Anomaly detection su i dati di uno smartwatch per rilevare le infezioni da Covid-19

Simone Gramegna

June 2, 2022

1 Introduzione

Il rilevamento delle anomalie o *Anomaly detection* è un'importante attività di analisi dei dati che rileva dati anomali o anormali da un dato insieme di dati. Si tratta di un'area interessante della ricerca sul data mining, in quanto comporta la scoperta di modelli affascinanti e rari nei dati ed è stata ampiamente studiata nell'ambito della statistica e dell'apprendimento automatico. Sebbene un'anomalia sia definita dai ricercatori in vari modi in base al suo dominio di applicazione, una definizione ampiamente accettata è quella di Hawkins ovvero *'Un'anomalia è un'osservazione che si discosta talmente tanto dalle altre osservazioni da suscitare il sospetto che sia stata generata da un meccanismo diverso'*. Le anomalie sono considerate importanti perché indicano eventi significativi ma rari e possono indurre a intraprendere azioni critiche in un'ampia gamma di domini applicativi; ad esempio, un modello di traffico insolito in una rete potrebbe significare che un computer è stato violato e i dati vengono trasmessi a destinazioni non autorizzate; un comportamento anomalo nelle transazioni con carta di credito potrebbe indicare attività fraudolente e un'anomalia in un'immagine di risonanza magnetica potrebbe indicare la presenza di un tumore maligno. L'anomaly detection è ampiamente applicata in innumerevoli ambiti , come la sanità pubblica e medica, il rilevamento delle frodi, il rilevamento delle intrusioni, i danni industriali, l'elaborazione delle immagini, le reti di sensori, il comportamento dei robot e i dati astronomici. [AMH16]

2 Algoritmi di Anomaly Detection

2.1 Tipi di anomalie

Un algoritmo per il rilevamento delle anomalie può dare in output o una *label* o etichetta utilizzata per determinare se una data istanza è un'anomalia (o meno) oppure uno *score* o punteggio ovvero un valore di confidenza che indica il grado di anormalità. Per il rilevamento delle anomalie supervisionato, spesso si utilizza un'etichetta, grazie agli algoritmi di classificazione disponibili mentre, per gli algoritmi di rilevamento delle anomalie semi-supervisionati e non supervisionati, i punteggi sono più comuni. Ciò è dovuto principalmente a ragioni pratiche, in quanto le applicazioni spesso classificano le anomalie e segnalano all'utente solo le anomalie migliori. In genere gli approcci non-supervisionati sono i più utilizzati in quanto non conosciamo a priori le istanze anomale e vogliamo scoprire i dati che differiscono significativamente dalla norma, istanze che in un dataset possono occorrere con grande frequenza ad esempio nel caso di un dataset relativo al traffico di una rete. Distinguiamo le anomalie in due tipologie:

1. *Single point anomaly*: Un'anomalia è rappresentata da una singola istanza
2. *Collective anomaly*: Un'anomalia è rappresentata da più istanze

Un altro fattore da considerare nel rilevamento delle anomalie è il contesto: un certo dato potrebbe essere normale in un certo contrsto mentre in altri contesti potrebbe essere anomalo ad esempio applicando l'anomaly detection alla frequenza cardiaca, una frequenza di 100 battiti al minuto è normale per un ragazzo mentre è anormale per una persona anziana.[GU16]

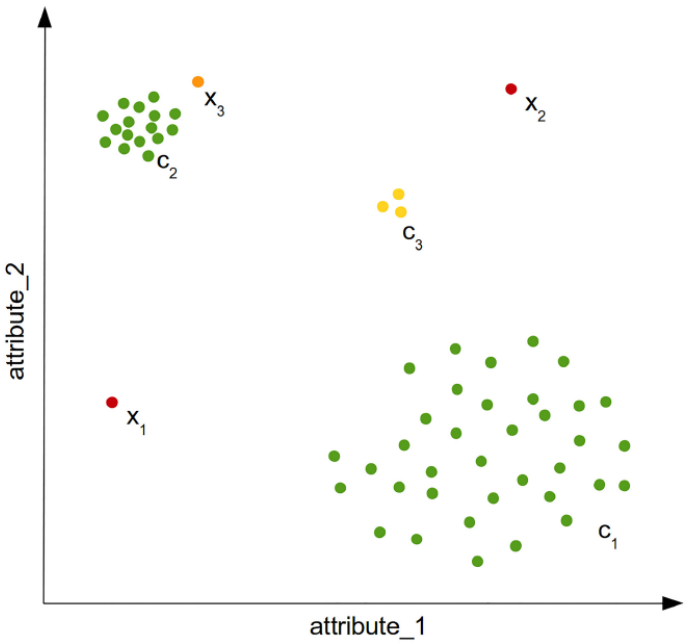

Figure 1: Tipologie di istanze

2.2 Tipi di algoritmi

Algoritmi Supervisionati
Gli algoritmi supervisionati necessitano di dati etichettati (training-set) per l'apprendimento di un modello per poi classificare le istanze di test utilizzando il modello appreso con l'assunzione che un classificatore è in grado di distinguere un'istanza normale ed un'istanza anomala mediante l'apprendimento in uno spazio di features. I modelli supervisionati più utilizzati sono:

- Alberi di decisione
- K-Nearest Neighbors
- Regressione Logistica
- Random Forest

Questi algorimi funzionano anche in presenza di più classi
Algoritmi basati su rete bayesiana
Le reti bayesiane sono utilizzate per l'anomaly detection secondo un'impostazione multi-classe, una tecnica di base per un insieme di dati categorici univariati che utilizza una rete bayesiana naıve stima la probabilità a-posteriori di osservare una label di una classe da un insieme di label di classe normali e la label di classe di anomalia, data un'istanza di dati di prova. L'etichetta di classe con la maggiore probabilità a posteriori e viene scelta come classe prevista per la data istanza di test. La probabilità di di osservare l'istanza di test, data una classe e laprobabilità a-priori della classe, viene stimata dall'insieme dei dati di addestramento. Nel caso di probabilità nulle, in particolare per le anomalie, viene effettuato lo *smmothing*, le probabilità nulle, in particolare per la classe di anomalia, vengono smussate utilizzando lo smoothing di Laplace.
Algoritmi Rule-Based
Questi algoritmi utilizzano delle regole per determinare il comportamento normale delle varie istanze e sono applicabili sia per determinare una sola classe di anomalie che per più classi. Una tecnica base rule-based multi classe consiste in due passi: 1- Apprendimento delle regole a partire dai dati di training assegnando un peso a ciascuna regola, 2 - Individuare le istanze di test che violano le regole apprese
Anomaly basati su Nearest neighbor L'assunzione alla base di questi algoritmi è la seguente: Le istanze di dati normali si verificano in "quartieri" densi (zone nello spazio in cui ci sono molti punti),

mentre le anomalie si verificano lontano dai loro vicini più prossimi, per calcolare le similarità (ad esempio la similarità del coseno). Questi algoritmi possono essere suddivisi in due principali categorie:

1. Tecniche che usano la distanza di un dato dal k-esimo vicino come anomaly score
2. Tecniche che utilizzano la densità relativa per calcolare l'anomaly score

Tecniche basate su clustering
Gli algoritmi di clustering lavorano su cluster o gruppi di istanze con lo scopo di individuare anomalie, possiamo suddividere questi algoritmi in base all'approccio utilizzato ed all'assunzione che sta alla base:

1. Le istanze normali appartengono a cluster di dati mentre le anomalie non appartengono ad alcun cluster
2. Le istanze normali sono più vicine al loro centroide mentre le anomalie sono più distanti
3. Le istanze normali appartengono a cluster grandi e densi mentre le anomalie appartengono a clusters piccoli e sparsi

