Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Tesi di Laurea

TITOLO ITALIANO

TITOLO INGLESE

ALESSANDRO PISCOPO

Relatore: *Tommaso Zoppi*
Correlatore: *Correlatore*

Anno Accademico 2022-2023

INDICE

ELENCO DELLE FIGURE

"Inserire citazione"
— Inserire autore citazione

1

INTRODUZIONE

Nell'era in cui viviamo l'analisi dei dati riveste un ruolo cruciale in molti settori della nostra società. In questo contesto il Machine Learning si è dimostrato un potente strumento per estrarre informazioni utili e conoscenza da dati complessi e voluminosi. Il Machine Learning (ML) è un sottoinsieme dell'intelligenza artificiale (AI) che si occupa di creare sistemi che apprendono e migliorano le proprie performance in base ai dati che utilizzano.

In particolare lo scopo del lavoro di Tesi è stato analizzare due approcci diversi all'apprendimento automatico: un approccio classico e un approccio time series. Una time series può essere definita come un insieme di osservazioni ordinate rispetto al tempo. La differenza sostanziale tra i due approcci è che con un approccio time series si hanno informazioni non solo sull'istante di tempo corrente, ma anche su un numero di istanti di tempo precedenti scelto arbitrariamente. Il fine ultimo del presente lavoro di Tesi è stato quello di comparare le performance di 4 modelli in condizioni diverse: Logistic Regression, Linear Discriminant Analysis, Random Forest, XGBoost. Le condizioni diverse sopracitate sono state date dall'addestramento dei modelli su set di dati differenti in base all'approccio utilizzato, classico o time series. L'ipotesi che abbiamo voluto verificare è quella che un approccio time series migliori le performance dei modelli rispetto ad un approccio classico, avendo a dispozione informazioni su una finestra di istanti di tempo precedenti e quindi più dati disponibili durante l'apprendimento.

Il lavoro è organizzato nel seguente modo:

- Capitolo 2: fornisce una panoramica su Machine Learning, Anomaly Detection e Time Series

- Capitolo 3: descrive le metodologie e le strategie utilizzate durante gli esperimenti
- Capitolo 4: analisi dei risultati ottenuti
- Capitolo 5: conclusioni della Tesi

2

STATO DELL'ARTE

2.1 MACHINE LEARNING

Il Machine Learning (ML) è una branca dell'intelligenza artificiale che si è sviluppata negli'ultimi decenni del XX secolo. Nel campo dell'informatica, l'apprendimento automatico è una variante alla programmazione tradizionale nella quale in una macchina si predispone l'abilità di apprendere dai dati in maniera autonoma, senza istruzioni esplicite. I metodi principali per l'apprendimento automatico sono due:

- **Apprendimento Supervisionato**: vengono utilizzati dati etichettati per effettuare il training del modello di ML. Viene quindi fornita la corrispondenza tra input e output durante la fase di training.
- **Apprendimento Non Supervisionato**: vengono utilizzati dati non etichettati durante l'addestramento. Non viene resa esplicita quindi nessuna relazione tra input e output, sarà l'algoritmo ad estrarre le informazioni necessarie a classificare o predire i risultati attesi.

Nel nostro caso, abbiamo utilizzato l'Apprendimento Supervisionato avendo a disposizione dei set di dati etichettati. Esistono principalmente due tipi di Apprendimento Supervisionato:

- **Classificazione**: un algoritmo (classificatore) è addestrato a classificare i dati di input su variabili discrete. Durante l'addestramento, gli algoritmi ricevono dati di input con un'etichetta "classe" e dovranno essere in grado, una volta addestrati, di restituire la classe di appartenenza di nuovi input forniti al modello.
- **Regressione**: un algoritmo (regressore) deve individuare una relazione funzionale tra i parametri di input e l'output. Il valore di output non è discreto come nella classificazione, ma è una funzione dei parametri di input.

