Università degli Studi di Padova

Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea in
Ingegneria dell'informazione

Anomaly Detection: analisi teorica e indagine sul metodo Isolation Forest

Relatore:
Prof. Gian Antonio Susto

Laureando:
Alberto Bressan
2000268

Anno Accademico 2022/2023
Data 29/09/2023

Abstract

Questo lavoro di tesi si concentra sull'analisi teorica e l'implementazione pratica dell' Anomaly detection, intesa come rilevamento di anomalie, tramite metodi di machine learning, per identificare eventi che risultano sospetti in relazione al modello di comportamento stabilito. Questa pratica trova importanti applicazioni in diversi settori, come quello industriale, finanziario e nel rilevamento di frodi. La tesi inizia con un'esaustiva revisione della letteratura sull'anomaly detection, esplorando i diversi approcci e algoritmi proposti nel campo. Si affrontano le sfide teoriche associate alla rilevazione delle anomalie, alla scelta dei metodi di apprendimento automatico più adatti e alla gestione della complessità dei dati. Viene poi presentato e approfondito il metodo dell'Isolation Forest, algoritmo di Anomaly detection basato sugli alberi binari, che permette il rilevamento di anomalie con limitati requisiti di memoria anche per quantità ingenti di dati. Viene illustrato il funzionamento teorico dell'Isolation Forest, esplorando i principi fondamentali e ne vengono analizzate le prestazioni reali, in relazioni agli altri metodi di anomaly detection descritti.

Indice

Capitolo 1

Introduzione

L'era dell'informazione ha trasformato il mondo in cui viviamo determinando una crescita della quantità di dati che vengono raccolti e delle modalità con le quali questi vengono utilizzati. Questa ricchezza di dati rappresenta un'arma a doppio taglio, poiché offre opportunità senza precedenti, ma allo stesso tempo accresce esponenzialmente la complessità delle sfide legate alla sicurezza e all'integrità dei dati. In questo contesto, la rilevazione delle anomalie (anomaly detection) ricopre un ruolo cruciale e in rapida crescita.

Nell'analisi dei dati l'anomaly detection (AD) è l'identificazione di eventi, osservazioni e istanze che si discostano sensibilmente dalla maggior parte dei dati, questa viene operata tramite l'uso di metodi e tecniche del machine learning e del deep learning e può essere applicata con modalità differenti a molti settori. In ambito medico, ad esempio, l'identificazione di anomalie permette di diagnosticare malattie o rilevare disturbi non evidenti nei pazienti, in ambito finanziario viene utilizzata per individuare eventuali frodi o movimenti sospetti di denaro. L'AD trova poi numerose applicazioni in contesti di ricerca scientifica, come la scoperta di nuovi corpi celesti, e industriali, come la manutenzione preventiva.

Questo lavoro di tesi descrive, nella prima parte, l'anomaly detection a partire dalla distinzione tra le diverse tipologie di anomalie, introducendo i principali metodi descritti in letteratura e le tecniche fondamentali per l'analisi delle prestazioni. Nella seconda parte della tesi viene approfondito il metodo Isolation Forest (iForest). Inizialmente introdotto da Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou nel 2008 [18] si è dimostrato essere uno dei metodi più efficaci ed efficienti nell'ambito dell'identificazione delle anomalie ed ha aperto la strada a diverse implementazioni, che ne hanno migliorato le prestazioni e le possibilità di applicazione a ambiti specifici. Nella parte finale si utilizza lo studio di Falcão

et al. (2019) [10], che applica i metodi descritti nella tesi per l'anomaly detection in dataset reali, per confrontare le prestazioni dei diversi metodi introdotti e mostrare l'ottimalità di iForest.

Capitolo 2

Anomaly Detection

In questo capitolo verranno definiti i concetti fondamentali e illustrati i principali metodi dell'anomaly detection e le tecniche di valutazione delle prestazioni di questi.

2.1 Anomalie

Esistono diverse definizioni di anomalia, che assumono significati differenti in base all'ambito di applicazione e che determinano concezioni diverse degli algoritmi di anomaly detection. Una delle più utilizzate è quella proposta da Hawkins et al. nel 1980 [13]:

"*Un'anomalia è un'osservazione che devia così tanto da altre osservazioni, da far scaturire il sospetto che essa sia stata generata da un meccanismo differente*".

Questa è una definizione generica che può essere applicata a diversi tipi di anomalia e non fornisce informazioni sul metodo di identificazione. A partire da essa, come descritto da Barbarbariol et al. (2022) [3], si può dedurre che un modello di AD deve misurare la deviazione fra punti, di conseguenza ogni punto ha una probabilità associata di essere un'anomalia. Nel caso di osservazione anomale, dunque, esiste un meccanismo diverso che implica l'esistenza di diverse distribuzioni di probabilità associate agli outliers e agli inliers.

In 2.1 si può osservare un esempio di rappresentazione di un dataset in due dimensioni. N_1 e N_2 sono le regioni normali, dato che vi si trovano la maggior parte dei dati, i punti O_1, O_2 e quelli appartenenti alla regione O_3 sono considerati anomalie, essendo sufficientemente distanti dalle regioni normali.

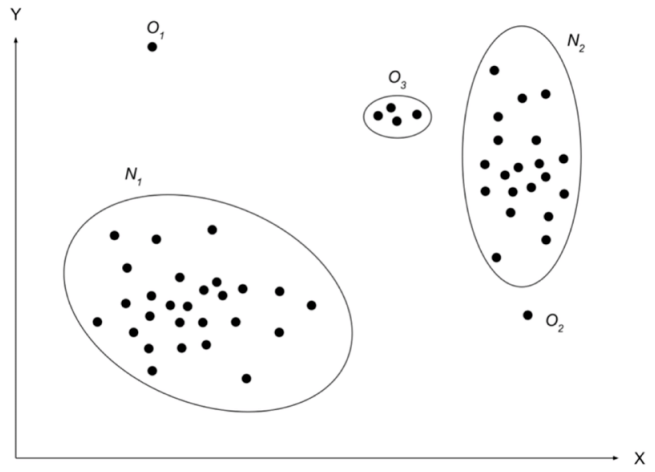


Figura 2.1: Rappresentazione di anomalie in un dataset a due dimensioni

2.1.1 Tipi di anomalie

La classificazione delle anomalie non risulta univoca in letteratura, tuttavia quella maggiormente riconosciuta, ad opera di Chandola et al. (2009), individua tre principali categorie: "point anomalies", "contextual anomalies" e "collective anomalies".

Point anomalies

Quando una singola istanza si discosta in modo significativo dal resto dei dati, essa è considerata un punto anomalo. In Fig. 2.1, O_1, O_2 e i dati appartenenti al gruppo O_3 possono essere considerati punti anomali, dato che si trovano al di fuori delle regioni normali.

Considerando l'ambito medico, si può proporre un esempio di point anomaly. Supponendo di avere una serie di misurazioni della pressione sanguigna di un paziente generalmente stabili in un range di valori normali, un punto anomalo potrebbe essere una determinata misurazione che si discosta notevolmente dalle altre, nonostante non siano avvenuti cambiamenti significativi nella condizione del paziente. L'identificazione di questa anomalia può portare a diagnosticare eventuali malattie o disturbi.

Contextual anomalies

Si considerano "anomalie di contesto" i dati che risultano anomali solo in un determinato contesto. Il contesto deve essere specificato nella formulazione del problema ed è definito dalla natura dei dati.

Nella fig 2.2, ad esempio, è rappresentata una serie temporale di temperature in una area specifica nel corso di alcuni anni. In questo caso le istanze t_1 e t_2 coincidono, tuttavia solo t_2 viene identificata come anomalia, non essendo coerente coi dati appartenenti al contesto.

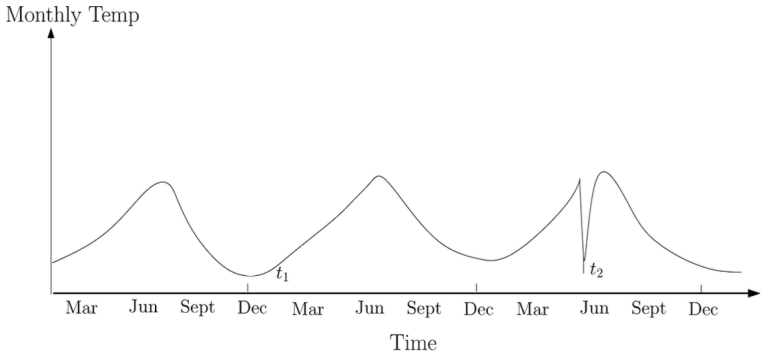


Figura 2.2: Esempio di anomalia di contesto Chandola et al. (2009)

Collective anomalies

Si identificano come "collective anomalies" i gruppi di dati che risultano anomali rispetto al dataset. In questo caso le singole istanze appartenenti al gruppo potrebbero essere coerenti, ma apparire anomale nel caso in cui si verificassero collettivamente. Questa tipologia di anomalie può ricorrere solo in collezioni di dati in cui le istanze sono tra loro correlate.