Ognuno di questi approcci ha vantaggi e svantaggi, ad esempio nel primo caso lo svantaggio principale è che le tecniche basate su questo approccio sono molto efficienti nel trovare clusters ma meno nell'individuare anomalie mentre il secondo approccio lavora male in dati in cui sono presenti veri e propri clusters di punti anomali mentre gli algoritmi.
Metodi statistici
Questi metodi utilizzano regole e proprietà della statistica inferenziale con l'assunzione alla base che: le istanze normali si presentano in regioni di maggiore probabilità di un modello stocastico mentre le anomalie sono presenti in regioni con probabilità inferiore. Le tecniche statistiche adattano un modello statistico (di solito per un comportamento normale) ai dati dati e poi applicano un test di inferenza statistica per determinare se un'istanza non vista appartiene o meno a questo modello. Le istanze che hanno una bassa probabilità di essere generate dal modello dal modello appreso, in base alla statistica applicata, vengono dichiarate anomale. Per adattare un modello statistico sono state applicate sia tecniche parametriche che non parametriche. Mentre le tecniche parametriche presuppongono la conoscenza della distribuzione sottostante e stimano i parametri dai dati forniti, le tecniche non parametriche non presuppongono generalmente la conoscenza della distribuzione sottostante. [CBK09]

2.3 Deep learning e anomaly detection

Le reti neurali sono costituiti da più layers interconnessi usati nella creazione di un complesso mapping tra variabili di input e variabili di output. In particolare gli autoencoders sono reti neurali addestrate per rendere simili i valori delle features di input ai valori delle features di output minimizzando la funzione di errore. È necessario definire parametri quali: numero di layer, tipo di funzione di trasformazione, pesi oltre ad un training set abbastanza grande.[MAP+20]
Generative Adversarial Networks
Questo approccio generalmente mira ad apprendere uno spazio delle caratteristiche latenti di una rete generativa G in modo che lo spazio latente catturi bene la normalità sottostante i dati dati dati. Una qualche forma di residuo tra l'istanza reale e l'istanza generata sono poi definiti come punteggio di anomalia. L'assunzione di questo modello è il poter generare dati normali più facilmente rispetto alle anomalie.
Predictability Modelling
I metodi basati sul predictability modelling imparano le rappresentazioni delle features facendo predizione dei dati attuali sulla base dei dati precedenti usando una finestra temporale come contesto. Per ottenere previsioni accurate, le rappresentazioni rappresentazioni sono obbligate a catturare la dipendenza temporale/sequenziale e ricorrente entro una data lunghezza di sequenza. Le istanze normali sono normalmente ben aderenti a tali dipendenze e possono essere ben previste, mentre le anomalie spesso violano queste dipendenze e sono imprevedibili. Pertanto, gli errori di predizione possono essere utilizzati per definire i punteggi di anomalia. Nell'anomaly detection gli auto-encoders sono utilizzati per apprendere regolarità nei dati in modo da minimizzare l'errore di ricostruzione. Per le anomalie minimizzare questo errore è molto difficile e hanno un grande errore di ricostruzione.

3 Anomaly detection e smartwatch: stato dell'arte

L'anomaly detection, come già visto, è applicata in diversi ambiti dalla sicurezza informatica all'e-health, di particolare interesse in questo settore è il rilevamento di anomalie su dati di pazienti raccolti mediante dispositivi wearable. Questi dispositivi si sono diffusi rapidamente sul mercato negli ultimi anni, includono numerose funzionalità ma soprattutto consentono il monitoraggio in maniera non invasiva dei parametri vitali dell'utente in tempo reale, dal battito cardiaco, al livello di stress, al numero di passi. In letteratura esistono diversi studi che trattano l'anomaly detection su dati estratti da uno smartwatch, in uno studio intitolato *REsCUE: A framework for REal-time feedback on behavioral CUEs using multimodal anomaly detection* dell'università di Tokyo, i ricercatori utilizzando i dati di uno smartwatch per individuare le anomalie nel comportamento degli studenti durante le lezioni con lo scopo di migliorare le spiegazioni. L'approccio utlizzato dai ricercatori è un approccio non-supervisionato ed include oltre ai dati dello smartwatch anche i dati sul comportamento raccolti mediante telecamera. [AY19]

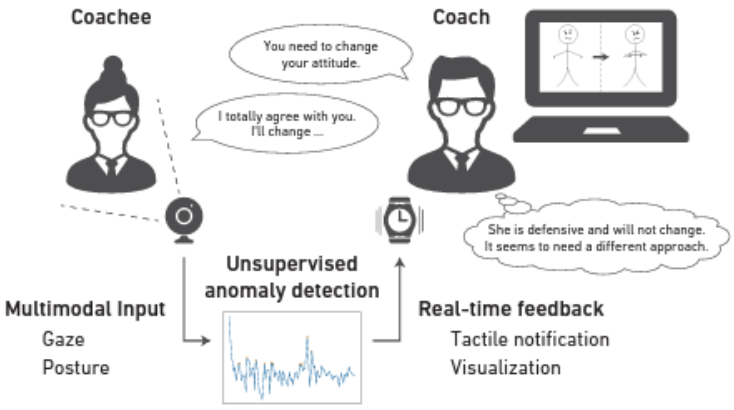

Figure 2: REsCUE rileva i segnali comportamentali dello studente e li comunica al coach in tempo reale per aiutarlo a comprendere gli stati interni dello studente.

Le tecniche di anomaly detection su i dati raccolti da dispostivi werable possono segnalare situazioni di potenziale pericolo o rischio per l'utente stesso, è il caso dello studio intitolato *AppWatchSecurity: Improving a Video Surveillance System by Integrating Smartwatch-Based Arousal Detection* che propone un prototipo di sistema che collabora con gli smartwatch alla videosorveglianza tradizionale. Combinando i concetti di user-centered design, ubiquitous wearable, psicofisiologia e Internet of Things (IoT), è presentato un sistema di videosorveglianza aggiornato in cui le anomalie basate sulla frequenza cardiaca possono far scattare automaticamente l'allarme. Come primo prototipo, il sistema è stato limitato a configurazioni sperimentali simili a quelle di una biblioteca e l'anomalia è stata definita dalla frequenza cardiaca di eccitazione - battiti cardiaci insolitamente elevati. Utilizzando uno smartwatch e un semplice questionario di tre domande,i ricercatori sono stati in grado di raccogliere i dati di riferimento sulla frequenza cardiaca di arousal di 25 soggetti sani insieme ai loro punteggi di valutazione individuali relativi a tre fattori di abitudine: fumare, bere alcolici e mangiare cibi grassi. Secondo i test semi-quantitativi sugli utenti in un ambiente bibliotecario controllato, il prototipo è stato in grado di connettersi e sincronizzare in modalità wireless tutti i dispositivi, inviare l'allarme ed eseguire la misurazione e il calcolo della frequenza cardiaca in tempo reale. [JS16]
In questo studio i ricercatori si sono concentrati sullo studio di un sistema di video-sorveglianza intelligente, video-sorveglianza e dati estratti da uno smartwatch che possono essere utili nel monitoraggio di pazienti anziani o fragili che vivono da soli la cui analisi delle anomalie unita ad altri sensori IOT possono segnalare un'emergenza medica (o una situazione simile). Lo scopo dello studio intitolato *Anomaly Detection Based on Fixed and Wearable Sensors in Assisted Living Environments* è stato proprio questo, mediante i dati raccolti da una serie di sensori non soltanto indossabili ma fissi in determinate parti della casa, dati che sono stati dati ad un modello che combina una rete neurale, una macchina a vettori di supporto SVM ed un Hidden Markov Model è stato possibile analizzare il compor-