2.1.1 *Algoritmi utilizzati*

Gli algortitmi utilizzati nel presente lavoro di tesi sono tutti classificatori, avendo preso come caso di studio un problema di classificazione, in particolare un problema di classificazione binaria. Sono stati utilizzati in totale 4 algoritmi di classificazione, 2 per ciascuna classe:

- **Modelli Statistici**: Logistic Regression, Linear Discriminant Analysis
- **Modelli basati su Alberi Decisionali**: Random Forest, XGBoost

In generale, i modelli statistici sono modelli più semplici rispetto ai modelli basati su alberi decisionali, sacrificando delle migliori performance in favore di una maggiore interpretabilità.

Modelli Statistici

La **Logistic Regression (regressione logistica)** è uno dei modelli statistici più utilizzati nell'ambito del ML. Per regressione logistica si intende l'analisi di regressione che si conduce quando la variabile dipendente è dicotomica, cioè binaria, ad esempio anomalia rilevata o non rilevata. Il modello di regressione logistica può essere utilizzato per spiegare la relazione tra una variabile binaria dipendente Y e una o più variabili indipendenti X_i. La Logistic Regression si compone delle seguente variabili:

- **Y, variabile dipendente dicotomica**: assume valore 0 quando l'evento non si verifica (anomalia non rilevata) e valore 1 quando l'evento si verifica (anomalia rilevata).
- X_i**, variabili indipendenti o regressori**: queste possono essere di qualsiasi natura, qualitative o quantitative e influenzano la variabile risposta Y.

Il modello da stimare è dato dall'espressione:

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n)}} + \epsilon \tag{2.1}$$

Dove i coefficienti $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ sono i coefficienti di regressione

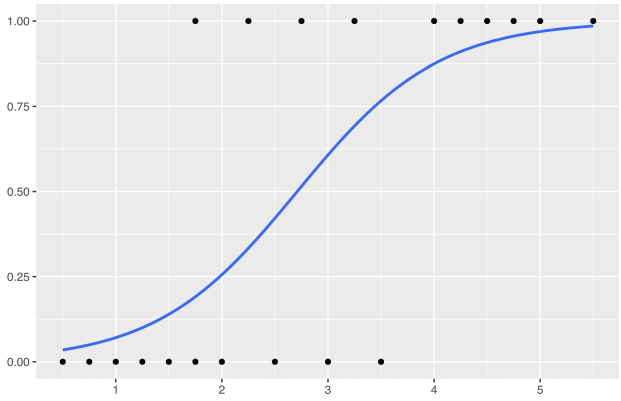

Figura 1: Logistic Regression

L'altro modello statistico utilizzato è stato la **Linear Discriminant Analysis (LDA)**. Questo modello, come il modello precedente, cerca di trovare una combinazione lineare di features che separino al meglio le classi nel dataset. La LDA funziona proiettando i dati su uno spazio a minore dimensionalità che massimizza la separazione tra le classi. Ciò avviene trovando un insieme di discriminanti lineari che massimizzano il rapporto tra la varianza inter-classe e la varianza intra-classe. In altre parole, trova le direzioni nello spazio delle features che meglio separano le diverse classi di dati, nel nostro caso anomalia rilveata o non rilevata. Il logaritmo della probabilità per LDA può essere scritto, considerando tutte le classi k e tutti i campioni di training x, come :

$$\log P(y = k|x) = \omega_k^t x + \omega_{k0} + Cst. \tag{2.2}$$

con $\omega_k = \Sigma^{-1}\mu_k$ e $\omega_{k0} = -\frac{1}{2}\mu_k^t\Sigma^{-1}\mu_k + \log P(y = k)$. Queste quantità corrispondono rispettivamente ai coefficienti e all'intercetta.

