenente medicinali e le relative caratteristiche ripreso da un articolo online [8].

Medicine							
Producer	**Name**	**Category**	**Symptoms**	**Weight**	**Cost**	**ActiveIgredient**	**Rx**
Angelini	Aulin	NSAID	Inflammation	100mg	$6,5	Nimesulide	Yes
Angelini	Aulin	NSAID	Inflammation	50mg	$5	Nimesulide	Yes
Dompé	Oki	NSAID	Inflammation	80mg	$4	Ketoprofen	Yes
Zambon It	Spidifen	NSAID	Headache	200mg	$5	Ibuprofen	No
Bayer	Aspirin	NSAID	Fever status	400mg	$4	AcetylsalicyclicAcid	No
Bristol lab.	Panadol	NSAID	Osteoarthritis	500mg	$5	Paracetamol	No

sul dataset presentato potrebbe valere la seguente RFD:

$$D_{TRUE}: ActiveIngredient_{EQ}\ \theta(\pi_{ActiveIngredient,\ \pi_{Rx}}) \leq 0.09 \rightarrow RX_{EQ}$$

Dove RX_{EQ} indica se una prescrizione medica è obbligatoria e θ è una funzione concava e subadditiva che calcola la più grande misura di impurezza sui blocchi Activeingredienti e Rx. Come θ sono diverse le scelte, noi per i nostri esperimenti abbiamo scelto di utilizzare la misura di Gini che in questo caso darebbe il 100% di purezza essendo che ad ogni valore di Activeingredient corrisponde un solo tipo di classe. Il metodo per calcolare l'impurità da noi utilizzato è l'impurità di Gini che è una misura di classificazione errata, che si applica in un contesto di classificazione multiclasse. La formula per l'impurità di Gini è la seguente:

$$G_i = \sum_{k=1}^{n} p_{i,k}^2$$

con p_{ik} che rappresenta la predizione i-esima sull'insieme di riferimento composto da k classi nell'i-esimo elemento. Un esempio di applicazione è dato un insieme S con tuple appartenenti ad n classi, con frequenza p_i della classe i in S. Se T è suddiviso in T_1 con tuple appartenenti alla classe n_1 e T_2 con tuple appartenenti alla classe n_2 allora il valore di Gini per T è:

$$\text{gini}_{\text{split}}(T) = \left(\frac{n1}{n}\right) \text{gini}(T_1) + \left(\frac{n2}{n}\right) \text{gini}(T_2)$$

L'impurità ha molte somiglianze con il G3, tanto è vero che presa una dipendenza funzionale entrambe vanno ad osservare le tuple che hanno come valore uguale gli attributi della parte sinistra della dipendenza funzionale e la diversità nella parte sinistra della dipendenza. La differenza sostanziale sta nel fatto che, mentre nel caso del g3 noi calcoliamo il numero minimo di tuple da eliminare o il massimo numero di tuple da poter considerare per fare in modo che venga considerata una RFD, l'impurità calcola la percentuale di diversità tra le tuple uguali ma con diversa classificazione.

2.3 Classificatori

Il Machine Learning è un metodo di analisi dei dati che automatizza la creazione di modelli analitici. È una branca dell'intelligenza artificiale basata sull'idea che i sistemi possono imparare dai dati, identificare schemi e prendere decisioni con il minimo intervento umano. Uno dei task principali del Machine Learning è la classificazione, cioè il problema di identificare la classe di un nuovo obiettivo sulla base di conoscenza estratta da un training set. Ogni oggetto/elemento osservato è detto "istanza" e l'algoritmo che implementa la classificazione è detto "classifier". Il classificatore esamina l'istanza e la etichetta con una classe. Il modello di classificazione è ottenuto istruendo la macchina utilizzando un apprendimento supervisionato, dove i dati di output nei modelli sono noti al sistema, oppure un apprendimento non supervisionato, il quale funziona con dati senza etichetta in cui l'output si basa solo sulla collezione di percezioni. Inoltre, gli algoritmi di classificazione possono essere lineari, quando il confine delle classi è una retta o un piano, oppure non lineari, quando il confine delle classi è una curva. Un esempio di tipico di classificazione è l'individuazione di email con classe spam e non spam. Per il nostro studio sono stati implementati i seguenti classificatori: SVM, Random Forest, Decision Tree e AdaBoost.

La scelta di utilizzare il **Linear Support Vector** Classification è dovuto alla sua velocità e alla sua capacità di scalare meglio su un grande numero di campioni rispetto agli altri metodi tra quelli inclusi nella categoria Support vector machine. I Support Vector Machine (SVM) sono un insieme di metodi di apprendimento supervisionato utilizzati per la classificazione, la regressione e il rilevamento di valori anomali. I SVM sono efficaci in spazi ad alta dimensionalità e nei casi in cui il numero di dimensioni è maggiore del numero di campioni. Utilizza un sottoinsieme di punti di addestramento nella funzione di decisione (chiamati vettori di supporto), rendendolo efficiente in termini di memoria. Sono forniti kernel comuni, ma è anche possibile specificare kernel personalizzati ed è possibile specificare diverse funzioni

del kernel per la funzione di decisione. SVM utilizza diversi classificatori come il Support Vector Classification, il Nu-Support Vector Classification ed il Linear Support Vector. Sono classi in grado di eseguire la classificazione binaria e multiclasse su un set di dati. Support Vector Classification applica una funzione kernel lineare per eseguire la classificazione, ha una maggiore flessibilità nella scelta delle penalità e delle funzioni di perdita e dovrebbe scalare meglio su un gran numero di campioni. Questa classe supporta input sia denso che sparso e il supporto multiclasse viene gestito secondo uno schema one-vs-the-rest [4].