tamento di utenti fragili e segnalare possibili problematiche al personale che si occupa dell'assistenza. [MSVJ19]
In questo studio sono stati combinati dati di sensori per il battito cardiaco, la respirazione il movimento ma anche degli accelerometri per registrare eventuali movimenti bruschi o inconsueti, su questi sensori di cui sono dotati praticamente tutti i dispostivi wearable si è concentrato lo studio della Texas State University intitolato: *SmartFall: A Smartwatch-Based Fall Detection System Using Deep Learning* in cui è presentata SmartFall, un'applicazione Android che utilizza i dati dell'accelerometro raccolti da un dispositivo Internet of Things (IoT) smartwatch basato su commodity per rilevare le cadute. Lo smartwatch è abbinato a uno smartphone che esegue l'applicazione SmartFall,eseguendo i calcoli necessari per la previsione delle cadute in tempo reale senza incorrere nella latenza di comunicazione con un cloud, preservando al contempo la privacy dei dati. Sono stati sperimentati algoritmi di apprendimento automatico tradizionali (Support Vector Machine e Naive Bayes) e non tradizionali (Deep Learning) per la creazione di modelli di rilevamento delle cadute utilizzando tre diversi set di dati (Smartwatch, Notch, Farseeing). I risultati mostrano che un modello di Deep Learning per il rilevamento delle cadute supera generalmente i modelli più tradizionali nei tre set di dati. Ciò è attribuito alla capacità del modello di Deep Learning di apprendere automaticamente caratteristiche sottili dai dati accelerometrici grezzi che non sono disponibili per Naive Bayes e Support Vector Machine, che si limitano ad apprendere da un piccolo insieme di caratteristiche estratte specificate manualmente. Inoltre, il modello di Deep Learning mostra una migliore capacità di generalizzazione a nuovi utenti nella previsione delle cadute, una qualità importante per qualsiasi modello che debba avere successo nel mondo reale. Presentiamo anche un'architettura di sistema IoT aperta a tre strati utilizzata in SmartFall, che può essere facilmente adattata per la raccolta e l'analisi di altre modalità di dati sensoriali (ad esempio, frequenza cardiaca, temperatura della pelle, modalità di camminata) che consentono di monitorare a distanza il benessere di un soggetto. [MCM+18]

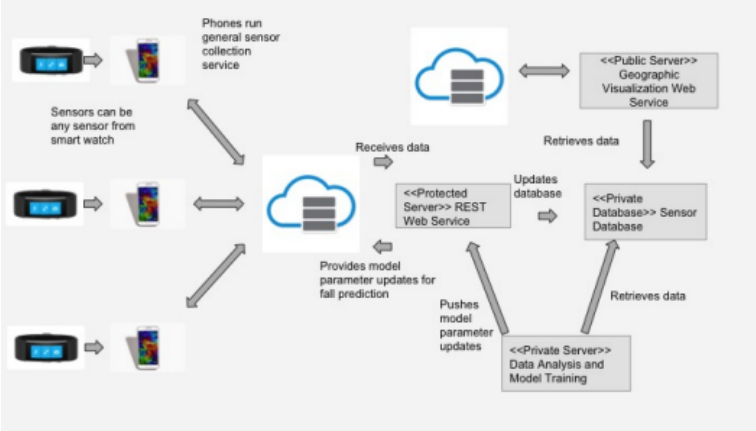

Figure 3: Architettura di SmartFall

Questi algoritmi rilevano situazioni anomale, di pericolo nel breve periodo ma è stato dimostrato che mediante i dati di uno smartwatch è possibile rilevare anomalie che durano per un periodo più lungo, la pandemia da Coronavirus ha spinto ulteriormente la ricerca su questo fronte, con gli ospedali in affanno ed il numero di casi che nel giro di pochi mesi è cresciuto in maniera esponenziale in tutto il mondo, si è creata la necessità di dover monitorare in remoto pazienti positivi al virus la cui analisi dei parametri vitali fornisce informazioni relative lo stato di salute del paziente e la gravità dell'infezione con i relativi trattamenti a cui il paziente deve essere sottoposto. Esistono in letteratura due principali studi che effettuano anomaly detection per rilevare l'infezione da Coronavirus: il primo intitolato *Pre-symptomatic detection of COVID-19 from smartwatch data* ed il secondo intitolato *Deep learning-based detection of COVID-19 using wearables data*. Entrambi gli studi sono molto simili, lavorando su i dati estratti da un Fitbit hanno dimostrato la possibilità di rilevare le infezioni mediante l'analisi di questi dati. I due studi utilizzano approcci diversi, il primo utilizza un metodo statistico non supervisionato chiamato *EllipticEnvelope* noto in letteratura mentre il secondo utlizza un modello più complesso basato su un *Autoencoder* ovvero una rete neurale in grado di generare dati dopo una fase di addestramento. Ho scelto di analizzare gli esperimenti di questi due papers in quanto si basano

su dati facilmente estraibili ovvero battito cardiaco e numero di passi inoltre i ricercatori hanno reso open-source non solo tutti i dati utilizzati nell'esperimento ma anche i modelli utilizzati. Ciò mi ha permesso di comprendere al meglio l'anomaly detection applicata ad una situazione reale, di capire i risultati descritti nel paper e di effettuare dei confronti con altri modelli che è l'obiettivo di questo mio caso di studio.

4 Deep learning-based detection of COVID-19 using wearables data - analisi dell'esperimento

In questo studio, i ricercatori hanno studiato in dettaglio i dati degli utenti provenienti dagli smartwatch Fitbit con lo scopo di rilevare retrospettivamente l'infezione COVID-19 ben prima della comparsa dei sintomi. I dati raccolti erano relativi a ritmo cardiaco e numero di passi misurati ad intervalli di 15/60 secondi. I dati di ogni partecipante sono stati divisi in allenamento o linea di base prendendo i giorni precedenti il periodo non infettivo e i dati di test prendendo i giorni durante il periodo infettivo COVID-19. I dati sono stati pre-processati per poi usare un autoencoder network-based per rilevare la frequenza cardiaca a riposo anormale nei dati del test rispetto alla linea di base dell'utente. Nello studio è stato rilevata una frequenza cardiaca a riposo anormale durante il periodo dell'infezione virale (7 giorni prima della comparsa dei sintomi e 21 giorni dopo) nel 92% (23 casi su 25) dei pazienti con COVID-19 confermata in laboratorio. Nel 56% (14) dei casi, il rilevamento LAAD ha identificato i casi nella loro fase presintomatica, mentre il 36% (9 casi) è stato rilevato dopo l'insorgenza dei sintomi con un punteggio medio di precisione di 0-91 (SD 0-13, 95% CI 0-854-0-967), un punteggio di richiamo di 0-36 (0-295, 0-232-0-487), e un punteggio F-beta di 0-79 (0-226, 0-693-0-888). Nei pazienti COVID-19 positivi, i modelli RHR anormali iniziano 5 giorni prima dell'insorgenza dei sintomi (6-9 giorni nei casi pre-sintomatici e 1-9 giorni dopo nei casi post-sintomatici). I pazienti COVID-19+ hanno periodi più lunghi di frequenza cardiaca anomala a riposo (89 ore o 3-7 giorni) rispetto agli individui sani (25 ore o 1-1 giorni).

4.1 Dataset utilizzati nell'esperimento

I dati sono stati raccolti da 25 individui positivi al covid, 11 senza sintomi da covid e 70 individui sani, su questi dati è stata utilizzata la funzione *sample* della libreria *Pandas* per assegnare casualmente una data dei sintomi per ogni individuo sano utilizzando gli intervalli di tempo dei loro dati di test con un seme casuale fisso. Per due set di dati COVID positivi senza date dei sintomi, sono state usate le date di diagnosi come date dei sintomi.