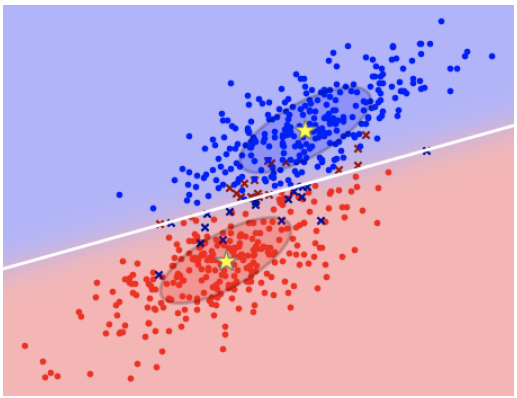

Figura 2: Linear Discriminant Analysis

Modelli basati su Alberi Decisionali

I modelli non statistici utilizzati sono entrambi basati sugli alberi decisionali, quindi prima di definire i singoli modelli, procederemo con un'introduzione su cosa sono gli alberi decisionali. Un **Albero Decisionale (DT)** è un algoritmo di apprendimento supervisionato, impiegato sia per attività di classificazione che di regressione. Si compone di una struttura ad albero gerarchica, ed è formato da un nodo radice, rami, nodi interni e nodi foglia.

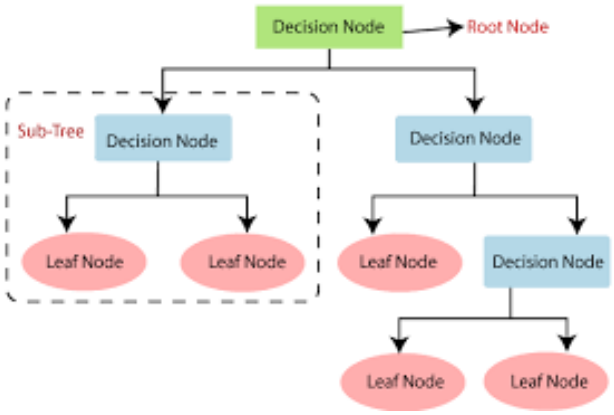

Figura 3: Schema Albero Decisionale

Come si può vedere dal diagramma soprastante, un albero decisionale inizia con un nodo radice, che non ha rami in entrata. I rami in uscita dal nodo radice alimentano i nodi interni, anche detti nodi decisionali. Entrambi i tipi di nodi contribuiscono a formare sottoinsiemi omogenei, che sono rappresentati dai nodi foglia. I nodi foglia corrispondono a tutti i risultati possibili nel set di dati. Il training dell'albero decisionale utilizza una strategia "dividi et impera" per cercare i punti di suddivisione ottimali all'interno di un albero. Vediamo nella prossima pagina un diagramma, molto semplice e a titolo di esempio, di un albero decisionale per determinare la presenza o meno di un'anomalia all'interno di un calcolatore.