Ad esempio, il caso in cui tutte le aziende di uno stesso settore registrino contemporaneamente un calo di vendite, può essere considerato una collective anomaly, mentre non è necessariamente anomalo il medesimo evento per una singola azienda.

2.2 Metodi di anomaly detection

L'identificazione delle anomalie può essere effettuata tramite l'uso di tecniche diverse, la scelta di quella ottimale viene effettuata innanzitutto in base alla natura dei dati in ingresso. L'*input* dei metodi di anomaly detection è una collezione di dati (*dataset*) costituita da istanze caratterizzate da degli attributi (chiamati anche caratteristiche, dimensioni, campi o variabili). I dati *univariati* sono quelli caratterizzati da un solo attributo, mentre si parla di dati *multivariati* se posseggono molteplici attributi. Nonostante molte delle tecniche di rilevamento delle anomalie utilizzino principalmente dati indipendenti, dove non si presume l'esistenza di relazioni intrinseche tra essi, è possibile applicare metodi di AD anche

a dataset le cui istanze siano dipendenti, come nel caso di dati con componenti temporali.

La natura dei dati di addestramento di un modello determina una prima distinzione fra i metodi di anomaly detection.

- I metodi ***supervisionati*** richiedono un dataset di apprendimento in cui ogni dato è etichettato come normale o anomalo. A partire dai dati di addestramento si costruisce un modello predittivo che assegna ad ogni nuova istanza una delle due etichette. La classificazione del dataset di addestramento risulta essere molto costosa in termini di tempo e risorse, poichè deve essere spesso effettuata manualmente. L'etichettatura delle anomalie, tipicamente, richiede un maggiore livello di complessità rispetto a quella dei dati normali, data la natura dinamica delle prime.

- I metodi ***semi-supervisionati*** richiedono che solo una parte dei dati del dataset di apprendimento sia etichettata, solitamente solo le istanze normali fanno parte dei dati etichettati, esistono tuttavia alcuni metodi in cui vengono utilizzate le anomalie come dati etichettati. L'anomaly detection semi-supervisionata risulta, dunque, meno dispendiosa di quella supervisionata e di conseguenza è più ampiamente utilizzata.

- I metodi ***non supervisionati***, infine, non richiedono che i dati siano etichettati, ma assumono che il numero di anomalie sia molto inferiore rispetto a quello di dati normali.

I metodi di anomaly detection differiscono anche per le modalità con cui le anomalie vengono segnalate. Alcuni metodi assegnano un punteggio (*anomaly score*) ad ogni istanza, in base alla probabilità che essa sia un'anomalia, mentre altri assegnano un'etichetta (*label*) (*normale* o *anomalo*). Nel primo caso è possibile scegliere un valore di soglia, per poter identificare le anomalie più rilevanti, in maniera diretta, mentre nel secondo caso si può fare indirettamente modificando i parametri di etichettatura.

Come anticipato in 2.1, in base alla definizione di anomalia utilizzata, esistono diversi algoritmi di anomaly detection, che associano alla stessa anomalia punteggi diversi. In questa sezione verranno descritti i principali approcci all'anomaly detection appartenenti a 6 classi. Non esistendo una classificazione univoca dei metodi, alcuni algoritmi potrebbero appartenere a diverse classi contemporaneamente.

2.2.1 Algoritmi statistical-based

Anscombe e Guttman (1960) [2] danno la seguente definizione di anomalia:

"*Un'anomalia è un'osservazione che si sospetta essere parzialmente o interamente irrilevante, poichè non è generata dal modello stocastico assunto.*"

A partire da questa si può assumere che le istanze normali si trovino nelle regioni ad alta probabilità del modello stocastico, mentre le anomalie si trovino in quelle a bassa probabilità. Gli algoritmi di outlier detection che si basano su metodi statistici sono costituiti da due fasi, la fase di addestramento, in cui viene adattato un modello statistico al dataset, e la fase di test, in cui si osserva se una nuova istanza appartiene al modello oppure no.

I modelli statistici possono utilizzare tecniche *parametriche* o *non parametriche*.

Le *tecniche parametriche* assumono di conoscere la distribuzione statistica e stimano i parametri a partire dai dati in esame. Nel modello Gaussiano, ad esempio, si considerano dati generati da una distribuzione Gaussiana e si stimano i parametri utilizzando il metodo della massima verosimiglianza, l'*anomaly score* di ogni istanza viene poi assegnato calcolando la distanza tra la media stimata e l'istanza stessa. Si sceglie, infine, un valore di soglia che separa le istanze normali da quelle anomale. In fig 2.3 si può osservare un esempio di distribuzione Gaussiana, in questo caso si è scelto 3σ come valore di soglia, dove σ è la deviazione standard della distribuzione, dunque tutti i dati che si trovano oltre questa soglia saranno considerati anomali.

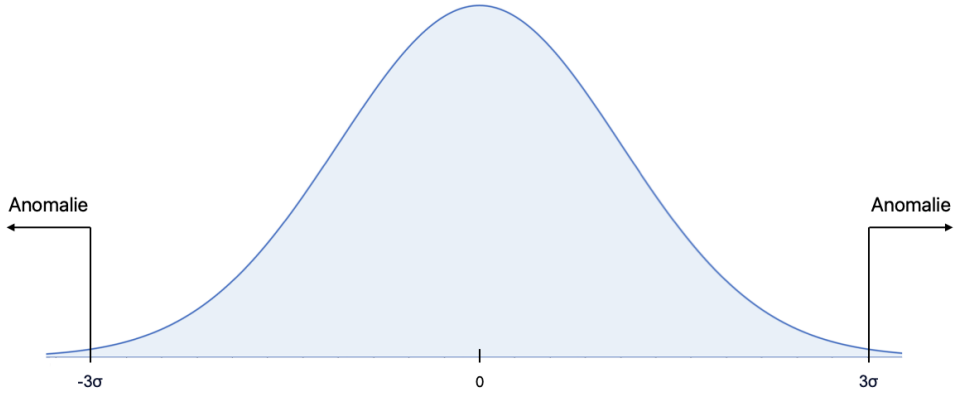


Figura 2.3: Esempio distribuzione normale con soglia 3σ

Nelle *tecniche non parametriche*, invece, il modello non è assunto a priori, ma viene generato a partire dai dati normali, il punteggio di anomalia viene assegnato misurando la deviazione tra l'istanza e il modello. Degli esempi di queste tecniche sono i modelli basati su istogrammi. In questi si crea un istogramma a partire

dalla distribuzione dei dati normali, l'istogramma suddivide l'intervallo dei dati in *bin* o intervalli discreti, ogni *bin* nell'istogramma contiene un conteggio delle istanze di dati che rientrano nell'intervallo corrispondente. Ogni nuova nuova istanza da analizzare viene rappresentata nell'istogramma, se l'istanza in esame cade in uno dei *bin* essa viene considerata normale. Al contrario, se l'istanza non rientra in nessun *bin* o cade in un *bin* con una bassa frequenza, viene classificata come anomala.

Il principale difetto dei modelli statistici è che essi si basano sull'ipotesi che tutti i dati in esame siano generati da un'unica distribuzione statistica, ciò tuttavia non risulta sempre vero, soprattutto nel caso di dataset reali. Anche nel caso in cui l'ipotesi possa essere ragionevole, risulta comunque complesso individuare la distribuzione corretta, specialmente per collezioni di dati multidimensionali.

2.2.2 Algoritmi distance-based

Si consideri la seguente definizione di anomalia *distance-based* (DB), fornita da Knorr et al. (2000) [16]:

"*Un'istanza O in un dataset T è una $DB(p, D)$ − anomalia se almeno un numero p di oggetti di T si trova ad una distanza maggiore di D da O*"

Gli algoritmi *distance-based* si basano su questa e consistono nell'analizzare la regione con raggio D intorno all'istanza O considerata. Nel caso in cui il numero di punti in essa non fosse sufficientemente grande O viene identificata come anomalia. In Fig. 2.4, ad esempio, il punto O può essere identificato come un'anomalia, considerando $p \leq 8$. Questa tipologia di algoritmi è applicabile a dataset di dimensione k, e risulta efficiente anche per k grandi (e.g $k \geq 5$), come evidenziato da Knorr et al. (2000) [16].