4.2 Data Pre-processing

Per ogni utente, sono stati selezioanti i dati della frequenza cardiaca unendo i dati dei passi con lo stesso esatto time-stamp della frequenza cardiaca. I dati fusi sono stati aggregati alla risoluzione di un minuto usando la funzione resample della libreria pandas. Successivamente, la frequenza cardiaca a riposo è stata calcolata selezionando gli intervalli di frequenza cardiaca in cui i passi erano zero per 12 minuti prima di un dato punto di tempo. Infine, l'utilizzo delle medie mobili (media = 400 ore) ha permesso di smussare i dati della serie temporale con la funzione *rolling* di pandas e ulteriormente aggregati alla risoluzione di un'ora prendendo la media. I dati sono stati etichettati con 3 labels:

1. Normali: Dati prima dell'infezione
2. Infettivi: Dati dei 21 giorni dell'infezione
3. Recupero: Dati dopo i 21 giorni

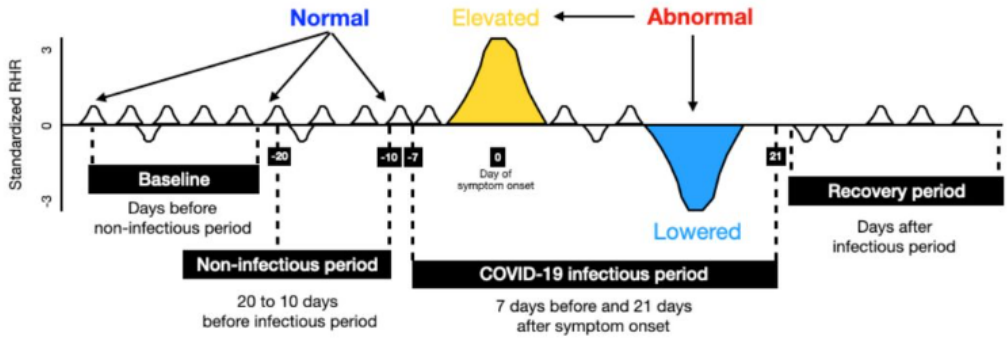

Figure 4: Labels dei dati

In questo modo i dati sono stati suddivisi in dati di training relativi al periodo in cui non c'era infezione e dati di test relativi al periodo durante e dopo l'infezione. I dati di training sono suddivisi ulteriormente in 95% dati di training e 5% dati di validazione. I dati di training e i dati di test sono stati standardizzati separatamente e trasformati in un formato tensore prima di inserirli nel framework LAAD. I dati sono stati standardizzati impostando la media a zero e la varianza a uno con l'aiuto di uno *standard scaler* della libreria sklearn. I ricercatori hanno diviso la sequenza RHR (Rest-Heart-Rate) e raggruppato i dati risultanti usando una finestra di lunghezza fissa (W) di dimensione 8. Il valore di W definisce quanti time-lag vengono elaborati dal LAAD che classifica l'input come anomalia o meno. Infine, il formato dei dati è stato rimodellato in un tensore con caratteristiche quali: numero di campioni (dimensione del lotto), numero di passi temporali per campione (lunghezza della sequenza), numero di caratteristiche.

Data augmenation

Sono state applicate 7 tecniche di data augmentation trovate in letteratura

1. Scaling: cambia la grandezza dei dati in un range moltiplicando per uno scalare. Lo scalare è stato determinato da una distribuzione Gaussiana
2. Rotazioni: applica rotazioni arbitrarie ai dati
3. Permutazioni casuali: perturba la posizione temporale degli eventi all'interno della finestra. Per perturbare la posizione dei dati in una singola finestra, prima i dati sono tagliati in N segmenti della stessa lunghezza, con N che va da 1 a 5, per poi permutare casualmente i segmenti per creare una nuova finestra.
4. Magnitude-warping: cambia la magnitudine di ogni serie temporale che è moltiplicata per una curva creata da una spline cubica con quattro nodi a magnitudini casuali
5. Time-warping: perturba la posizione temporale utilizzando una curva di warping casuale e liscia generata da una spline cubica con quattro nodi di grandezza casuale
6. Window-warping: seleziona una finestra casuale del 10% dei dati originali e lo velocizza di 2 o lo rallenta di 0.5
7. Window-slicing: una finestra grande il 90% della serie temporale originale è scelta casualmente

Applicando questi metodi sono stati creati dati di base grandi 8 volte i dati di partenza.

4.3 Autoencoder LSTM

È stato utilizzato un autoencoder con lo scopo di ricostruire i dati relativi al resting-heart-rate usando una rappresentazione codificata delle sequenza temporali di input ed è costituito da *encoder* e *decoder*. Un encoder LSTM apprende una rappresentazione del vettore di lunghezza fissa (dei dati delle serie temporali) e un decoder LSTM usa questa rappresentazione per ricostruire i dati ed il valore predetto

dal time-step precedente. I ricercatori hanno allenato un autoencoder con i dati delle sequenze temporali in cui gli utenti erano sani con un basso errore di ricostruzione per poi usarlo per rilevare le anomalie nei dati di test. Il modello è costituito da 4 livelli:

- un layer RepeatedVector
- un layer TimeDense
- 128 neuroni nascosti

L'errore di ricostruzione è calcolato secondo L'errore quadratico medio e l'algoritmo ADAM è usato per ottimizzare il processo di apprendimento. È stato impostato un valore masssimo di errore di ricostruzione dai dati di base come soglia e annotato qualsiasi valore nei dati di prova che è maggiore di questa soglia come un'anomalia.
L' *anomaly distance* è stata calcolata sottraendo data e ora dell'evento anomalo durante l'infezione dalla data di inizio dei sintomi. L'*anomaly signal strength* è stata calcolata dividendo il numero di eventi anomali nella finestra pre-sintomatica (7 giorni prima) con la perdita calcolata dall'errore quadratico medio per poi essere divisa successivamente per la lunghezza della finestra temporale e moltiplicata per 100. Nei casi post sintomatici la finestra temporale considerata è di 21 giorni, i casi con *anomaly signal strenght* maggiore di 6 sono stati raggruppati come forti mentre i restanti deboli.
Per ogni utente, sono state contate un certo numero di ore RHR anomale durante il periodo infettivo COVID-19. Per ogni utente, il delta RHR è stato calcolato sottraendo l'RHR totale delle anomalie nei dati di test (periodo infettivo COVID-19) dai dati di base/training. Il delta RHR è stato ulteriormente raggruppato come elevato se l'RHR era positivo e ridotto se l'RHR era negativo.

4.4 Dettagli di implementazione

Come prima fase, prcedente al preprocessing il Resting Heart Rate è stato dedotto dai dati sul battito cardiaco combinati con i dati sul numero di passi, il modello ha preso dei dati ingranditi dopo la fase di pre-processing (per prevenire l'over-fitting) e per ottenere prestazioni ottimali gli iper-parametri del modello sono impostati in questo modo:

- 6 layer nascosti
- 128 neuroni
- batch size = 64
- 1200 epoche
- tasso di apprendimento = 0.0001
- ottimizzatore Adam

Usando la tecnica di reconstruction loss, l'errore quadratico medio è stato usato per creare una soglia, se l'errore quadratico medio è più grande di questa soglia l'esempio di test è anomalo (il valore di soglia è stato fissato dai ricercatori).

4.5 Risultati e performance

Il modello ha avuto in media una precisione pari a 0.91 ed un recall di 0.36, questo negli individui affetti da covid. I ricercatori hanno investigato su 11 soggetti non-covid (ma malati) ed hanno scoperto segnali anomali in 7 individui (durante il periodo pre-sintomatico) ottenendo una precisione di 0.803 ed un recall di 0.472.
Nella ricerca su i 70 individui sani, 44 di loro hanno riportato eventi anomali durante il periodo pre-sintomatico selezionato casualmente mentre 15 di loro hanno riportato eventi anomali durante il periodo post-sintomatico. La precisone ottenuta è stata di 0.803 mentre il recall è stato di 0.289.
I casi COVID-19+ hanno mostrato un maggior numero di ore di RHR anomalo (89 ore o 3-7 giorni) durante il periodo infettivo rispetto ai casi non COVID-19 (87-5 ore, 3-65 giorni) e ai casi sani (25 ore, 1-04 giorni), suggerendo che l'infezione da COVID-19 dura più a lungo dei periodi anomali dei casi sani e leggermente più a lungo di altri tipi di infezione.In totale, il 78.3% (18 su 23 casi) dei casi

COVID-19+ presentava più di un giorno di segnale RHR anomalo rispetto al 70% (7 su 10 casi) dei casi non COVID19 e al 51-56% (33 su 64 casi) dei casi sani. Questi risultati suggeriscono che la durata dei RHR anormali può aiutare a predire il covid discriminandoli dai casi sani.[BS21]