Random Forest è adatto a situazioni in cui disponiamo di un set di dati di grandi dimensioni e l'interpretabilità non è un problema. Tuttavia, con l'aumentare del numero di alberi in una foresta aumenta anche il tempo computazionale. L'obiettivo dei metodi ensemble è di combinare le previsioni di diversi stimatori di base costruiti con un dato algoritmo di apprendimento al fine di migliorare la generalizzabilità/robustezza su un singolo stimatore. Il modulo sklearn.ensemble include due algoritmi di media basati su alberi decisionali randomizzati: l'algoritmo RandomForest e il metodo Extra-Trees. Entrambi gli algoritmi sono tecniche perturb-and-combine specificamente progettate per gli alberi. Ciò significa che viene creato un insieme diversificato di classificatori introducendo la casualità nella costruzione del classificatore. La previsione dell'insieme è data come previsione media dei singoli classificatori. Nelle RandomForestClassifier e RandomForestRegressor, ogni albero nell'ensemble è costituito da un campione estratto con sostituzione (ovvero un campione bootstrap) dal set di addestramento.
Inoltre, quando si divide ogni nodo durante la costruzione di un albero, la migliore suddivisione si trova da tutte le caratteristiche di input o da un sottoinsieme casuale di dimensioni max_features. Lo scopo di queste due fonti di casualità è ridurre la varianza dello stimatore della foresta. In effetti, i singoli alberi decisionali mostrano tipicamente un'elevata varianza e tendono a sovrapporsi. La casualità iniettata nelle foreste produce alberi decisionali con errori di previsione in qualche modo disaccoppiati. I Random forest ottengono una varianza ridotta combinando alberi diversi, a volte al costo di un leggero aumento della distorsione. In pratica, la riduzione della varianza è spesso significativa e quindi produce un modello complessivamente migliore. Il Random Forest Classifier è un meta stimatore che si adatta ad un numero di classificatori di alberi decisionali su vari sottocampioni del set di dati e utilizza la media per migliorare l'accuratezza predittiva e controllare l'overfitting. La dimensione del sottocampione è controllata con il parametro max_samples se bootstrap=True (predefinito), altrimenti l'intero set di dati viene utilizzato per costruire ogni albero [5].

Gli alberi decisionali sono utili poiché sono più facili da interpretare e più veloci da addestrare ed è questa la motivazione che ci ha portato ad utilizzarlo nel nostro caso di studio. I **Decision Tree (DT)** sono un metodo di apprendimento supervisionato

non parametrico utilizzato per la classificazione e la regressione. L'obiettivo è creare un modello che preveda il valore di una variabile target apprendendo semplici regole decisionali dedotte dalle caratteristiche dei dati. Più profondo è l'albero, più complesse sono le regole decisionali e più adatto è il modello. Sono semplici da capire e da interpretare e gli alberi possono essere visualizzati. Richiede poca preparazione dei dati. Sono inoltre in grado di gestire problemi multi-output. E' possibile convalidare un modello utilizzando test statistici e consente di tenere conto dell'affidabilità del modello. Funziona bene anche se le sue ipotesi sono in qualche modo violate dal vero modello da cui sono stati generati i dati. D'altrocanto i learner dell'albero decisionale possono creare alberi troppo complessi che non generalizzano bene i dati, ovvero generando l'overfitting. Gli alberi decisionali possono essere instabili perché piccole variazioni nei dati potrebbero generare un albero completamente diverso. Questo problema viene mitigato utilizzando alberi decisionali all'interno di un insieme. I learner dell'albero decisionale creano alberi distorti se alcune classi dominano. Nel caso in cui ci siano più classi con la stessa e più alta probabilità, il classificatore predirà la classe con l'indice più basso tra quelle classi [6].

Il principio fondamentale di **AdaBoost** è quello di adattare una sequenza di learner deboli (cioè modelli che sono solo leggermente migliori delle ipotesi casuali, come piccoli alberi decisionali) su versioni ripetutamente modificate dei dati. Le previsioni di tutti loro vengono quindi combinate attraverso un voto a maggioranza ponderata (o somma) per produrre la previsione finale. Le modifiche ai dati ad ogni cosiddetta iterazione di potenziamento consistono nell'applicazione di pesi $w_1,w_2,...,w_n$ a ciascuno dei campioni di addestramento. Inizialmente, questi pesi sono tutti impostati su $w_i=1/N$, in modo che il primo passaggio formi semplicemente uno learner debole sui dati originali. Per ogni successiva iterazione, i pesi dei campioni vengono modificati individualmente e l'algoritmo di apprendimento viene riapplicato ai dati riponderati. In un dato passaggio, gli esempi di addestramento che sono stati previsti in modo errato dal modello potenziato indotto nel passaggio precedente hanno i loro pesi aumentati, mentre i pesi sono diminuiti per quelli che sono stati previsti correttamente. Man mano che le iterazioni procedono, gli esempi difficili da prevedere ricevono un'influenza sempre maggiore. Ogni successivo learner debole è quindi costretto a concentrarsi sugli esempi che mancano ai precedenti nella sequenza. AdaBoost può essere usato sia per problemi di classificazione che di regressione. Un AdaBoost Classifier è un meta-estimatore che inizia inserendo un classificatore sul set di dati originale e quindi inserisce copie aggiuntive del classificatore sullo stesso set di dati ma in cui i pesi delle istanze classificate in modo errato vengono regolati in modo tale che i classificatori successivi si concentrino maggiormente su casi difficili [7].