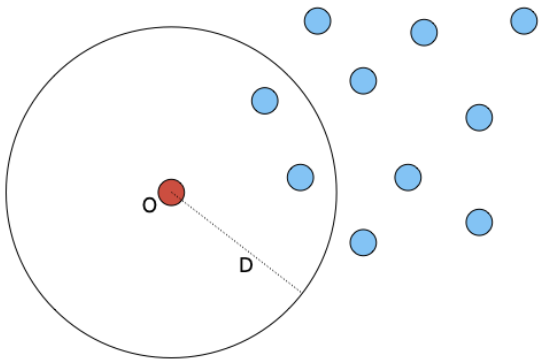


Figura 2.4: Esempio di anomalia basata sulla distanza in 2D

Esistono diverse categorie di metodi basati sulla distanza, che ottengono complessità diverse a seconda della dimensione k e del numero totale N di dati presenti nel dataset. Gli algoritmi *index-based*, ad esempio, consistono nel contare il numero di punti presenti nella regione di raggio D di ogni istanza O, quando si trovano almeno $(M + 1)$ punti vicini, dove $M = N(1 - p)$ è il numero massimo di oggetti presenti nella regione di un outlier, O può essere etichettato come non anomalo. Questa tecnica ha complessità al caso peggiore $O(kN^2)$, che risulta essere la migliore per gli algoritmi *distance-based*.

Gli algoritmi *nested-loop* permettono di ottenere le stesse prestazioni, ma eliminano il costo legato alla costruzione dell'indice, utilizzando una progettazione a blocchi più efficiente. Si possono, infine, utilizzare algoritmi *cell-based*, che hanno complessità lineare a N, ma esponenziale rispetto a k, sono dunque ottimali per dataset con dimensione limitata.

2.2.3 Algoritmi density-based

I metodi basati sulla densità si basano sul presupposto che le anomalie all'interno di un dataset siano meno comuni rispetto alle istanze corrette. Questa tipologia di algoritmi, dunque, identifica un'istanza come anomala quando essa si trova in una zona a bassa densità. La misura della densità della zona circostante ad ogni istanza dipende da due parametri, la distanza, ossia il raggio della zona considerata, e il numero di punti nell'area.

Nel caso in cui i dati abbiano molteplici zone con densità diverse, gli algoritmi *density based* non ottengono risultati soddisfacenti. In Fig 2.5, ad esempio si può osservare che l'insieme C_1 ha una densità inferiore a quella di C_2, in questo caso la densità circostante a p_1 potrebbe risultare simile a quella di C_1 e di conseguenza p_1 verrebbe classificata come istanza normale, invece che anomala. Al contrario p_2 verrebbe identificata come istanza anomala, dato che risulta essere isolata rispetto alle altre.

Per ovviare a questo problema Breunig et al. (2000a) [6] fanno una distinzione fra anomalie globali, definite in 2.2.2, e anomalie locali, ossia istanze che risultano anomale solo rispetto ai dati ad esse vicini (e.g. il punto p_1 in Fig 2.5). Il "vicinato" di un'istanza viene individuato scegliendo i k punti più vicini ad essa, con k adeguato. Gli algoritmi basati su questa definizione assegnano ad ogni istanza X un *anomaly score*, chiamato *Local Outlier Factor (LOF)*, calcolando la densità relativa di ogni istanza nel seguente modo:

$$LOF(X) = \frac{\text{Densità media delle istanze vicine}}{\text{Densità di X}} \tag{2.1}$$

dove la densità di X è l'inverso della distanza media fra i k punti più vicini e la densità media delle istanze vicine considera i k punti del "vicinato". I dati anomali, dunque, saranno quelli con un valore LOF maggiore.

Papadimitriou et al. (2003) [23] introducono una tecnica di anomaly detection chiamata LOcal Correlation Integral (LOCI) che utilizza una variazione del LOF e permette di individuare non solo le singole istanze anomale, ma anche gruppi di anomalie.

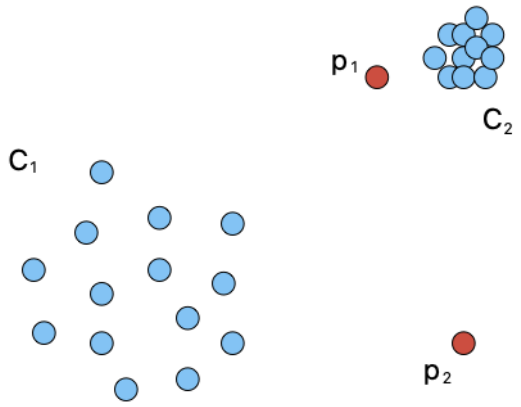


Figura 2.5: Esempio di dataset in 2D con zone a densità diversa

2.2.4 Algoritmi angle-based

I metodi descritti finora utilizzano in maniera diversa i concetti di prossimità e distanza, tuttavia alcuni studi, come Aggarwal et al. (2001) [1], mostrano che queste misure perdono significato, al crescere della dimensionalità dei dataset, in particolare si osserva che la differenza relativa tra il punto più distante e quello più vicino converge a 0, quando la dimensionalità $d \to 0$.

Kriegel et al. (2008) [17] introducono i metodi angle-based per ovviare a questo problema e ottenere risultati accettabili per l'identificazione di anomalie in dataset ad elevata dimensionalità. Il metodo utilizza le misure degli angoli fra i vettori che collegano un punto del dataset e una coppia qualsiasi di dati (e.g gli angoli α, β e γ in fig 2.2.4). L'intuizione alla base di questa tecnica può essere spiegata facendo riferimento alla figura 2.2.4, si osserva che il punto p_1 è un punto normale e gli angoli formati con tutte le coppie di dati, ad esempio gli angoli α e β, hanno un'alta varianza tra di loro, al contrario gli angoli formati dal punto anomalo p_2 sono fra loro più simili, data la distanza elevata del punto dal gruppo

di istanze normali. Si può assegnare, dunque, un *anomaly score* ad ogni punto misurando la varianza degli angoli formati fra l'istanza in esame ed ogni coppia di punti del dataset, i dati con varianza minore saranno più probabilmente anomali.

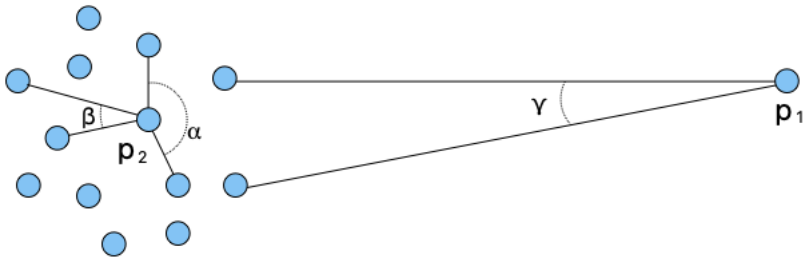


Figura 2.6: Esempio di angle-based anomaly detection su dataset in 2D

2.2.5 Algoritmi clustering-based

Il clustering è un processo di analisi dei dati mediante il quale vengono raggruppate istanze di dati in base alle loro caratteristiche comuni. I *cluster* vengono costruiti minimizzando le distanze tra i punti all'interno di essi e massimizzando quelle fra i differenti *cluster*. A partire da questa tecnica si possono costruire diversi algoritmi di anomaly detection che valutano ogni istanza confrontandola con il *cluster* a cui essa appartiene, a differenza dei metodi *distance-based* (2.2.2) in cui l'istanza viene analizzata rispetto ai punti vicini ad essa, Kang (2018) [15].

I metodi basati sul clustering si possono dividere in tre gruppi, alla base dei quali ci sono tre diverse ipotesi Chandola et al (2009) [7]:

Ipotesi 1: I dati normali appartengono ad un cluster mentre le anomalie non vi appartangono.

I metodi basati su questa ipotesi applicano algoritmi di *clustering* noti e individuano ogni istanza che non appartiene ad un cluster come anomala. Per fare ciò il processo di clustering deve essere operato tramite algoritmi che non forzino ogni elemento del dataset in un cluster. Il principale difetto di utilizzare questi metodi per l'anomaly detection è che essi non sono ottimizzati per individuare gli outlier, ma per costruire il cluster di dati.