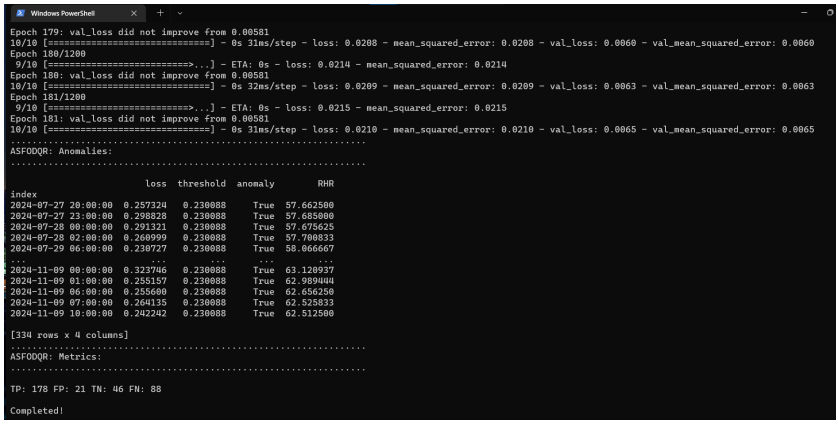

Figure 5: https://github.com/gireeshkbogu/LAAD

5 Pre-symptomatic detection of COVID-19 from smartwatch data - esperimento

Il problema precedente è stato affrontato dagli stessi scienziati usando un modello più semplice ed una feature diversa. Lo studio ha convolto 5262 partecipanti, di questi 4642 hanno dichiarato di possedere ed utilizzare uno smartwatch. Gli autori in particolare si sono soffermati sull'analisi dei dati di 32 individui, utenti di dispositivi Fitibit, i quali hanno avuto un test positivo con la data della diagnosi e l'inizio dei sintomi. Nel dettaglio i dati presi dal fitibit sono relativi a:

- Frequenza cardiaca
- Numero di passi
- Dati del sonno

dati che sono stati processati, eliminando duplicati, rimuovendo dati inverosimili (ritmo cardiaco maggiore di 200 o minore di 30) inoltre i time-stamps sono stati unificati ad un unico fuso-orario. Sono state estratte le caratteristiche della frequenza cardiaca, come la frequenza cardiaca mediana al minuto, la frequenza cardiaca media al minuto, la RHR notturna e così via. Inoltre, sono stati calcolati i passi giornalieri. Per le caratteristiche del sonno, è stata calcolata la durata totale del sonno per notte, così come la durata delle fasi di veglia, luce, profonda e REM e le loro percentuali corrispondenti per ogni notte. Questi dati sono stati integrati ai dati dei sintomi e della diagnosi provenienti da questionari compilati dagli utenti. I ricercatori hanno utilizzato un modello non-supervisionato chiamato **HROS-AD** che consiste in due step principali:

1. Nel data-preprocessing, i ricercatori hanno messo insieme i dati del ritmo cardiaco ed i dati del numero di passi creando una feature chiamata HROS. Successivamente, hanno usato medie mobili (media = 400 h) e down-sampling (media = 1 h) per regolare i dati della serie temporale e standardizzarli ulteriormente con una trasformazione del punteggio Z.

2. Nella fase di rilevamento delle anomalie, quando un punto di dati di HROS si discostava notevolmente dagli altri in un campione, veniva chiamato anomalia o outlier. Qualsiasi altra osservazione attesa è stata etichettata come un inlier.

Alla base di questo metodo vi è il fitting di una distribuzione Gaussiana ai dati per poi calcolare le distanze dei punti delle features HROS mediante la classe della libreria *sk-learn* EllipticEnvelope. Il metodo utilizza un parametro-chiave chiamato *contaminazione* che determina la proporzione di outliers da considerare, il valore di questo parametro può essere incrementato. Non sono state prese in esame

tutte le predizioni, sono state eliminate le predizioni sovrapposte tra le 6.00 e le 0.00 di ogni giorno oppure i passi mancanti nella finestra di allerta compresa tra i 7 giorni prima della comparsa dei sintomi e i 21 giorni dopo la comparsa dei sintomi. Come il modello presente nello studio precedente, anche questo esperimento utilizza la feature Resting-Heart-Rate ovvero il battito cardiaco medio a riposo, con questa feature è stato addestrato il modello Elliptic Envelope ed i risultati ottentuti da entrami i modelli sono stati confrontati. Il confronto ha evidenziato la sovrapposizione pressoché totale dei periodi in cui gli utenti dichiarano sintomi o malattia con i periodi in cui si rilevano valori anomali di HROS ed RHR. [MWM+20]

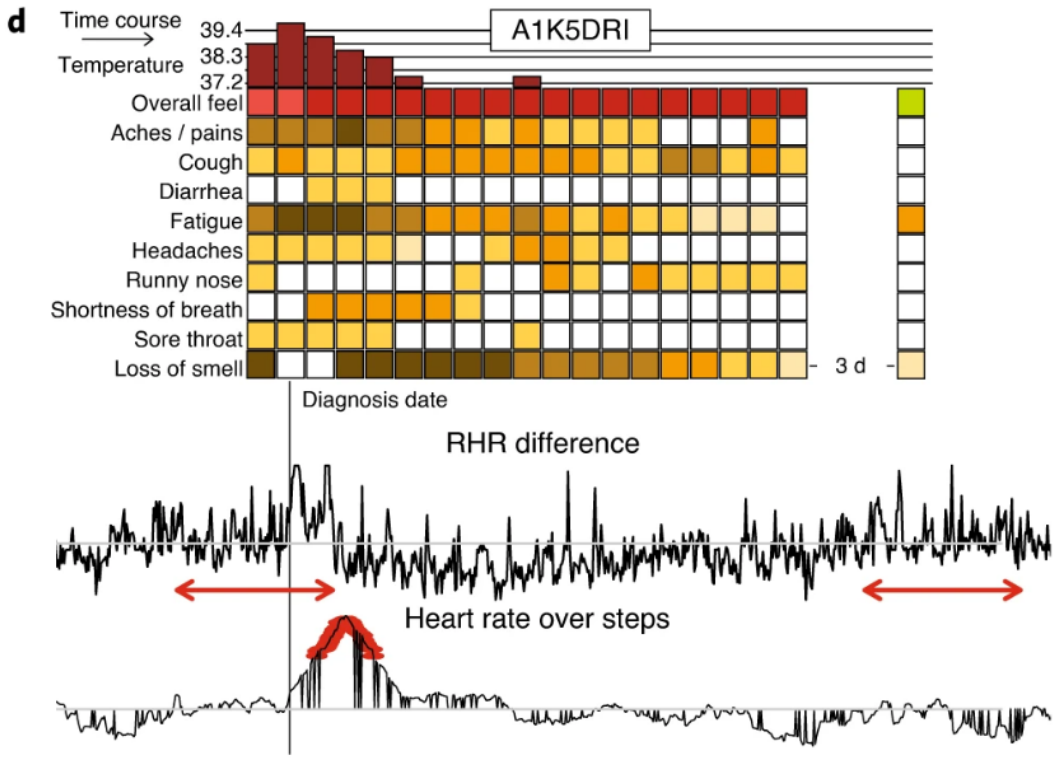

Figure 6: Anomalie e risultati di un utente

Nonostante l'accurato lavoro eseguito da parte dei ricercatori, questi modelli sono addestrati per individuare anomalie nel battito cardiaco date da infezioni da covid e non sono in grado di distinguere anomalie per altre infezioni, inoltre non si tiene conto di altre misure quali: livello di ossigenazione dal sangue, frequenza respiratoria, temperatura della pelle ed elettrocardiogramma che possono fornire preziose informazioni nel rilevamento di infezioni e malattie. Nonostante questi aspetti i modelli hanno dimostrato la possibilità di individuare le anomalie date da un'infezione in tempo reale usando dispositivi indossabili accessibili a tutti ed in modo non invasivo.