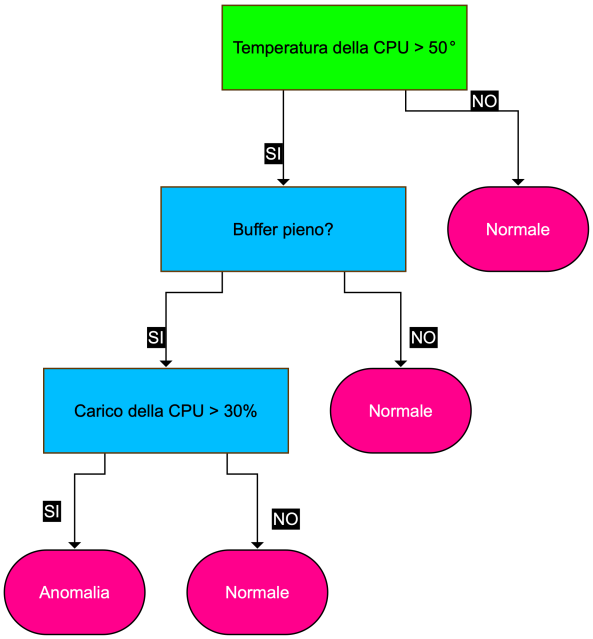

Figura 4: Esempio Albero Decisionale

Nel nostro caso non abbiamo usato direttamente degli alberi decisionali, ma dei modelli basati su di essi. L'algoritmo **Random Forest (RF)** è un classificatore d'insieme basato su alberi decisionali, ovvero è costituito da un insieme di classificatori, in questo caso DT, e le loro previsioni vengono aggregate per identificare il risultato più diffuso. In particolare, l'algoritmo RF utilizza una tecnica dell'apprendimento d'insieme, chiamata bagging, in cui più DT vengono addestrati su insiemi di dati diversi, ciascuno ottenuto dal dataset di addestramento iniziale. La casualità delle caratteristiche su cui vengono addestrati i singoli alberi della foresta garantisce una bassa correlazione tra le singole strutture ad albero, riducendo così uno dei maggiori problemi degli Alberi Decisionali, l'overfitting. L'overfitting è un problema che si verifica quando un modello si adatta esattamente ai suoi dati di addestramento. Quando questo accade, l'algoritmo non funziona correttamente in presenza di dati non osservati in precedenza e non è quindi in grado di generalizzare, risultando inutile. Possiamo schematizzare il comportamento dell'algoritmo Random Forest nella figura seguente.

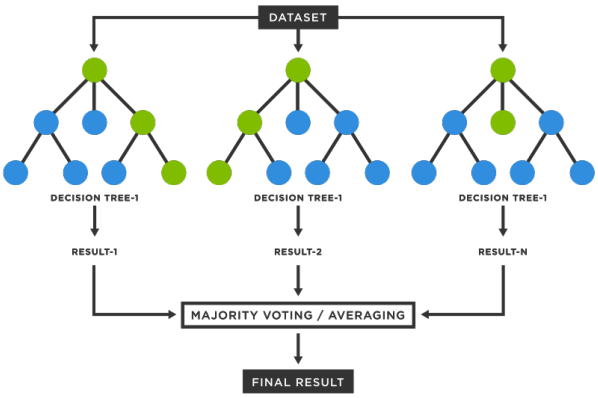

Figura 5: Schema Random Forest

Nel nostro caso, ovvero nel caso della classificazione, il risultato del modello corrisponderà al voto di maggioranza dei singoli alberi sulla classe prevista.

L'altro algoritmo basato su alberi decisionali utlizzato nel presente lavoro di Tesi è **XGBoost**, il cui nome completo è Extreme Gradient Boosting. L'idea di base dell'algoritmo, che è alla base di tutti gli algoritmi di ensemble learning, è che un insieme di modelli deboli che lavorano insieme per ottenere una previsione abbia risultati migliori rispetto a un singolo modello forte. Come si deduce dal nome dell'algoritmo, XGBoost è basato sul boosting, ovvero un metodo di apprendimento d'insieme, proprio come lo Ã¨ il bagging per Random Forest. Il boosting combina una serie di learner deboli che vanno a formare un learner forte per ridurre al minimo gli errori di addestramento. Nel boosting, viene effettuata una selezione casuale dal set di dati e il modello viene addestrato in modo sequenziale, ovvero ogni modello successivo nella squenza cerca di migliorare rispetto al predecessore. I learner deboli , che non sono nient'altro che alberi decisionali, vengono chiamati anche "regole deboli" e vengono combinate per formare una regola di previsione forte. La differenza principale con la tecnica di bagging utilizzata da Random Forest è che nel bagging i learner deboli sono addestrati in parallelo. mentre nel boosting apprendono in sequenza. Questo significa

che ad ogni iterazione del modello, vengono aumentati i pesi dei dati erroneamente classificati nel modello precedente. Questa ridistribuzione dei pesi permette all'algoritmo di identificare i parametri su cui su cui deve porre attenzione per migliorare le sue performance. XGBoost (Extreme Gradient Boosting), come si deduce nuovamente dal nome, è un implementazione di Gradient Boosting, progettata per velocità e scalabilità. Il Gradient Boosting funziona aggiungendo in sequenza i learner a un insieme, in modo che ciascuno corregga gli errori del suo predecessore e viene addestrato sugli errori residui del precedente predittore. Il nome, Gradient Boosting, viene utilizzato poiché combina l'algoritmo di discesa del gradiente, un algoritmo di ottimizzazione molto utilizzato nel ML, e il metodo di boosting.