3. Esperimenti

Sono stati utilizzati diversi dataset per verificare il nostro studio. I primi dataset descritti sono i dataset generati attraverso l'algoritmo "Difficult dataset generation algorithm". Gli altri dataset invece sono stati presi da L'UCI Machine Learning Repository e da Kaggle. L'UCI Machine Learning Repository è una raccolta di database, teorie di dominio e generatori di dati utilizzati dalla comunità di machine learning per l'analisi empirica degli algoritmi di machine learning [9].
Kaggle è una community online di data scientist e professionisti del machine learning. Kaggle consente agli utenti di trovare e pubblicare set di dati, esplorare e costruire modelli in un ambiente di data science basato sul web, lavorare con altri data scientist e ingegneri di machine learning e partecipare a concorsi per risolvere le sfide di data science [10]. Per confrontare i risultati dei modelli sui dataset abbiamo utilizzato uno dei metodi più comprensivi per rappresentare i risultati delle valutazioni dei classificatori è attraverso l'utilizzo del parametro di *accuracy* che si calcola attraverso l'utilizzo della formula seguente:

$$\cdot\, accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

dove TP e TN rappresentano rispettivamente i valori True Positive e True Negative mentre, FP e FN rappresentano False Positive e False Negative. Questi valori compongono la cosiddetta Matrice di Confusione 2x2 che serve per rappresentare la valutazione dei risultati ottenuti, nello specifico:

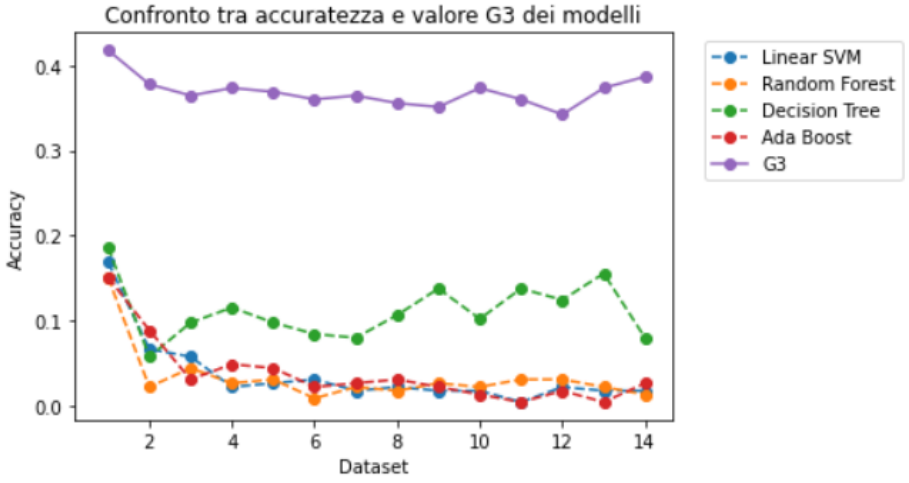
- True Positive (TP): casi in cui un'istanza è correttamente classificata come 'positiva';
- True Negative (TN): casi in cui un'istanza è correttamente classificata come 'negativa';
- False Positive (FP): casi in cui un'istanza è erroneamente classificata come 'positiva';
- False Negative (FN): casi in cui un'istanza è erroneamente classificata come negativa.

3.1 Difficult Dataset Generation Algorithm

L'algoritmo "Difficult Dataset Generation Algorithm" è stato usato per la generazione di un dataset di interi. Ogni dataset prodotto conterrà una serie di colonne che rappresenteranno gli attributi di ogni tupla e l'ultima colonna rappresenta l'attributo