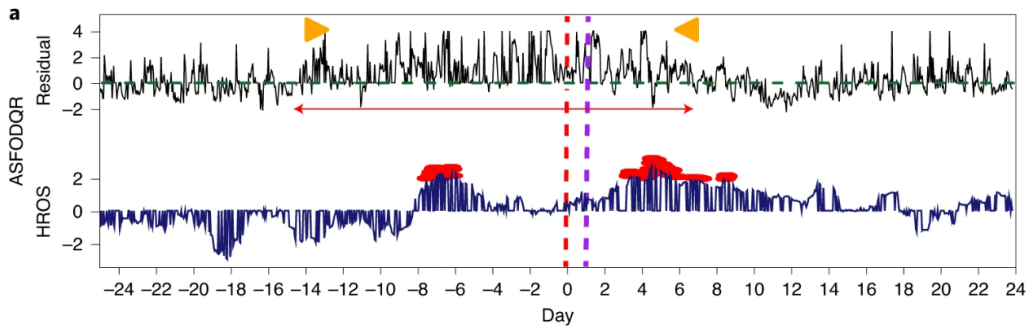

Figure 7: HROS ed RHR a confronto

6 Analisi dell'esperimento effettuato e dei risultati ottenuti

Entrambi gli esperimenti hanno l'obiettivo di individuare le anomalie in dati orari della feature Resting-Heart-Rate ovvero una feature ottenuta considerando la media del battito cardiaco in un periodo in cui per 12 minuti consecutivi il numero di passi è pari a 0, la cui analisi ha portato a rilevare effettivamente i periodi infettivi. Tuttavia ciò che manca è un confronto tra le performance dei due approcci utilizzati, il primo basato su AutoEncoder ed il secondo basato su un modello statistico ovvero l'EllipticEnvelope inoltre non ci è un confronto di altri modelli di anomaly detection conosciuti in letteratura che possono essere altrettanto validi magari fornendo risultati migliori.
Il mio caso di studio ha l'obiettivo di sopperire a questa mancanza mediante il confronto tra i risultati del primo modello (basato su autoencoder) ed i risultati ottenuti usando non soltanto l'EllipticEnvelope (già presente) ma anche applicando altri modelli di anomaly detection non supervisionati quali: IsolationForest, LocalOutlierFactor e OneClassSVM.
Il modello basato su Autoencoder, molto complesso e sicuramente uno dei migliori modelli per questo caso, è utilizzato come *gold standard* infatti le metriche calcolate per gli altri modelli (metriche non presenti nel secondo paper) sono basati sulle anomalie rilevate da questo modello. Come prima cosa eseguiamo il modello di riferimento usando i dati di battito cardiaco e numero di passi di un paziente di cui si conosce la data del tampone positivo ovvero: 14 Agosto 2024 (l'anno è falsato a causa di un problema nella conversione del timestamp misurato dal Fitbit) ed otteniamo questi risultati:

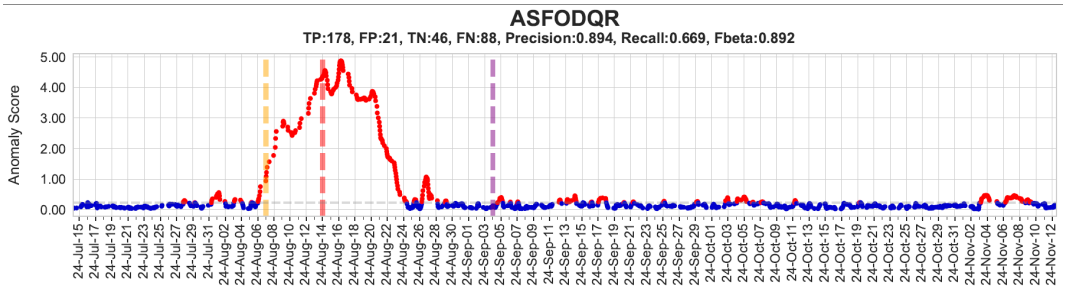

Figure 8: Risultati ottenuti con l'autoencoder

I risultati mostrano che le anomalie si verificano già a partire da una settimana prima del tampone positivo e proseguono fino ai 21 gironi successivi con eventi anomali sporadici nel periodo successivo ai 21 giorni passati dal tampone positivo. La precisione calcolata è pari a 0.894 mentre il recall è di 0.669, tutte le anomalie rilevate dal modello, pari a 350, sono salvate in un file csv chiamato *ASFODQR_anomalies.csv*, file che sarà usato per confrontare le anomalie rilevate con questo modello con le anomalie rilevate da parte dei modelli a confronto.

6.1 EllipticEnvelope

Questo modello statistico assume alla base che i dati normai sono distribuiti secondo una distribuzione conosciuta ovvero la distribuzione Gaussiana, questo modello crea una curva ellettica immaginaria attorno al dataset, tutte le istanze che rientrano in questa curva sono considerate normali, le restanti sono considerate outliers. [Ala] Questo modello è disponibile nella libreria sk-learn ed è utilizzato nel nostro esperimento.

```
def experiment_EllipticEnvelope(anomaly_truth: str):
    model = RHRAD_offline()

    df1 = model.RHR(fitbit_oldProtocol_hr, fitbit_oldProtocol_steps)
    df2 = model.pre_processing(df1)
    std_data = model.standardization(df2)

    model_EllipticEnvelope = EllipticEnvelope(
        contamination=outliers_fraction, random_state=RANDOM_SEED, support_fraction=0.7)
```

Figure 9: EllipticEnvelope nell'esperimento

In questo caso è possibile inserire la frazione di outliers (pari a 0.1 nell'esperimento) ed il random_state ovvero un generatore di numeri pseudo-causali per mescolare i dati. Una volta addestrato il modello sono effettuate le predizioni per cercare gli outliers ed i risultati dati con il confronto con le anomalie ottenute dal modello principale sono stati i seguenti:

- La precisione è pari a 0.845
- Il recall è pari a 0.915
- L'accuratezza è pari a 0.825
- L' F1 score è pari a 0.878
- L'F beta score è pari a 0.900

La matrice di confusione ottenuta è la seguente:

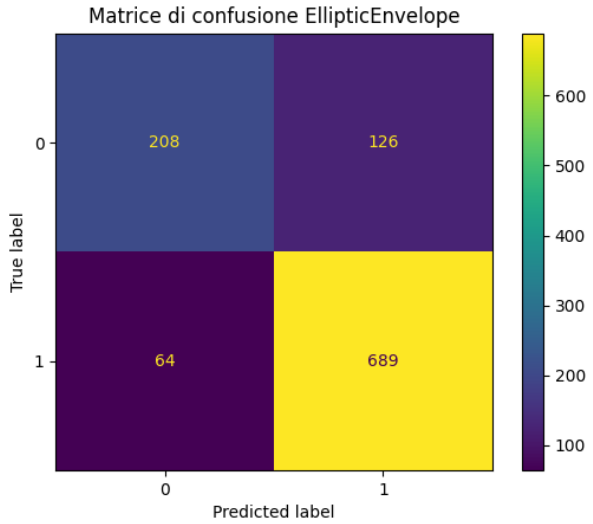

Figure 10: Matrice di confusione EllipticEnvelope

Anche questo modello ha rilevato delle anomalie nel periodo infettivo e ciò lo si può vedere dando uno sguardo al seguente grafico