Ipotesi 2: I dati normali si trovano vicino al baricentro del cluster più vicino, mentre le anomalie ne sono distanti.

Gli algoritmi basati sulla seconda ipotesi si compongono di due fasi, nella prima il dataset viene diviso in cluster, nella seconda viene assegnato l'*l'anomaly*

score ad ogni istanza, misurando la distanza fra essa e il baricentro del cluster più vicino. I dati con AS maggiore, di conseguenza, saranno più probabilmente delle anomalia.

Ipotesi 3: I dati normali appartengono a cluster grandi e densi mentre le anomalie appartengono a cluster piccoli e rarefatti.

In questo caso viene scelto un valore di soglia per il parametro di dimensione o densità dei cluster e identificate come anomalie le istanze appartenenti a cluster al di sotto della soglia

Confronto tecniche Il confronto diretto tra i vari metodi di anomaly detection non è possibile, dato che essi utilizzano parametri differenti, si analizzano, dunque, le prestazioni legate all'applicabilità e all'utilità nei diversi ambiti di utilizzo. Mandhare e Idata (2017) forniscono una comparazione fra i metodi cluster-based, distance-base e density-based, riassunta in fig 2.7.

Parameters	**Techniques / algorithms.**		
	Cluster based	***Distance based***	***Density based***
Compu-tation cost	Low	Low	High
Efficiency	Very efficient	Efficient	Efficient
High-dimensional data	Applicable	Applicable	Applicable.
Complexity	Less complex	Moderately Complex	Highly complex

Figura 2.7: Confronto tra metodi distance, density e cluster based

Da questo appare evidente che i metodi *cluster based* siano i migliori in relazione a efficienza e complessità, mentre quelli basati sulla densità risultano essere i più complessi. Un confronto più esaustivo, che considera l'analisi delle prestazione di tutti i metodi introdotti in questo capitolo, mediante l'applicazione a dataset reali, viene effettuato nel capitolo 4.

2.2.6 Algoritmi isolation-based

L'approccio più recente è quello basato sull'isolamento, che offre prestazioni notevoli essendo stato sviluppato proprio con lo scopo di risolvere problemi di anomaly detection, a differenza dei metodi descritti sopra, che applicano tecniche non specifiche al problema. Esso si basa su due proprietà fondamentali delle anomalie,

ossia che esse sono presenti in numero inferiore e differiscono di molto rispetto ai dati normali in un dataset. Essendo le anomalie "poche e differenti", dunque, risulta più semplice isolare un outlier piuttosto che un inlier. Uno dei metodi appartenente a questa categoria è l'Isolation Forest Liu et al. (2008) [18], basato sulla struttura di albero binario. Il metodo consiste nel suddividere ricorsivamente il dataset in due parti fino ad aver isolato ogni dato, il numero di divisioni necessarie ad isolare un'istanza è l'anomaly score dell'istanza stessa. Le istanze con un punteggio più basso avranno più probabilità di essere anomalie, dato che sono state isolate più facilmente, secondo le proprietà descritte sopra. Esistono diverse iterazioni del metodo, che modificano le tecniche di divisione e quelle di assegnazione dell'anomaly score. Il metodo Isolation Forest viene descritto e analizzato più in specifico nel capitolo 3.

2.3 Valutazione delle prestazioni

Come illustrato nel capitolo 2.2.6, esiste un'ampia gamma di approcci nell'ambito dell'anomaly detectione e il numero di algoritmi disponibili è in continua e rapida espansione. È diventato, pertanto, essenziale essere in grado di valutare le prestazioni di tali algoritmi, sia per convalidarne l'efficacia, sia per effettuare confronti tra i differenti approcci.

2.3.1 Metriche di valutazione delle prestazioni

Nel caso di anomaly detection basata su classificazione binaria le istanze del dataset vengono separate in due classi: positiva (P) e negativa (N). Si distinguono quattro possibili risultati,in particolare due risultati di classificazione corretta, vero positivo (VP) e vero negativo (VN), e due risultati di classificazione errata, falso positivo (FP) e falso negativo (FN). A partire da questi si costruisce la matrice 2x2 in fig 2.8, chiamata *matrice di confusione*, da cui si definiscono le metriche di valutazione illustrate di seguito.

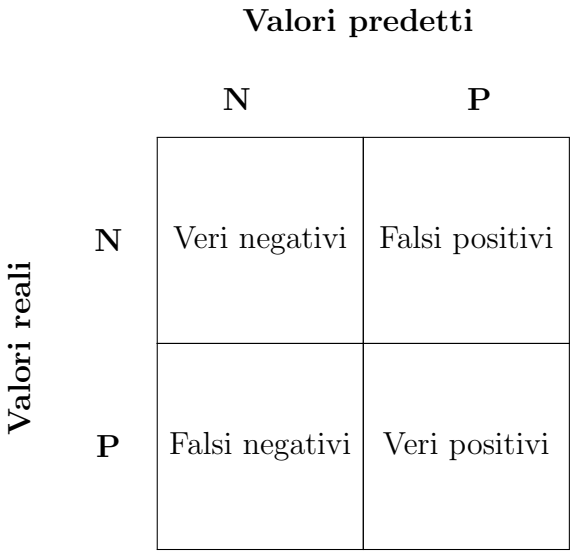


Figura 2.8: Matrice di confusione per classificazione binaria

Accuratezza. L'accuratezza è il rapporto fra il numero di predizioni corrette e il numero di predizioni totali del modello:

$$Accuratezza = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{VN} + \text{FP} + \text{FN}}. \tag{2.2}$$

Questa metrica non risulta efficace per modelli in cui le classi non sono bilanciate, dato che l'accuratezza risulterebbe elevata anche se il modello predicesse sempre la classe maggioritaria, o in casi in cui l'impatto di falsi positivi o falsi negativi è determinante, ad esempio nella diagnosi medica.

Sensibilità e specificità. La sensibilità (o *recall*) e la specificità misurano rispettivamente la capacità del modello di individuare correttamente i dati positivi e quelli negativi.

$$Sensibilità = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{2.3}$$

$$Specificità = \frac{\text{VN}}{\text{VN} + \text{FP}}. \tag{2.4}$$

Precisione. La precisione misura quanto le previsioni positive fatte dal modello siano effettivamente corrette, calcolando la proporzione di veri positivi rispetto al numero totale di veri positivi e falsi positivi:

$$Precisione = \frac{\text{VP}}{\text{VP} + \text{FP}} \tag{2.5}$$

La precisione è particolarmente utile quando l'obiettivo è limitare i falsi positivi, cioè quando è importante evitare di classificare erroneamente esempi negativi come positivi.

F-score. La metrica *F-score* utilizza come parametri precisione (p) e recall (r) di un modello e ne calcola la media armonica. La formula generale è la seguente:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{p} \cdot \text{r}}{(\beta^2 \cdot \text{p}) + \text{r}} \tag{2.6}$$

dove solitamente si considera $\beta = 1$ ($F_1 - score$), caso in cui precisione e recall sono bilanciati, mentre viene favorito il parametro di precisione con $\beta > 1$, e quello di recall altrimenti.

La F1-score è particolarmente utile quando si ha a che fare con classi sbilanciate, dove ci sono molte più istanze di una classe rispetto all'altra, essa fornisce una valutazione più equilibrata delle prestazioni del modello.

2.3.2 Receiver operating characteristic (ROC)

La curva ROC è una rappresentazione grafica in due dimensioni che illustra la capacità del modello di discriminare tra le due classi al variare della soglia di classificazione. Ogni punto sulla curva ROC rappresenta un diverso equilibrio tra la capacità del modello di identificare correttamente gli esempi positivi e la probabilità di classificare gli esempi negativi come positivi. Gli assi del grafico rappresentano rispettivamente la sensibilità (*asse y*) e la specificità (*asse x*) del modello, come raffigurato in fig 2.9 . L'efficacia complessiva del modello nell'effettuare previsioni corrette su entrambe le classi è proporzionale all'estensione dell'area sottesa alla curva ROC (Area Under Curve, AUC). L'AUC rappresenta una porzione del quadrato unitario e assume, dunque, valori compresi fra 0 e 1. Tuttavia, dato che la classificazione casuale genera come ROC la linea retta che collega i punti (0,0) e (1,1) che ha AUC pari a 0.5 (curva B in fig 2.9), nessun metodo di classificazione può avere un'area sottesa inferiore a 0.5 otterrebbe, altrimenti, prestazioni peggiori della classificazione casuale (Fawcett 2006 [11]).