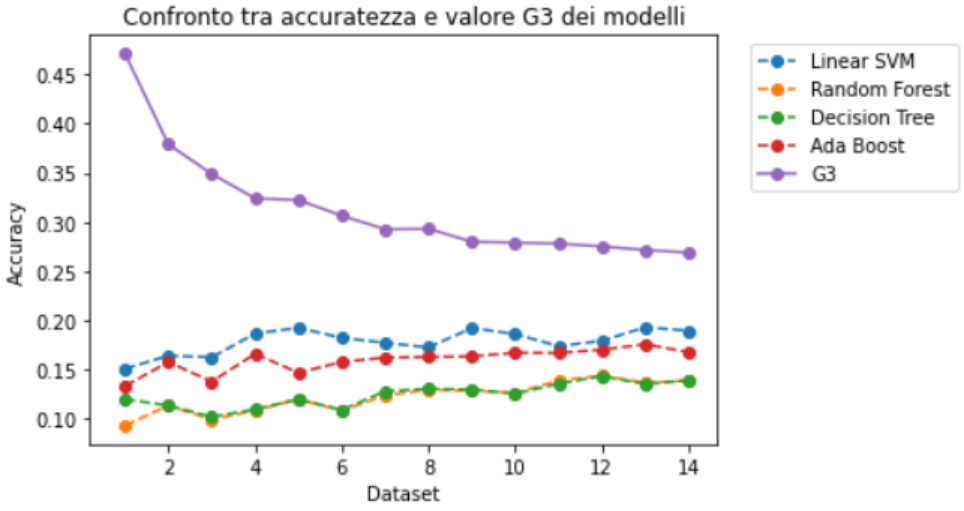
classificatore. Sono stati creati diversi dataset andando a cambiare ogni volta i parametri di ingresso dell'algoritmo in maniera iterativa. I parametri sono N, k, sf che rappresentano rispettivamente il numero di tuple prima della duplicazione, il numero di classi e il fattore di scala. Il numero di attributi invece rimane sempre uguale e anche la dipendenza funzionale valutata con il G3 che consiste in tutto l'insieme di attributi meno che l'ultimo nella parte sinistra e l'ultimo a destra che è l'attributo classificatore. Di seguito presentiamo tutte le tipologie di esperimenti valutati, per ognuno di questi è stato modificato un parametro alla volta. Abbiamo utilizzato un grafico contenente nell'asse delle ascisse il numero indicante un determinato dataset e sull'asse delle ordinate l'accuracy ottenuta dai classificatori e dal G3.

- `esperimento 1: N = 100,k = (i*10), sf = 10`
  Nell'esperimento 1 è stato cambiato il valore di **k**, ovvero il numero di classi, andando a moltiplicare 10 per **i**, con **i** che assume valori da 1 a 15. Con l'aumento delle classi i classificatori utilizzati per gli esperimenti hanno prodotto valori di accuracy peggiori e molto simili tra loro, tranne il Decision Tree che ha restituito dei risultati migliore rispetto agli altri classificatori, ma comunque inferiori a 0.2. Il G3 invece, rispetto ai classificatori, ha prodotto risultati migliori, rimanendo in un range tra 0.3 e 0.5, denotando quindi che con l'aumento del numero di classi analizzate la valutazione del G3 non peggiora particolarmente. I risultati sono visibili attraverso il seguente grafico, dove possiamo osservare in particolare che ogni curva che rappresenta le accuracy dei modelli è sempre più in basso rispetto alla curva del g3.



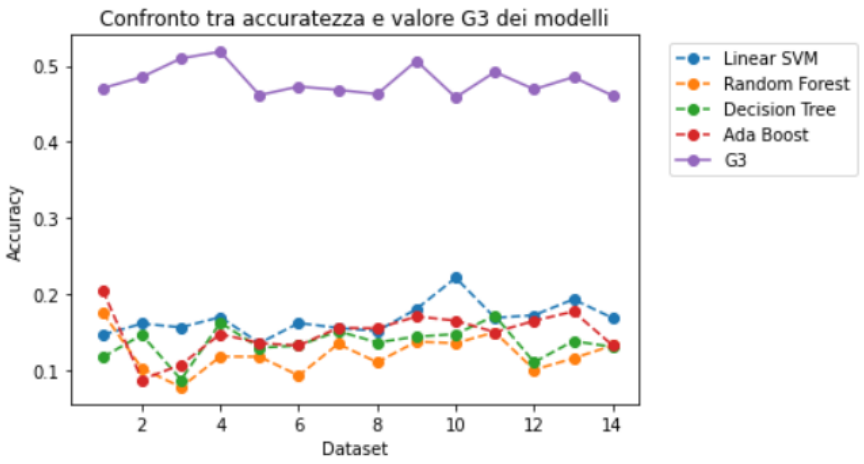
- `esperimento 2: N = 100,k = 5, sf = (10*i)`
  In questo secondo esperimento il valore cambiato è **sf**, ovvero il fattore di scala che ci indica il numero di volte che viene duplicato un determinato dataset, quindi di quante volte viene amplificato il dataset originale, ottenuto andando a moltiplicare 10 per **i**, con **i** che assume valori da 1 a 15.

Diversamente da prima, con l'aumentare del valore di sf i classificatori producono risultati di accuracy sempre maggiori, con il linearSVM come classificatore che produce risultati migliori. Invece il G3 con l'aumentare del numero di relazioni produce risultati sempre peggiori, probabilmente il g3 peggiora man mano che il dataset diventa grande, il che è probabile visto che comunque parliamo di un numero di attributi basso e un numero di classi basso, quindi spesso si ripeteranno gli stessi elementi ma classificati in modo diverso.