2.1.2 *Metriche di valutazione*

In questa sezione illustriamo le metriche di valutazione utilizzate per valutare le performance dei modelli. Prima di menzionare le singole metriche, è necessario introdurre il concetto di *confusion matrix (matrice di confusione)*. La matrice di confusione è una matrice utilizzata per valutare le prestazioni di un algoritmo di ML, che può essere così schematizzata:

		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

Figura 6: Confusion Matrix

Ora che abbiamo introdotto la confusion matrix possiamo meglio definire le metriche di valutazione. In totale sono state utilizzate 4 metriche di valutazione:

- **Accuracy (ACC)**: il valore è dato dal numero delle classificazioni corrette diviso il numero totale di classificazioni. Formalmente, guardando alla confusion matrix, il valore è dato da:

$$ACC = \frac{TP + TN}{P + N} \tag{2.3}$$

- **Error Rate (1-ACC)**: il valore è dato dalla differenza tra 1 e l'Accuracy, ovvero $1 - ACC$
- **Matthews correlation coefficient (MCC)**: questa metrica di valutazione, rispetto alle precedenti, è sicuramente la più robusta per valutare le performance dei modelli. Questo perchè, a differenza dell'Accuracy, questa metrica non è affetta dal problema degli *unbalanced datasets (set di dati sbilanciati)*, ma torneremo su questo problema tra poco. Guardando alla matrice di confusione il valore dell'MCC è dato da:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.4}$$

- **Speed Score (SS)**: questa metrica è stata creata ed utilizzata appositamente per l'analisi delle performance dei dataset usati in questo lavoro di Tesi. Lo scopo di questa metrica è quello di misurare la velocità con cui il modello rileva l'anomalia. Ovviamente nell'Anomaly Detection è molto importante che le anomalie vengano rilevate nel minor tempo possibile. Sarà spiegato successivamente nel dettaglio la composizione del dataset, per comprendere la formula del SS basta sapere che le anomalie sono sono sempre presenti per 5 istanti di tempo (5 secondi), ovvero 5 righe del dataset. La formula per lo Speed Score è una somma pesata delle frequenze di rilevamento con decrescita quadratica e non lineare, per dare maggiore importanza alle rilevazioni nei primi istanti di tempo, diviso il totale delle anomalie rilevate.

$$SS = \frac{x_0 \cdot 1 + x_1 \cdot 0.8^2 + x_2 \cdot 0.6^2 + x_3 \cdot 0.4^2 + x_4 \cdot 0.2^2}{TotaleAnomalie} \tag{2.5}$$

  Con x_0, x_1, x_2, x_3, x_4 che indicano rispettivamente il numero di anomalie rilevata al primo, secondo, terzo, quarto e quinto istante di tempo.

Riprendiamo per un momento il problema degli unbalanced datasets. Questo problema avviene quando, in una classificazione binaria ad

esempio,la frequenza di una delle due etichette risulta sbilanciata rispetto all'altra, ovvero una delle due etichette risulta molto più frequente nel dataset rispetto all'altra. Per comprendere al meglio il problema facciamo un esempio che risalti gli aspetti critici degli unbalanced datasets. Supponiamo di avere un dataset dove il 95% dei dati è etichettato come 'normale', mentre il restante 5% è etichettato come 'anomalia'. Se avessimo un modello chiamato *dumb model* che classifica come 'normale' tutti i dati in input, avrebbe un'ACC del 95% sul nostro dataset. Se prendessimo l'ACC come metrica di riferimento, il nostro dumb model sembrerebbe un ottimo classificatore di anomalie, quando a stento potremmo definirlo classificatore. Abbiamo preso quindi l'MCC come metrica di riferimento perchè molto più robusta, non risentendo degli effetti dei set di dati sbilanciati.