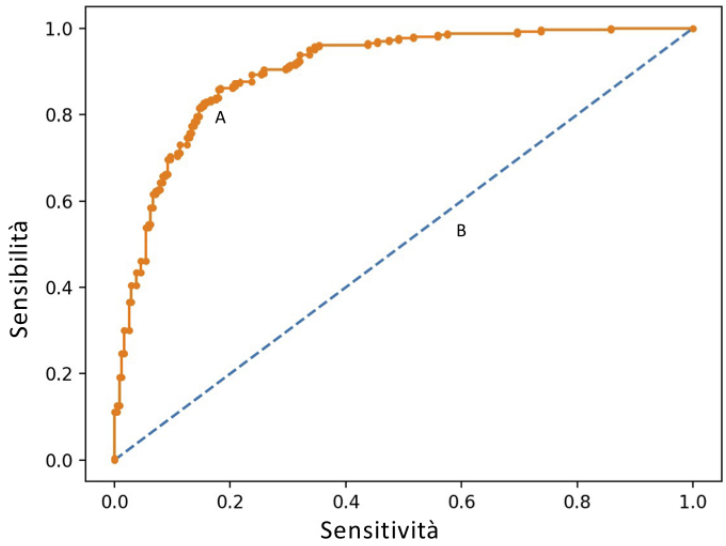


Figura 2.9: Esempio di ROC per un modello reale (A) e un modello inutile (B)

2.3.3 Precision Recall Curve

Come la curva ROC, la curva precisione-recall è uno strumento di valutazione per la classificazione binaria, che permette di visualizzare le prestazioni di un modello al variare della soglia. Questa curva risulta più efficace per dataset con classi non bilanciate, in cui una delle due classi appare più frequentemente. In questo caso vengono rappresentati i parametri di precisione sull'asse delle ordinate e recall (o sensibilità) su quello delle ascisse. Anche in questo caso la misura generale delle prestazioni del modello avviene tramite la misura dell'area sottesa alla curva (AUCPR).

Capitolo 3

Isolation Forest

Come anticipato in 2.2.6, il più recente approccio all'anomaly detection riguarda gli algoritmi basati sull'isolamento. Nelle altre tipologie di metodi, descritte in 2.2, la capacità di individuare le anomalie è un effetto collaterale di algoritmi originariamente costruiti per altri scopi, questo determina due problemi principali (Liu et al. 2008) [18]:

i) non essendo ottimizzati per l'AD questi metodi offrono prestazioni limitate e spesso determinano in un elevato numero di "falsi allarmi",

ii) questi approcci non sono, solitamente, applicabili a dataset multi-dimensionali o che presentano quantità ingenti di dati.

Gli algoritmi isolation-based nascono con lo scopo di individuare le anomalie ed eliminano la necessità di conoscere le misure di densità o di distanza in un dataset, l'isolamento delle istanze si può ottenere in diversi modi, in questo capitolo si analizzerà il metodo Isolation Forest, basato sulla struttura ad albero.

3.1 Metodi ad albero

I metodi ad albero si basano sulla struttura ad albero. Queste strutture sono costituite da due elementi, i nodi, che contengono l'informazione, e i rami, che stabiliscono collegamenti gerarchici fra i nodi. I nodi senza "figli" vengono chiamati *foglie*, un nodo non foglia è detto *nodo interno*. La struttura ad albero, come descritto da Barbariol et al (2022) [3], è largamente utilizzata per algoritmi di anomaly detection, si trovano diverse implementazioni in metodi basati sulla distanza e sulla densità, quelle più rilevanti, tuttavia, riguardano i metodi basati

sull'isolamento. Nel caso di algoritmi isolation-based, gli alberi vengono costruiti suddividendo ricorsivamente il dataset di partenza in sottoinsiemi inseriti nei nodi, la costruzione gerarchica permette, poi, di utilizzare diversi algoritmi (e.g. *PathLength* alg. 3) per assegnare degli *anomaly score*. Nel metodo Isolation Forest, trattato in questo capitolo, si utilizza in particolare la struttura di albero binario, in cui ogni nodo ha un numero di figli pari a 0 o 2.

3.2 Isolation forest

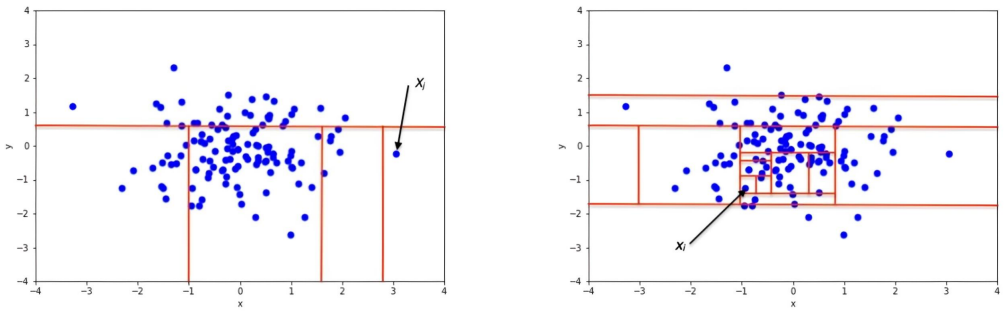
Liu et al. (2008) [18] introducono il metodo Isolation Forest (iForest), un metodo di anomaly detection, basato sull'isolamento delle istanze tramite strutture ad albero. L'algoritmo è stato descritto assumendo un approccio non supervisionato e non parametrico, utilizzando dati a valori continui. Questo metodo viene proposto come alternativa a quelli già in uso per i seguenti vantaggi che lo caratterizzano:

i) complessità computazionale limitata, non essendo necessaria alcuna misura di distanza o densità,

ii) complessità temporale lineare rispetto a una costante piccola e requisiti di memoria minimi,

iii) capacità di analizzare problemi con alta dimensionalità e dataset con un numero elevato di dati.

L'idea alla base dell'algoritmo è la seguente, essendo le anomalie "poche e differenti" in un dataset esse saranno più facilmente isolabili rispetto ai dati normali. A livello pratico ciò significa che dividendo in maniera causale un dataset ricorsivamente, fino ad aver separato ogni istanza, si può assegnare un'anomaly score a ogni dato in base al numero di divisioni che sono state necessarie ad isolarlo. Le istanze con associato il numero più basso di divisioni saranno più probabilmente anomale. In fig 3.1 si possono osservare due esempi di isolamento di due diverse istanze dello stesso dataset, nel primo caso sono necessarie 4 suddivisioni per isolare il punto x_j, nel secondo vengono fatte 13 divisioni per isolare x_i, ciò indica che x_j è più probabilmente un'anomalia, rispetto a x_i.



(a) isolamento di x_j **(b)** isolamento di x_i

Figura 3.1: Esempi di isolamento di istanze per un dataset con distribuzione normale in 2D

3.2.1 Isolation Trees (iTree)

La partizione ricorsiva del dataset può essere rappresentata utilizzando un albero binario definito come *isolation tree* (iTree), diventa così immediato calcolare il numero di divisioni necessario ad isolare un'istanza, misurando la lunghezza del tragitto nell'albero che separa la radice dalla foglia in cui l'istanza è contenuta. L'iTree viene definito da Liu et al. (2008) [18] come una struttura ad albero che possiede la seguente proprietà: *per ogni nodo T nell'albero, T è o un nodo esterno senza alcun figlio (foglia), o un nodo interno con un "test" associato ed esattamente due figli.* Dove il "test" è costituito da un attributo q, che determina rispetto a quale caratteristica del dataset viene fatta la partizione, e da un valore di separazione p, ossia il valore di soglia con cui si decide se una particolare istanza va inserita nel nodo figlio di sinistra (T_l) o di destra (T_l).

Per costruire l'albero, dunque, l'algoritmo separerà, in modo ricorsivo, il sottoinsieme del dataset di partenza, presente in ogni nodo, scegliendo in maniera casuale un attributo q e un valore p, finchè non si raggiunge una delle seguenti situazioni:

1. l'albero raggiunge una *altezza* prestabilita,
2. il nodo contiene un sottoinsieme costituito da un solo elemento,
3. tutti i dati nel sottoinsieme hanno gli stessi valori.

Al termine della costruzione, tutte le istanze del dataset di partenza saranno isolate in una delle foglie dell'albero. Il numero di nodi esterni sarà pari al numero n di istanze nel dataset di partenza, un iTree completo sarà, quindi, composto da $2n - 1$ nodi.