- `esperimento 3: N = (i*15),k = 5, sf = 10`

  Infine, nel terzo ed ultimo esperimento riguardante la generazione di dataset difficili il valore cambiato è **N**, che rappresenta il numero di tuple, ottenuto andando a moltiplicare 15 per **i**, con **i** che assume valori da 1 a 15. Sia il G3 che i classificatori non sembra risentano troppo dell'aumento del numero di tuple, producendo valori più o meno simili tra tutti i dataset analizzati. Anche in questo caso la modifica tra i vari dataset sta nel numero di tuple per dataset che aumenta man mano come per il secondo esperimento, solo che in questo caso i dataset vengono amplificati in maniera meno esponenziale perché ad aumentare sono le tuple pre duplicazione e non il numero delle duplicazioni.



La generazione di Dataset difficili ci mostra che, cambiando i parametri definiti in precedenza, i classificatori non riescono a classificare correttamente questi problemi. I classificatori utilizzati tendono, nella maggior parte dei casi, a dare risultati in output molto simili tra di loro. Tuttavia questi non sono sufficientemente soddisfacenti infatti, come si può notare stesso dai grafici, l'accuracy non supera mai il 20%. Quindi è molto semplice che il g3 risulti essere un limite superiore a tali accuratezze visto il loro basso valore, quindi passiamo ora a valutare degli studi su dataset reali che danno accuracy maggiori.

3.2 Dataset Reali

Conclusa l'analisi sui dataset difficili generati, ci siamo focalizzati sullo studio dei seguenti dataset reali:

- **Mammographic Mass Data Set**
  Questo dataset descrive la discriminazione delle masse mammografiche benigne e maligne in base agli attributi BI-RADS e all'età del paziente. La mammografia è il metodo più efficace per lo screening del cancro al seno disponibile oggi. Questo set di dati può essere utilizzato per prevedere la gravità (benigna o maligna) di una lesione di massa mammografica dagli attributi BI-RADS e dall'età del paziente. Contiene una valutazione BI-RADS, l'età del paziente e tre attributi BI-RADS insieme alla verità fondamentale (il campo di gravità) per 516 benigni e 445 masse maligne che sono state identificate su mammografie digitali a pieno campo raccolti presso l'Istituto di Radiologia del Università Erlangen-Norimberga tra il 2003 e il 2006. Ogni istanza ha una valutazione BI-RADS associata che va da 1 (decisamente benigna) a 5 (altamente indicativo di malignità) assegnato in un processo di doppia revisione da medici [11].

- **Parkinson Data Set**
  Quest'altro set di dati, invece, è stato creato da Max Little dell'Università di Oxford, in collaborazione con il National Center for Voice and Speech, Denver, Colorado, nel quale hanno registrato i segnali vocali. Lo studio originale ha pubblicato i metodi di estrazione delle caratteristiche per i disturbi vocali generali. Questo set di dati è composto da una gamma di misurazioni vocali biomediche di 31 persone, 23 con malattia di Parkinson (MdP). Ogni colonna della tabella è una particolare misura vocale e ogni riga corrisponde a una delle 195 registrazioni vocali di questi individui (colonna "nome"). Lo scopo principale dei dati è discriminare le persone sane da quelle con PD, secondo la colonna "stato" che è impostata su 0 per sani e 1 per PD.I dati sono in formato ASCII CSV, le righe del file CSV contengono un'istanza corrispondente a una registrazione vocale. Ci sono circa sei registrazioni per paziente e il nome del paziente è identificato nella prima colonna [12].

- **MoCap Hand Postures Data Set**
  Il MoCap Hand Postures Data Set è un dataset nel quale è stato utilizzato un sistema di telecamere di cattura del movimento Vicon per registrare 12 utenti che eseguivano 5 posizioni delle mani con pennarelli attaccati a un guanto per mancini. Questo set di dati può essere utilizzato per una varietà di compiti, il più ovvio dei quali è il riconoscimento della postura tramite la classificazione. Si può anche tentare l'identificazione dell'utente [13]. Il dataset è un file csv composto da interi e reali con 78095 istanze, ognuna costituita da 38 attributi. Un'intestazione fornisce il nome di ciascun attributo. Un record fittizio iniziale composto interamente da 0 dovrebbe essere ignorato. Un punto interrogativo '?' viene utilizzato per indicare un valore mancante. Una registrazione corrisponde a un singolo istante o fotogramma registrato dal sistema della fotocamera.

- **Internet Firewall Data Data Set**
  Questo set di dati è stato raccolto dai record di traffico Internet sul firewall di un'università. Il dataset è un file csv composto da valori interi e reali con 65532 istanze, ognuna costituita da 12 attributi. Gli attributi sono: Source Port,Destination Port, NAT Source Port, NAT Destination Port, Action, Bytes, Bytes Sent, Bytes Received, Packets,Elapsed Time (sec), pkts_sent, pkts_received [14].

- **Synthetic Financial Datasets For Fraud Detection**
  PaySim simula le transazioni di denaro mobile sulla base di un campione di transazioni reali estratte da un mese di registri finanziari da un servizio di

denaro mobile implementato in un paese africano. I log originali sono stati forniti da una multinazionale [15]. Questo dataset sintetico viene ridimensionato di 1/4 del set di dati originali. Il dataset è composto da file csv con tuple contenenti valori interi, stringhe e decimali ed ogni riga è costituita da 12 attributi.