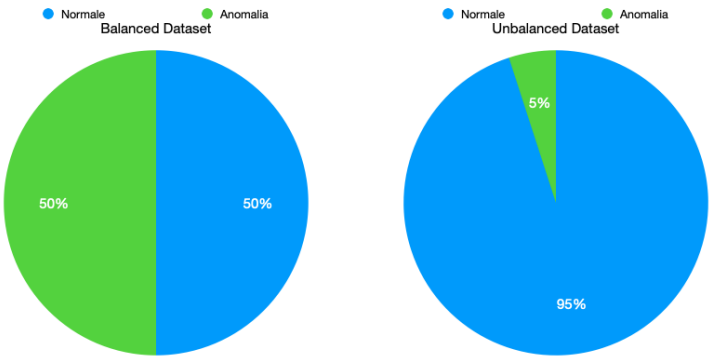

Figura 7: Balanced e Unbalanced Datasets

2.2 ANOMALY DETECTION

L'*Anomaly Detection* (o *rilevamento di anomalie* in italiano) consiste nella rilevazione di eventi rari che non rientrano nella definizione di comportamento normale dei dati. Il rilevamento di anomalie è particolarmente importante nel settore della sicurezza informatica, ma anche nei settori quali finanza, medicina e molti altri ancora.

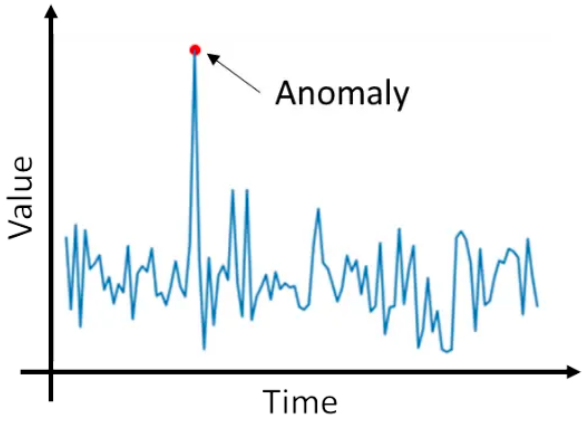

Figura 8: Esempio Anomalia

Un tempo chi si occupava di anomaly detection era solito eseaminare manualmente i dati, alla ricerca di comportamenti fuori dal normale, spesso non trovando le cause principali delle anomalie. Ad oggi il rilevamento di anomalie si basa quasi totalmente sul machine learning. Per definizione le anomalie sono eventi rari e quindi avremo a che fare spesso con dataset sbilanciati, con maggior presenza di dati etichettati come 'normali' rispetto a quelli etichettati come 'anomalie'. Le 3 categorie principali di tecniche di anomaly detection sono:

- **Anomaly Detection Supervisionata**: richiedono un set di dati etichettato in due classi, 'normale' e 'anomalia', e implicano l'addestramento di un classificatore. È il caso preso in analisi nel presente lavoro di Tesi.
- **Anomaly Detection Semi-Supervisionata**: richiedono che una porzione dei dati sia etichettata.
- **Anomaly Detection Non Supervisionata**: i dati non sono etichettati e sono sicuramente le tecniche più diffuse oggigiorno.

2.3 TIME SERIES

Il tempo è una variabile fondamentale per fare delle previsioni sul futuro. Per introdurre questo fattore nei nostri modelli ricorriamo alle *Time Series*(o *serie storiche* in italiano), che definiamo come un insieme di osservazioni ordinate rispetto al tempo. L'analisi delle serie storiche non coincide meramente con l'atto di raccogliere e analizzare dati nel tempo.

Ciò che contraddistingue una serie strorica da altri tipi di dati è che in una serie storica è possibile vedere come le variabili o features cambino nel tempo. Il tempo è una variabile cruciale e fornisce delle informazioni aggiuntive per i modelli. Le serie storiche vengono solitamente analizzate con modelli specifici per *time series forecasting*, che consiste nell'utilizzare un modello per predire valori futuri basandosi sui valori precedentemente osservati , come ad esempio il modello ARIMA. In letteratura, difficilmente vegnono utilizzati algoritmi di ML come Random Forest o XGBoost per l'analisi di serie storiche. Quello che ci siamo proposti in questo lavoro di Tesi è stato unire un problema di classificazione binaria (Anomaly Detection Supervisionata) con un approccio time series per dare maggiori informazioni ai modelli in fase di training.

3

METODOLOGIE

4

ANALISI RISULTATI

5

CONCLUSIONI

NULL

BIBLIOGRAFIA

[1] Autore - *titolo*

[2] Autore - *Titolo* - altre informazioni