Lo pseudocodice per la costruzione di un albero di isolamento è fornito nell'algoritmo 2.

In fig 3.2 si può osservare un esempio di iTree, in questo caso il tragitto in rosso è evidentemente più corto di quello blu, l'istanza associata al primo, dunque, sarà più probabilmente un'anomalia, rispetto a quella associata al secondo.

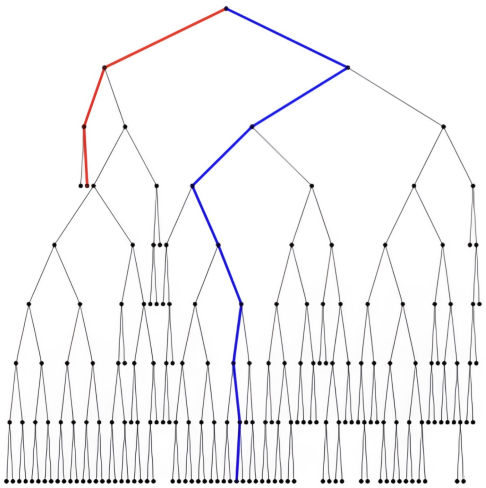


Figura 3.2: Rappresentazione di un albero

3.2.2 Anomaly score

L'anomaly score di un'istanza viene assegnata in base alla profondità del nodo che la contiene nell'albero. Più precisamente si definisce l'anomaly score s dell'istanza x nel seguente modo:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{3.1}$$

Dove $h(x)$ è la *lunghezza del tragitto*, ossia il numero di rami che compongono il tragitto che lega la radice al nodo in cui si trova x, $E(h(x))$ è la media fra le lunghezze di tragitto di x di diversi alberi, e $c(n)$ è un fattore di normalizzazione dato dal valore medio di $h(x)$ in un albero. Si osserva che:

- se $E(h(x)) \to c(n)$, allora $s \to 0.5$
- se $E(h(x)) \to 0$, allora $s \to 1$
- se $E(h(x)) \to n - 1$, allora $s \to 0$.

Un'istanza con s vicino a 1 ha, dunque, una *lunghezza di tragitto* prossima allo zero, ciò significa che è stata facilmente isolata e quindi è quasi certamente un'anomalia. Al contrario un'istanza con s molto minore di 0.5 si trova in un nodo molto distante dalla radice e di conseguenza è altamente improbabile che sia

un'anomlia. Nel caso in cui tutte le istanze riportino un valore $s \approx 0.5$, l'intero dataset non presenta anomalie distinguibili, dato che ogni istanza è stata isolata mediamente con lo stesso numero di divisioni.

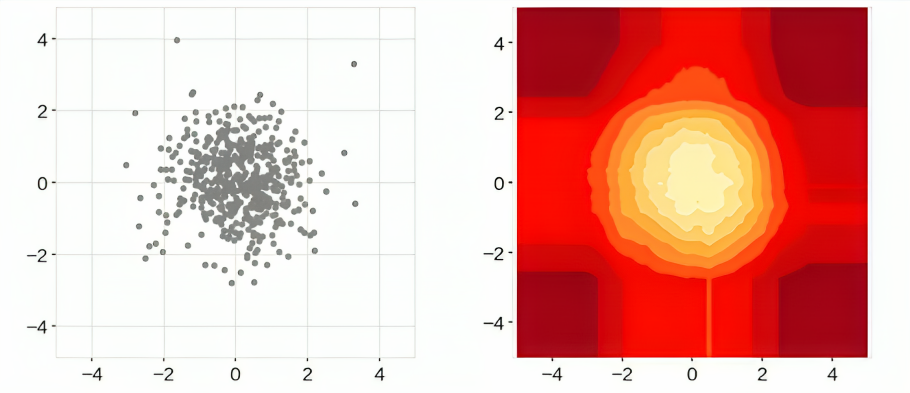


Figura 3.3: Dati con distribuzione normale in 2D e mappa degli anomaly score associati. Le zone più scure rappresentano un AS più alto

In fig 3.3 è rappresentato un dataset con distribuzione normale in due dimensioni, con media nulla e la matrice di identità come covarianza, e un grafico degli anomaly score associati ad ogni zona, in cui ad un AS più alto è associato un colore più scuro. Ci si aspetterebbe di osservare l'AS minimo nel punto (0,0) e una crescita radiale e simmetrica di questo, all'aumentare della distanza dal centro. Nella realtà, tuttavia, il metodo Isolation Forest ottiene i risultati previsti vicini al centro, mentre si allontana da questi nelle zone più distanti. Questo comportamento determina l'insorgere di errori nell'identificazione di anomalie, una possibile soluzione al problema viene introdotta con il metodo *Extended Isolation Forest*, Hariri et al. (2019) [12].

3.3 Anomaly Detection mediante Isolation Forest

Per ottenere un metodo di anomaly detection a partire dagli alberi di isolamento è necessario distinguere due fasi, una prima fase di addestramento, in cui si costruiscono gli alberi, e una seconda di test in cui si ottengono i valori di AS per ogni istanza.

3.3.1 Fase 1: Addestramento iForest

Nella fase di addestramento vengono costruiti gli alberi di isolamento descritti in 3.2.1. Dato che per il rilevamento di anomalie sono utili solo i nodi presenti

nei livelli più alti degli alberi, non è necessario costruire gli alberi completi per il dataset, solitamente è sufficiente raggiungere la profondita media $l = \lceil log_2(n) \rceil$, che viene impostata come altezza limite. In questa fase, con le modalità descritte dal pseudocodice 1, si costruisce la *foresta di isolamento* ossia una lista contenente t alberi di isolamento, costruiti con il metodo iTree (pseudocodice 2).

Algorithm 1 $iForest(X, t, \psi)$

Input: X- dataset, t- numero di alberi, ψ- dimensione sample
Output: una lista di t alberi di isolamento

```
forest ← lista vuota di dimensione n;
h_max ← ⌈log2|X|⌉;
for i = 1 to t do
    X' ← sample(X, ψ);
    forest[i] ← IsolationTree(X', 0, h_max);
end for
return forest
```

Algorithm 2 $IsolationTree(X, h, h_{max})$

Input: X- dataset, h- profondità attuale dell'albero, h_{max}- limite di profondità
Output: Albero di isolamento

```
if h ≥ h_max or |X| ≥ 1 then
    return Leaf{ size ← |X| };
else
    q ← attributo di X scelto casualmente;
    p ← soglia scelta casualmente tra [minX^(q), maxX^(q)];
    X_L ← filter(X, X^(q) ≥ p);
    X_R ← filter(X, X^(q) < p);
    return Node{
        left ← IsolationTree(X_L, h + 1, h_max)
        right ← IsolationTree(X_R, h + 1, h_max)
        splitAtt ← q
        splitValue ← p };
end if
```

Liu et al. (2008) forniscono come valori ottimali dei parametri $\psi = 256$ e $t = 100$ nell'algoritmo 1, indicano, inoltre, che la complessità della fase di addestramento del metodo iForest è $O(t\psi log(\psi))$

3.3.2 Fase 2: Test

La fase di test consiste nel calcolo e l'assegnazione dell'anomaly score descritto nella sezione 3.2.2. Con l'algoritmo 3 si deriva una singola lunghezza di percorso

$h(x)$, contando i rami tra la radice e la foglia contenente x. Quando il conteggio viene terminato con x non isolato, ma contenuto in un nodo esterno (nodo con $Size > 1$), al valore di output viene sommato un valore di regolazione $c(Size)$, che considera il sottoalbero che supera l'altezza limite. Una volta calcolati i valori di $h(x)$ per ogni albero, si ottiene l'anomaly score di x calcolando $s(x, \psi)$ nell'Equazione 3.1 Liu et al. (2008) [18] indicano che la complessità della fase di test del metodo iForest è $O(ntlog(\psi))$.