- **Musk (Version 2) Data Set**
  Questo set di dati descrive un insieme di 102 molecole di cui 39 sono giudicate da esperti umani come muschi e le restanti 63 molecole sono giudicate non muschiate. L'obiettivo è imparare a prevedere se le nuove molecole saranno muschi o non muschi [16]. Il dataset è composto da file data e csv, con tuple contenenti valori interi e ogni riga è costituita da 168 attributi.

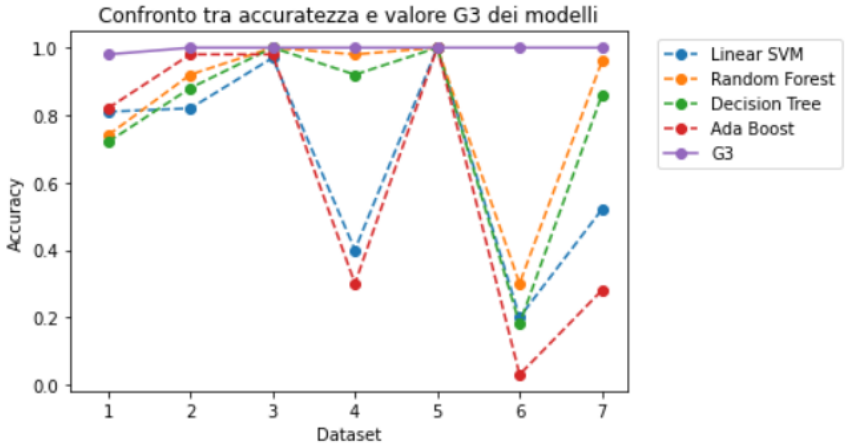
- **Letter Recognition Data Set**
  La ricerca per questo articolo ha studiato la capacità di diverse varianti dei sistemi di classificazione adattiva, in stile olandese, di imparare a individuare correttamente le categorie di lettere associate ai vettori di 16 attributi interi estratti dalle immagini di scansione raster delle lettere. L'obiettivo è identificare ciascuno di un gran numero di display pixel rettangolari in bianco e nero come una delle 26 lettere maiuscole dell'alfabeto inglese. Ogni stimolo è stato convertito in 16 attributi numerici primitivi (momenti statistici e conteggi degli archi) che sono stati poi ridimensionati per adattarsi a un intervallo di valori interi da 0 a 15. Tipicamente ci alleniamo sui primi 16000 elementi e quindi utilizziamo il modello risultante per prevedere il categoria di lettere per i restanti 4000 [17]. Il dataset è composto da file data e con tuple contenenti valori interi e ogni riga è costituita da 16 attributi.

Analisi dei risultati sui dataset reali

Nello studio dei dataset reali è stata necessaria come prima operazione una fase di data cleaning, per garantire una certa soglia di affidabilità dei dataset, poiché nelle tabelle originali ci sono diversi valori nulli che vengono visualizzati con “?”, sono stati rimossi per non compromettere lo studio perché andava ad inficiare sul valore finale del G3 e soprattutto causava errori durante le operazioni di training dei classificatori utilizzati per lo studio. Oltre ad eliminare i valori nulli per alcuni dataset è stato necessario anche convertire i valori da stringa ad interi per permettere ai classificatori di lavorare correttamente. Dopo questo lavoro di rifinitura sono stati implementati gli stessi classificatori descritti precedentemente. Tutti i valori dello studio sono presenti all’interno del seguente schema comprensivo anche di tempi di elaborazione del G3.

Data set	Ada Boost	SVM Linear	Random Forest	Decision Tree	G3	Gini	Numero righe	Numero Classi	Tempo esecuzione G3 (sec)
Mammographic Mass	0.84	0.84	0.81	0.79	0.98	0.99	205	2	0.01
Parkinsons	0.96	0.86	0.98	0.98	1	0.98	49	2	0.01
MoCap Hand Postures	0.99	0.98	1	1	1	1	16383	4	3.17
Internet Firewall Data	0.66	0.46	0.98	0.92	1	1	18710	6	5.38
Synthetic Financial Datasets For Fraud	1	1	1	1	1	1	1590655	2	289.02
Musk (Version 2)	0.01	0.19	0.29	0.21	1	0.99	119	76	0.34
Letter Recognition	0.28	0.65	0.96	0.87	1	1	5000	26	1.34

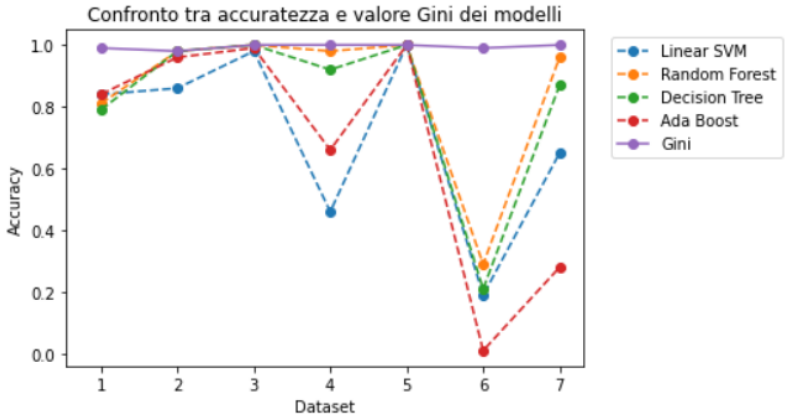
Come prima abbiamo sintetizzato tutti i risultati ottenuti in un grafico, di seguito mostriamo tali risultati, i grafici sono indicati con un numero rispetto all'ordine in cui sono stati precedentemente presentati.