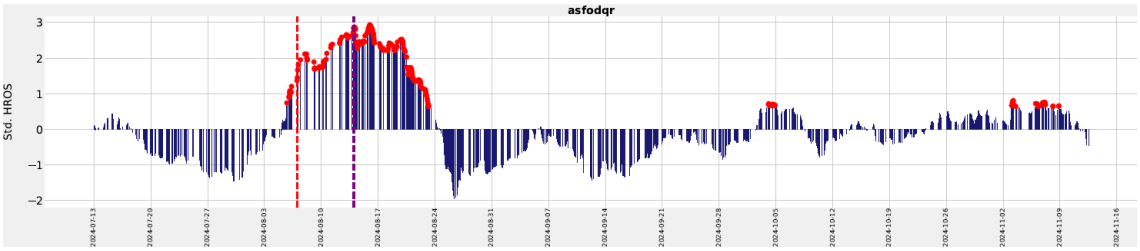

Figure 11: Anomalie rilevate - EllipticEnvelope

6.2 IsolationForest

L' IsolationForest è un modello di anomaly detection non supervisionato basato sugli alberi, i dati suddivisi in modo casuale vengono elaborati in una struttura ad albero basata su caratteristiche selezionate in modo casuale. I campioni che si trovano più in profondità nell'albero hanno meno probabilità di essere anomalie, poiché hanno richiesto più tagli per essere isolati. Allo stesso modo, i campioni che finiscono nei rami più corti indicano anomalie, poiché è stato più facile per l'albero separarli dalle altre osservazioni. [Aks]

```
def experiment_IsolationForest(anomaly_truth: str):
    model = RHRAD_offline()

    df1 = model.RHR(fitbit_oldProtocol_hr, fitbit_oldProtocol_steps)
    df2 = model.pre_processing(df1)
    std_data = model.standardization(df2)

    model_IsolationForest = IsolationForest(contamination=0.25)
```

Figure 12: IsolationForest nell'esperimento

In questo caso ho deciso di includere il 25% di outliers, i risultati ottenuti sono i seguenti:

- La precisione è pari a 0.838
- Il recall è pari a 0.907
- L'accuratezza è pari a 0.814
- L' F1 score è pari a 0.871
- L'F beta score è pari a 0.892

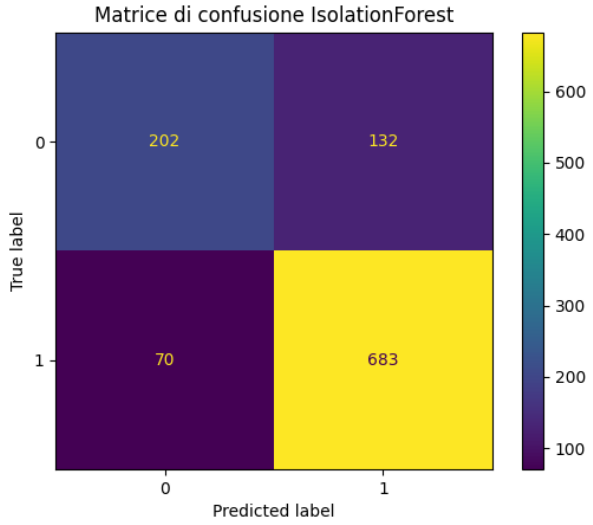

Figure 13: Matrice di confusione IsolationForest

Il grafico ottenuto per rilevare le anomalie è molto simile al precedente con la differenza che questo modello trova anomalie nelle fasi appena precedenti al periodo di incubazione e dopo molto tempo dalla comparsa dei sintomi.

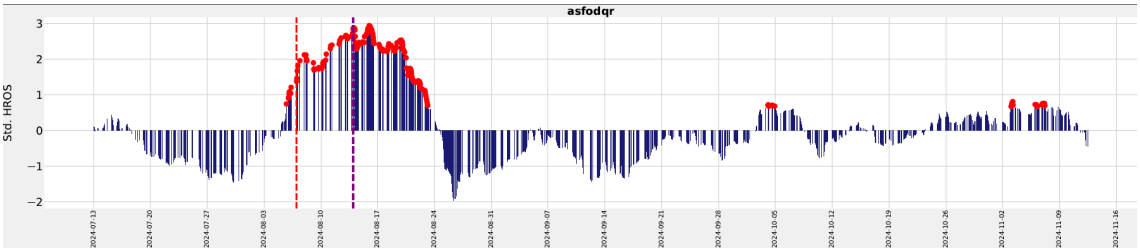

Figure 14: Anomalie rilevate - IsolationForest

6.3 LocalOutlierFactor

Il LocalOutlierFactor è un algoritmo per l'anomaly detection non supervisionato basato considerando la densità del vicinato dell'istanza considerata, l'idea alla base con cui l'algoritmo individua le anomalie è che i punti anomali sono presenti in un vicinato con minore densità di punti.

```
def experiment_LocalOutlierFactor(anomaly_truth: str):
    model = RHRAD_offline()

    df1 = model.RHR(fitbit_oldProtocol_hr, fitbit_oldProtocol_steps)
    df2 = model.pre_processing(df1)
    std_data = model.standardization(df2)

    model_LocalOutlierFactor = LocalOutlierFactor(novelty=True, contamination=0.25)
```

Figure 15: LocalOutlierFactor nell'esperimento

Le metriche calcolate per questo modello sono le seguenti:

- La precisione è pari a 0.714
- Il recall è pari a 0.812

- L'accuratezza è pari a 0.645
- L' F1 score è pari a 0.760
- L'F beta score è pari a 0.791

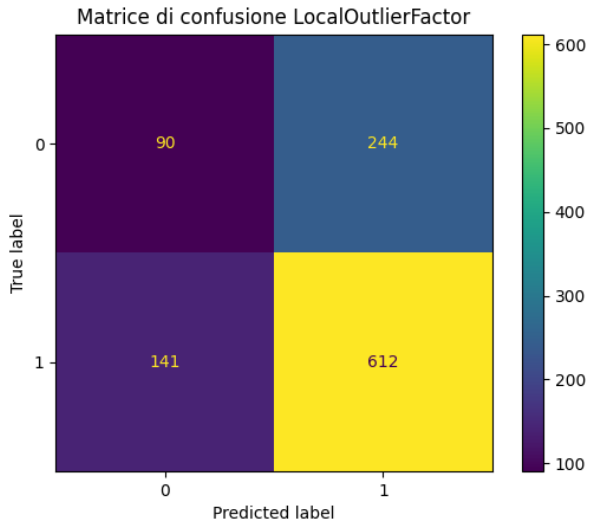

Figure 16: Matrice di confusione LocalOutlierFactor

È abbastanza evidente dal grafico che il modello non rileva gli outliers trovati da parte degli altri modelli, dando per normali punti che è evidente a occhio che sono anomalie.

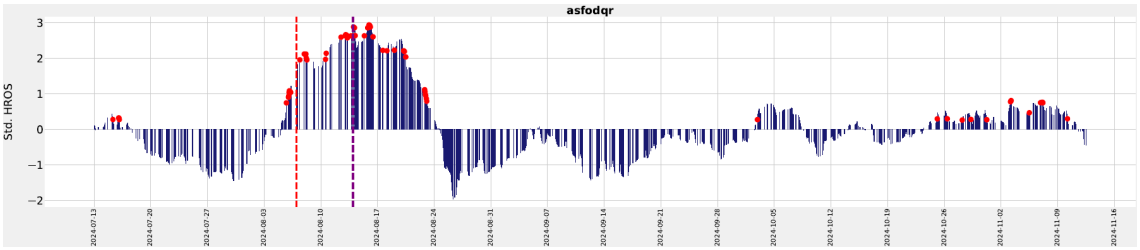

Figure 17: Anomalie rilevate - LocalOutlietFactor

6.4 One-Class SVM

Questo algoritmo nasce come algoritmo di classificazione in quanto è in grado di trovare un iperpiano nello spazio delle features che separa le due classi ma la differenza rispetto al classificatore è la seguente: quando si modella una classe, l'algoritmo cattura la densità della classe maggioritaria e classifica gli esempi agli estremi della funzione di densità come outlier.Con questa modifica della SVM *(Support Vector Machine)* l'algoritmo viene definito come SVM OneClass. [Bro]

```python
def experiment_OneClassSVM(anomaly_truth: str):
    model = RHRAD_offline()

    df1 = model.RHR(fitbit_oldProtocol_hr, fitbit_oldProtocol_steps)
    df2 = model.pre_processing(df1)
    std_data = model.standardization(df2)

    outliers_fraction = 0.25
    model_OneClassSVM = OneClassSVM(nu=0.95*outliers_fraction)
```

Figure 18: One-Class SVM nell'esperimento

Le metriche calcolate per questo modello sono le seguenti:

- La precisione è pari a 0.804
- Il recall è pari a 0.885
- L'accuratezza è pari a 0.771
- L' F1 score è pari a 0.843
- L'F beta score è pari a 0.868

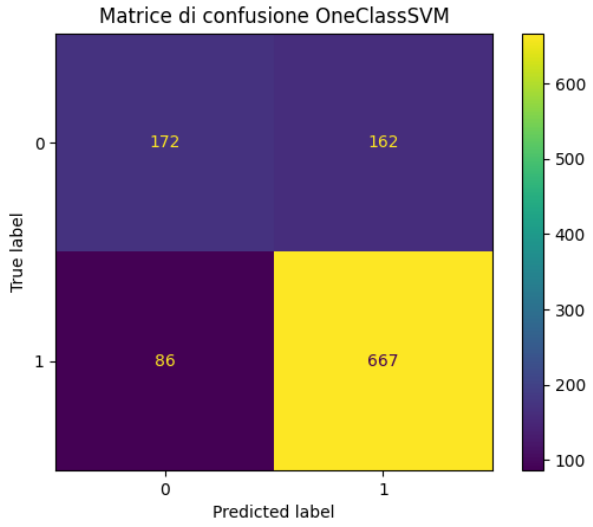

Figure 19: Matrice di confusione OneClassSVM

Dal grafico notiamo le seguenti anomalie

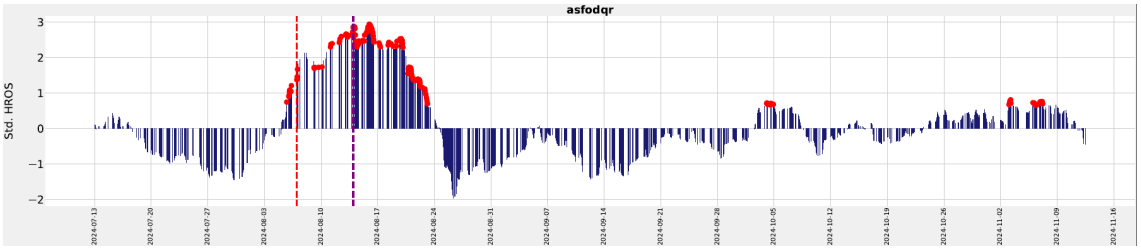

Figure 20: Anomalie rilevate - OneClassSVM

7 Conclusioni

	EllipticEnvelope	IsolationForest	LocalOutlierFactor	OneClassSVM
Accuratezza	0,825	0,814	0,645	0,771
Precision	0,845	0,838	0,714	0,804
Recall	0,915	0,907	0,812	0,885
F1 score	0,878	0,871	0,760	0,843
Fbeta score	0,900	0,892	0,791	0,868
Anomalie	208	202	90	172

Figure 21: Tabella dei risultati ottenuti

L'analisi di questi risultati porta a conclusioni interessanti, quasi inaspettate dal mio punto di vista, il modello utilizzato nell'esperimento originale ovvero L'EllipticEnvelope riesce rispetto agli altri tre modelli ad individuare la maggior parte delle anomalie trovate dall'autoencoder, prestazioni molto simili sono state ottenute applicando il modello IsolationForest mentre più distaccato come numero di anomalie rilevate troviamo il OneClassSVM ed infine il LocalOutlierFactor che sbaglia quasi tutti i punti anomali durante il periodo infettivo. Possiamo giustificare i risultati *scadenti* ottenuti da questo modello in quanto l'assunzione utilizzata dal LocalOutlierFactor per trovare anomalie è basata sulla minore densità del vicinato dei punti anomali, cosa che in questo caso non c'è in quanto trattandosi di un periodo infettivo i punti anomali sono molto vicini e densi tra di loro. Tutte le metriche ottenute dai modelli sono calcolate sulla base dei risultati ottenuti dall'Autoencoder del framework LAAD, modello che ha mostrato notevoli performance nel rilevare anomalie della *Resting Heart Rate - RHR* sia nel periodo pre-sintomatico, durante l'incubazione del virus, che nel periodo post-sintomatico il cui inizio coincide con la data del tampone positivo. Le anomalie trovate dal modello sono salvate in un file csv che viene letto dai quattro esperimenti per calcolare le metriche una volta che tutti i modelli sono stati addestrati su i dati di tutto il periodo secondo l'esperimento originale dopo che è stata calcolata la resting heart rate e sono stati standardizzati.
Questo esperimento nato a partire da due papers, i primi ad applicare l'anomaly detection a dati di un dispositivo di uso comune ovvero un semplice smartwatch, dimostra che è possibile rilevare con un buon livello di accuratezza l'infezione da Covid-19. Nonostante, grazie ai vaccini ed alla ricerca scientifica, la fase critica della pandemia sia stata oramai superata, l'esperimento offre uno spunto nell'applicazione di queste tecniche nel monitoraggio di pazienti con problemi cardiaci o respiratori in modo da segnalare al personale sanitario competente possibili situazioni anomale, di rischio intervenendo tempestivamente facendo la differenza tra la vita e la morte del paziente.
Questo esperimento può essere ulteriormente ampliato e modificato considerano i dati di uno smartwatch ma in un dominio diverso ad esempio il fitness oppure il rilevamento dello stress in determinate situazioni. A livello personale questo progetto mi ha dato la possibilità di approfondire un ambito del machine-learning che ritengo di grande interesse e che non è stata trattato in altri corsi, sicuramente userò le conoscenze apprese in questo corso ed in questo progetto in progetti futuri.

References

[Aks] Akshara. Anomaly detection using isolation forest – a complete guide. https://www.analyticsvidhya.com/blog/2021/07/anomaly-detection-using-isolation-forest-a-complete-guide/. Accessed: 25-05-2022.

[Ala] Mahbubul Alam. Machine learning for anomaly detection: Elliptic envelope. https://towardsdatascience.com/machine-learning-for-anomaly-detection-elliptic-envelope-2c90528df0a6. Accessed: 25-05-2022.

[AMH16] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31, 2016.

[AY19] Riku Arakawa and Hiromu Yakura. Rescue: A framework for real-time feedback on behavioral cues using multimodal anomaly detection. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2019.

[Bro] Jason Brownlee. 4 automatic outlier detection algorithms in python. https://machinelearningmastery.com/model-based-outlier-detection-and-removal-in-python/. Accessed: 25-05-2022.

[BS21] Gireesh K Bogu and Michael P Snyder. Deep learning-based detection of covid-19 using wearables data. *MedRxiv*, 2021.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[GU16] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173, 2016.

[JS16] Supasit Jansrithep and Thitirat Siriborvornratanakul. Appwatchsecurity: Improving a video surveillance system by integrating smartwatch-based arousal detection. In *CYTED-RITOS International Workshop on Groupware*, pages 167–175. Springer, 2016.

[MAP+20] Jorge Meira, Rui Andrade, Isabel Praça, João Carneiro, Verónica Bolón-Canedo, Amparo Alonso-Betanzos, and Goreti Marreiros. Performance evaluation of unsupervised techniques in cyber-attack anomaly detection. *Journal of Ambient Intelligence and Humanized Computing*, 11(11):4477–4489, 2020.

[MCM+18] Taylor R Mauldin, Marc E Canby, Vangelis Metsis, Anne HH Ngu, and Coralys Cubero Rivera. Smartfall: A smartwatch-based fall detection system using deep learning. *Sensors*, 18(10):3363, 2018.

[MSVJ19] Katarina Mandarić, Pavle Skočir, Marin Vuković, and Gordan Ježić. Anomaly detection based on fixed and wearable sensors in assisted living environments. In *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6. IEEE, 2019.

[MWM+20] Tejaswini Mishra, Meng Wang, Ahmed A Metwally, Gireesh K Bogu, Andrew W Brooks, Amir Bahmani, Arash Alavi, Alessandra Celli, Emily Higgs, Orit Dagan-Rosenfeld, et al. Pre-symptomatic detection of covid-19 from smartwatch data. *Nature biomedical engineering*, 4(12):1208–1220, 2020.