Algorithm 3 $PathLength(x, T, e)$

Input: x- istanza, T- albero di isolamento, e- lunghezza attuale del tragitto (inizializzare a 0 quando inizialmente chiamato sulla radice)
Output: Lunghezza del tragitto di x

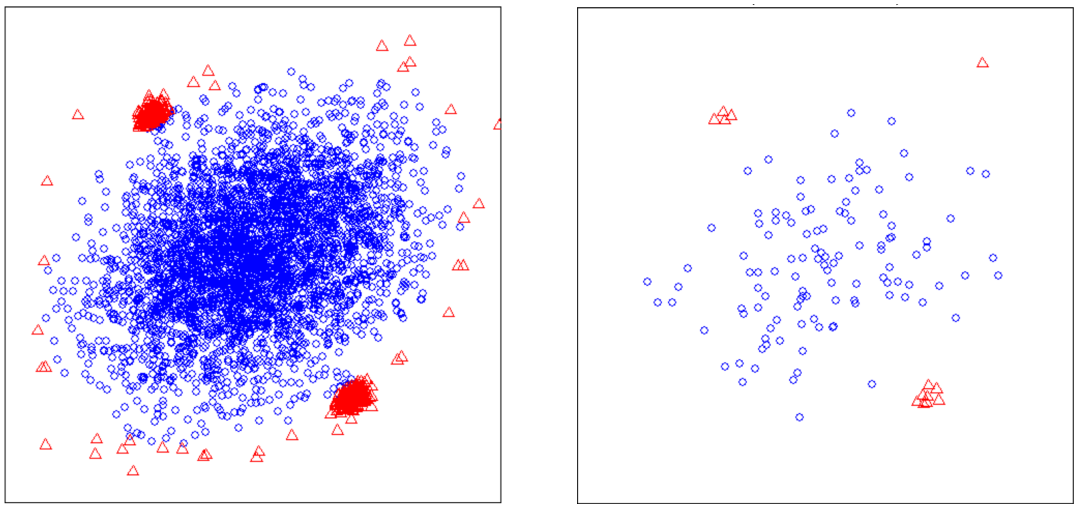
```
if T è un nodo esterno then
    return e + c(T.size);
end if
a ← T.splitAtt
if x_a < T.splitValue then
    return PathLength(x, T.left, e + 1)
else{ x_a ≥ T.splitValue}
    return PathLength(x, T.right, e + 1)
end if
```

3.4 Proprietà di iForest

Per l'identificazione delle anomalie con il metodo iForest non è necessario che tutte le istanze normali vengano isolate, come descritto nella sezione 3.3.1, questo permette di utilizzare dei sottoinsiemi del dataset originale. Questa pratica è chiamata *sub-sampling* e permette di ottenere prestazioni ottime, se applicata a metodi di isolamento, eliminando i problemi determinati da *swamping*, *masking* e da dataset ad elevata dimensionalità.

Swamping e masking Il fenomeno di *swamping* si verifica quando eventi normali vengono erroneamente etichettati come anomalie, questo avviene, nel caso di iForest, quando le anomalie sono troppo simili ai dati normali e necessitano di molte divisioni per essere isolate. L'effetto di *masking* si osserva, invece, quando un'anomalia non viene identificata a causa della presenza di molte altre adiacenti ad essa che la "mascherano". Ambedue gli effetti sono determinanti dalla presenza di troppi dati ai fini dell'anomaly detection, per questo motivo l'utilizzo

di sottoinsiemi del dataset di partenza (*sub-sampling*) rappresenta un vantaggio. Un esempio di ciò e riportato in fig 3.4, nella prima immagine si osserva che il dataset completo, composto da 4096 elementi, presenta punti anomali (triangoli rossi) non distanziati dai dati normali e due *cluster* di dati anomali con densità maggiore a quella dei dati normali. In questa condizione l'identificazione delle anomalie risulta poco efficace, Liu et al. riportano AUC pari a 0.67 quando iForest viene applicato al dataset. Nella seconda immagine, ottenuta considerando un sottoinsieme di 128 istanze del dataset, appaiono, invece, evidenti i *cluster* anomali e il valore di AUC riferito da Liu et al. sale a 0.91.



(a) Dataset originale (4096 dati) **(b)** Sottoinsieme (128 dati)

Figura 3.4: Esempi di isolamento di sub-sampling Liu et al.

Dataset ad alta dimensionalità Come già visto per i metodi basati sulla distanza (sez. 2.2.2) le tecniche di anomaly detection riportano una riduzione sostanziale delle prestazioni al crescere della dimensionalità dei dataset. Anche il metodo iForest risente della cosiddetta "maledizione della dimensionalità", tuttavia la possibilità di selezionare gli attributi secondo cui vengono fatte le divisioni dei sample nella costruzione degli *iTree*, determina un miglioramento dell'accuratezza nell'identificazione di anomalie e una riduzione dei tempi di elaborazione.

Le proprietà sopra descritte determinano prestazioni ottime del metodo Isolation Forest nel riconoscimento delle anomalie in dataset reali, un esempio di ciò e riportato nel capitolo 4.

Capitolo 4

Confronto tra i metodi di Anomaly detection

In questo capitolo viene proposto un confronto fra i diversi metodi di anomaly detection descritti in precedenza utilizzando i dati forniti da Falcão et al. (2019) [10].

4.1 Caratteristiche del confronto

Il confronto operato da Falcão et al. [10] si basa sull'analisi delle metriche di precisione, recall (o sensibilità), accuratezza, punteggio F1 e AUC del grafico ROC, tutte descitte nel paragrafo 2.3, ricavate dall'applicazione di 12 diversi algoritmi di anomaly detection non supervisionata su 5 dataset differenti. Gli algoritmi sono stati suddivisi in 6 categorie, in base alla tipologia di metodi a cui appartengono, in particolare le categorie sono:

1. classification-based, in cui si utilizza il metodo iForest,
2. density-based (sez. 2.2.3),
3. statistical-based (sez. 2.2.1),
4. distance-based (sez. 2.2.2),
5. clustering-based (sez. 2.2.5),
6. angle-based (sez. 2.2.4).

I dataset utilizzati sono KDD-CUP 99 [24], NSL-KDD [28], ADFA-LD [9], ISCX2012 [25], and UNSW-NB15 [21].

4.2 Analisi dei risultati

In fig 4.1 sono raffigurati i risultati ottenuti, raggruppati secondo le categorie sopra elencate, nel grafico le colonne rappresentano la mediana dei valori ottenuti, mentre le barre di errore rappresentano la deviazione standard. Le diverse categorie di algoritmi sono ordinate in base al valore F1 in maniera decrescente, si osserva facilmente che rispetto a questa metrica i metodi basati sulla classificazione risultano i migliori, quelli statistical-based e density-based ottengono valori simili, mentre quelli basati sulla distanza hanno un valore basso della mediana, ma elevata deviazione standard, i metodi basati sul clustering e sulla misura degli angoli, infine, risultano essere i meno performanti rispetto a questa metrica. Questo ordinamento dei metodi rimane praticamente invariato considerando le metriche di precisione e recall, se si considera l'accuratezza, invece, si osserva che i metodi angle-based superano quelli clustering e distance-based. Questa differenza può indicare che gli algoritmi angle-based hanno una percentuale più elevata di dati *veri negativi*, che vengono considerati nella misura di accuratezza, ma non in quella dell'F1-score.

Ogni algoritmo utilizza ed è ottimizzato per parametri diversi, per questo motivo sono state adoperate diverse combinazioni di parametri per costruire la curva ROC e, di conseguenza, calcolare l'AUC. Si può dunque osservare come gli algoritmi basati sulla classificazione, fra cui iForest, siano fortemente dipendenti dalla scelta dei parametri e non siano, di conseguenza, sempre da considerare come la scelta ottimale. Essi ottengono, infatti, i risultati peggiori rispetto alla metrica AUC. In maniera contraria le tecniche angle-based ottengono risultati sensibilmente migliori nell'AUC rispetto a tutti gli altri metodi, ciò significa che gli algoritmi appartenenti a questa categoria hanno una bassa dipendenza dalla scelta dei parametri.

Osservando la tabella 4.2 si conclude che il metodo iForest risulta il migliore rispetto agli altri, esso, infatti, ottiene risultati ottimi per l'AD nella valutazione di precisione, recall e accuratezza, che hanno tutti valori prossimi al 99%.