Il valore del G3 solo nel primo dataset dà come valore 0.98, mentre negli altri dataset restituisce sempre il valore 1, quindi questo vuol dire che a prescindere dal valore delle varie accuratezze esso rappresenterà sempre un limite superiore visto che tocca come risultato quasi sempre il massimo. Come si può notare dal grafico, il dataset su cui i classificatori producono i risultati peggiori è il numero 6, probabilmente dovuto al numero eccessivo di classi presenti all'interno del dataset; invece quando il dataset presenta un numero di classi ridotte, notiamo che i classificatori restituiscono dei risultati migliori, come nei casi dei dataset 2 e 3. In linea generale il classificatore che si è comportato meglio rispetto agli altri è il Random Forest, riuscendo ad ottenere in due dataset un accuracy pari a 1 e in altri

tre valori superiori a 0.92. Sia il Linear SVM che il Decision Tree hanno ottenuto valori nella media, anche se il secondo presenta valori leggermente superiori del primo. Nel caso del dataset numero 5 da sottolineare, che tutti i classificatori ottengono come accuracy il valore 1, probabilmente dovuto al fatto che il numero di righe è molto elevato e le classi sono 5; tuttavia, il tempo del G3 su questo dataset è stato piuttosto lungo, dovuto proprio al numero elevato di tuple. Il tempo di esecuzione invece su questi dataset è stato inferiore ad un secondo, dovuto anche al fatto che la maggior parte dei dataset presenta un ridotto numero di tuple e di classi, rendendo quindi veloce l'esecuzione del G3.

Nella parte di teoria abbiamo introdotto anche l'impurità come metrica poichè volevamo provare se anche altre metriche oltre il g3 potessero essere utili allo scopo che ci siamo preposti, nel caso ovviamente in cui il g3 fosse stato considerato utile alla causa dopo i nostri esperimenti. Siccome gli esperimenti sul g3 hanno dato buoni risultati, e possiamo dichiarare effettivamente il g3 come limite superiore ora proviamo con gli stessi dataset reali e con gli stessi modelli a fare il confronto con l'impurità.



Con l'impurità di Gini abbiamo ottenuto dei risultati simili a quelli del g3, ma era una cosa che si poteva predire visto comunque la somiglianza concettuale delle metriche come spiegato nel capitolo di teoria. Quindi proprio come il g3 è risultato essere un parametro utile a tracciare un limite superiore per le accuratezze dei modelli, avendo valori molto simili anche questa metriche è utilizzabile. Nella maggior parte dei casi, come nel G3, abbiamo ottenuto come valore 1, mentre nei casi in cui il numero di righe è più piccolo rispetto agli altri dataset il risultato ottenuto da Gini restituisce un valore poco inferiore ad 1, ovvero 0.98 e 0.99, nei dataset 1,2 e 6. Per quanto riguarda i tempi di elaborazione non sono stati trascritti nella tabella ma come i risultati anche i tempi sono molto simili anche perchè l'algoritmo implementato per Gini ha la fase di ricerca delle tuple uguale al g3 ciò che cambia è solo il calcolo finale in cui calcoliamo il risultato con la funzione di Gini anziché cercare il valore minimo delle tuple da eliminare.

Conclusioni

Il nostro progetto si propone di verificare la teoria di Marie Le Guilly, Jean-Marc Petit, Vasile-Marian Scuturici ripresa dall'articolo “Evaluating Classification Feasibility Using Functional Dependencies”, la quale dichiara l’esistenza di una relazione tra le dipendenze funzionali e i modelli di machine learning. Il punto su cui, sia gli autori del documento che il nostro gruppo, ci siamo focalizzati è quello di utilizzare le misure delle RFD che rilassano sull'extend che potessero rappresentare dei valori utili anche per avere informazioni sui modelli. Quindi, il primo passo è stato replicare lo stesso tipo di esperimento fatto da loro e testare il valore del g3 come limite superiore dell’accuratezza dei vari modelli. Dunque abbiamo implementato lo stesso algoritmo di generazione dei datasets, andando a fare diversi esperimenti, aumentando i parametri per ottenere una varietà di dataset più grande e per capire se effettivamente ci sono alcune situazioni in cui il g3 potrebbe essere considerato limite e in altre no. Conclusi gli esperimenti, notiamo che il g3, anche variando di molto, tende sempre ad essere un valore limite rispetto ai risultati restituiti dai classificatori. Successivamente, abbiamo spostato l’attenzione sulle effettuare i test con dataset reali, sperimentando 7 dataset con differenti caratteristiche (come grandezza, quantità di tipi di classi e tipologie di attributi) per avere sempre una diversità di esperimenti, in maniera tale da testare la flessibilità della teoria. Anche in questo caso siamo riusciti ad ottenere conferme, infatti il g3 può, secondo i nostri esperimenti, essere definito come valore limite.
Fin dall’inizio si è parlato in maniera più generica del rapporto tra le dipendenze funzionali e i modelli di machine learning, in particolare dell’utilizzo di varie metriche. Quindi, seguendo le loro sperimentazioni sul g3, abbiamo testato anche il valore dell’impurità, utilizzando come funzione di calcolo la cosiddetta impurità di Gini. Con l’impurità abbiamo potuto notare la sua tendenza, così come per il g3, ad essere un limite superiore per le accuratezze dei vari modelli. In conclusione, dopo tutti gli esperimenti effettuati, possiamo dedurre che il collegamento esiste e l’utilizzo delle metriche delle RFD sull’extent può essere utile a chi lavora con i modelli di machine learning a ottenere informazioni prima di utilizzare tali modelli. Inoltre, è possibile recuperare anche informazioni che possono suggerire se esiste o meno una funzione e quindi se è il caso di costruire un modello, oppure si possono anche identificare rilevanti controesempi, cioè tuple nel set di dati che causerebbero il fallimento di un classificatore. La presenza di questi valori “limite” è utile per gli esperti poiché permetterebbe loro di prendere decisioni su come procedere conoscendo quello che è il massimo potenziale a cui si può arrivare oppure procedere con ulteriore pulizia dei dati o ancora effettuare altri tipi di miglioramenti al dataset. In futuro crediamo che queste metriche possano essere applicate per trarre informazioni sui dataset prima di implementare i modelli in modo da evitare implementazioni inutili e procedere preventivamente a effettuare una ulteriore fase di

pre-processing, soprattutto crediamo che una tecnica combinata di metriche possa aumentare ancora di più la precisione di limitazione di cui finora abbiamo parlato.

Riferimenti

[1] D. Maier, The theory of relational databases, Computer Science Press, Rockville, 1983. - H. Mannila & K.-J. Raiha, The design of relational databases, Addison-Wesley, Wokingham, 1992. - Y. Huhtala, J. K¨arkk¨ainen, P. Porka & H. Toivonen, Efficient Discovery of Functional and Approximate Dependencies Using Partitions (Extended version), Report C-1997-79, Dept. of Computer Science, University of Helsinki, 1997.

[2] “Impurity measures in databases”, Dan A. Simovici, Dana Cristofor, Laurentiu Cristofor, 2001

[3] M. Le Guilly, J.M. Petit, V.M. Scuturici, Evaluating Classification Feasibility Using Functional Dependencies. Transactions on Large-Scale Data- and Knowledge-Centered Systems, Springer Berlin / Heidelberg, 2020, pp.132-159. ff10.1007/978-3-662-62271-1_5ff. ffhal-03108892

[4] scikit-learn developers “Support Vector Machines”,
“https://scikit-learn.org/stable/modules/svm.html#svm-classification”

[5] scikit-learn developers “Random Forest”,
“https://scikit-learn.org/stable/modules/ensemble.html#forest”

[6] scikit-learn developers “Decision Tree”
“https://scikit-learn.org/stable/modules/tree.html#tree”

[7] scikit-learn developers “AdaBoost”,
“https://scikit-learn.org/stable/modules/ensemble.html#adaboost”

[8] L. Caruccio, V. Deufemia and G. Polese, "Relaxed Functional Dependencies—A Survey of Approaches," in IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 1, pp. 147-165, 1 Jan. 2016, doi: 10.1109/TKDE.2015.2472010.

[9] Dua, Dheeru and Graff, Casey “Machine Learning Repository” 2017

[10] contributori di Wikipedia, “Kaggle”

[11] M. Elter, R. Schulz-Wendtland and T. Wittenberg (2007), The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical Physics 34(11), pp. 4164-4172, “https://archive.ics.uci.edu/ml/datasets/Mammographic+Mass”

[12] 'Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection', Little MA, McSharry PE, Roberts SJ, Costello DAE, Moroz IM. BioMedical Engineering OnLine 2007,
“https://archive.ics.uci.edu/ml/datasets/Parkinsons”

[13] Dua, D. e Graff, C. “MoCap Hand Postures Data Set” (2019). UCI Machine Learning Repository,
“https://archive.ics.uci.edu/ml/datasets/MoCap+Hand+Postures”

[14] Dua, D. e Graff, C. “Internet Firewall Data Data Set” (2019). UCI Machine Learning Repository, “https://archive.ics.uci.edu/ml/datasets/Internet+Firewall+Data”

[15] E. A. Lopez-Rojas , A. Elmir, and S. Axelsson. "PaySim: A financial mobile money simulator for fraud detection". In: The 28th European Modeling and Simulation Symposium-EMSS, Larnaca, Cyprus. 2016,
“https://www.kaggle.com/ealaxi/paysim1”
[16] Dua, D. e Graff, C. “Musk (Version 2) Data Set” (2019). UCI Machine Learning Repository, “https://archive.ics.uci.edu/ml/datasets/Musk+(Version+2)”
[17] Dua, D. e Graff, C. “Letter Recognition Data Set” (2019). UCI Machine Learning Repository, “https://archive.ics.uci.edu/ml/datasets/Letter+Recognition”