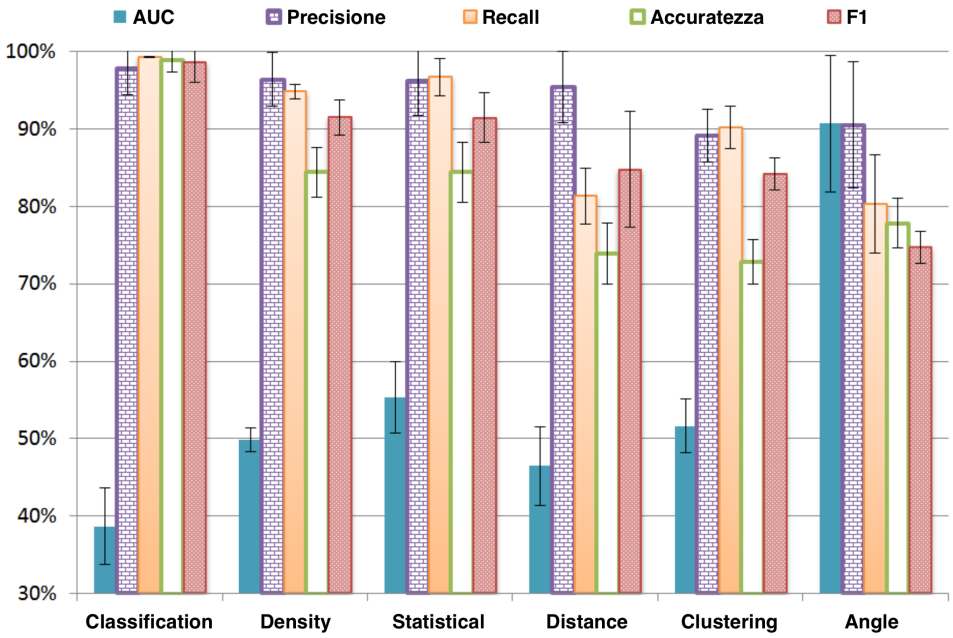


Figura 4.1: Confronto tra metodi di AD

Algorithm	Family	AUC	Precision	Recall	Accuracy	F1
Isolation Forest	Classification	37.2 ± 0.4	99.9 ± 0.3	99.3 ± 0.4	99.7 ± 0.3	99.6 ± 0.3
One-Class SVM	Classification	53.4 ± 2.9	96.6 ± 3.2	99.3 ± 0.0	96.2 ± 3.2	98.0 ± 1.9
COF	Density-Based	48.8 ± 1.7	93.6 ± 3.4	97.8 ± 0.1	91.7 ± 3.1	95.7 ± 2.0
ODIN	Distance-Based	49.9 ± 1.7	96.6 ± 2.4	99.9 ± 0.4	89.8 ± 1.6	94.6 ± 1.1
HBOS	Statistical	57.8 ± 5.5	92.6 ± 5.8	99.5 ± 4.3	89.2 ± 4.7	94.3 ± 4.8
rPCA	Statistical	55.0 ± 4.0	97.5 ± 3.4	95.0 ± 1.0	83.1 ± 3.2	90.6 ± 2.0
LOF	Density-Based	50.0 ± 1.3	96.6 ± 3.5	88.0 ± 1.1	81.3 ± 3.1	89.6 ± 2.1
LDCOF	Clustering	49.9 ± 2.3	82.4 ± 1.8	94.4 ± 0.2	77.9 ± 1.5	87.4 ± 0.7
KNN	Distance-Based	35.9 ± 6.7	91.9 ± 5.8	75.1 ± 3.4	71.4 ± 4.0	82.8 ± 4.3
K-Means	Clustering	54.4 ± 8.9	95.7 ± 5.3	68.5 ± 2.8	65.6 ± 3.4	78.3 ± 3.5
ABOD	Angle-Based	90.5 ± 7.8	69.2 ± 8.1	92.4 ± 8.3	90.0 ± 1.8	75.5 ± 10.2
FastABOD	Angle-Based	86.4 ± 9.2	90.6 ± 7.8	77.4 ± 5.3	67.6 ± 3.2	74.7 ± 6.1

Figura 4.2: Confronto tra metodi di AD

Bibliografia

[1] Aggarwal, C. C., Hinneburg, A., and Keim, D. A. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory—ICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings 8* (2001), Springer, pp. 420–434.

[2] Anscombe, F. J. Rejection of outliers. *Technometrics 2*, 2 (1960), 123–146.

[3] Barbariol, T., Chiara, F. D., Marcato, D., and Susto, G. A. A review of tree-based approaches for anomaly detection. *Control Charts and Machine Learning for Anomaly Detection in Manufacturing* (2022), 149–185.

[4] Bottarelli, E., and Parodi, S. Un approccio per la valutazione della validità dei test diagnostici: le curve roc (receiver operating characteristic). *Ann. Fac. Medic. Vet. di Parma 23* (2003), 49–68.

[5] Boyd, K., Eng, K. H., and Page, C. D. Area under the precision-recall curve: point estimates and confidence intervals. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13* (2013), Springer, pp. 451–466.

[6] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (2000), pp. 93–104.

[7] Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM computing surveys (CSUR) 41*, 3 (2009), 1–58.

[8] Chiang, J.-T., et al. The masking and swamping effects using the planted mean-shift outliers models. *Int. J. Contemp. Math. Sciences 2*, 7 (2007), 297–307.

[9] Creech, G., and Hu, J. Generation of a new ids test dataset: Time to retire the kdd collection. In *2013 IEEE wireless communications and networking conference (WCNC)* (2013), IEEE, pp. 4487–4492.

[10] Falcão, F., Zoppi, T., Silva, C. B. V., Santos, A., Fonseca, B., Ceccarelli, A., and Bondavalli, A. Quantitative comparison of unsupervised anomaly detection algorithms for intrusion detection. In *Proceedings of the 34th ACM/SIGAPP symposium on applied computing* (2019), pp. 318–327.

[11] Fawcett, T. An introduction to roc analysis. *Pattern recognition letters 27*, 8 (2006), 861–874.

[12] Hariri, S., Kind, M. C., and Brunner, R. J. Extended isolation forest. *IEEE transactions on knowledge and data engineering 33*, 4 (2019), 1479–1489.

[13] Hawkins, D. M. *Identification of outliers*, vol. 11. Springer, 1980.

[14] Hawkins, S., He, H., Williams, G., and Baxter, R. Outlier detection using replicator neural networks. In *International Conference on Data Warehousing and Knowledge Discovery* (2002), Springer, pp. 170–180.

[15] Kang, M. *Machine Learning: Anomaly Detection.* John Wiley & Sons, Ltd, 2018, ch. 6, pp. 131–162.

[16] Knorr, E. M., Ng, R. T., and Tucakov, V. Distance-based outliers: algorithms and applications. *The VLDB Journal 8*, 3 (2000), 237–253.

[17] Kriegel, H.-P., Schubert, M., and Zimek, A. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (2008), pp. 444–452.

[18] Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation forest. In *2008 eighth ieee international conference on data mining* (2008), IEEE, pp. 413–422.

[19] Liu, F. T., Ting, K. M., and Zhou, Z.-H. On detecting clustered anomalies using sciforest. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part II 21* (2010), Springer, pp. 274–290.

[20] Mandhare, H. C., and Idate, S. A comparative study of cluster based outlier detection, distance based outlier detection and density based outlier detection techniques. In *2017 international conference on intelligent computing and control systems (ICICCS)* (2017), IEEE, pp. 931–935.

[21] Moustafa, N., and Slay, J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)* (2015), IEEE, pp. 1–6.

[22] Muruti, G., Rahim, F. A., and bin Ibrahim, Z.-A. A survey on anomalies detection techniques and measurement methods. In *2018 IEEE conference on application, information and network security (AINS)* (2018), IEEE, pp. 81–86.

[23] Papadimitriou, S., Kitagawa, H., Gibbons, P. B., and Faloutsos, C. Loci: Fast outlier detection using the local correlation integral. In *Proceedings 19th international conference on data engineering (Cat. No. 03CH37405)* (2003), IEEE, pp. 315–326.

[24] Rosset, S., and Inger, A. Kdd-cup 99: knowledge discovery in a charitable organization's donor database. *ACM SIGKDD Explorations Newsletter 1*, 2 (2000), 85–90.

[25] Shiravi, A., Shiravi, H., Tavallaee, M., and Ghorbani, A. A. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *computers & security 31*, 3 (2012), 357–374.

[26] Smiti, A. A critical overview of outlier detection methods. *Computer Science Review 38* (2020), 100306.

[27] Sokolova, M., Japkowicz, N., and Szpakowicz, S. Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. In *Australasian joint conference on artificial intelligence* (2006), Springer, pp. 1015–1021.

[28] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (2009), Ieee, pp. 1–6.

[29] Woolley, T. W. An investigation of the effect of the swamping phenomenon on several block procedures for multiple outliers in univariate samples. *Open Journal of Statistics 3*, 5 (2013), 299–304.

[30] Zhao, Y., Nasrullah, Z., and Li, Z. Pyod: A python toolbox for scalable outlier detection. *arXiv preprint arXiv:1901.01588* (2019).