SBDIO I4.0
Big Data
for Industry

O4.2

Prototipo degli algoritmi per l'estrazione delle features

Code	O4.2
Date	30/09/2020
Type	Confidential
Participants	CIRI-MAM
Authors	Alberto Regattieri (CIRI-MAM), Francesca Calabrese (CIRI-MAM)
Corresponding Authors	Alberto Regattieri

POR FESR EMILIA-ROMAGNA 2014/2020
UNIONE EUROPEA
Fondo europeo di sviluppo regionale
RegioneEmilia-Romagna

SBDIO I4.0
Big Data
for Industry

Indice

SBDIO I4.0
Big Data
for Industry

Abstract

Il presente report si focalizza sugli algoritmi di estrazione delle features dai segnali grezzi raccolti da un componente/sistema in ambito industriale. Le features sono delle caratteristiche rilevanti e sintetiche dei segnali, che possono essere direttamente estratte dai segnali grezzi, o selezionate a seguito di un'analisi nei domini del tempo, frequenza e tempo-frequenza. In particolare, in questo report, verranno descritti alcuni algoritmi per la riduzione della dimensionalità, che hanno l'obiettivo di selezionare o estrarre un numero limitato di features al fine di migliorare l'accuratezza dei modelli di classificazione, o clustering (per la diagnostica) e di degradazione (per la prognostica). Tra i diversi algoritmi presenti in letteratura, il focus è stato posto su quelli che si basano su un apprendimento di tipo non supervisionato e/o incrementale, in quanto utilizzabili nei casi in cui i comportamenti al guasto dei componenti/sistemi in analisi non sono completamente noti, e pertanto particolarmente adatti nelle applicazioni industriali.

SBDIO I4.0
Big Data
for Industry

Il contesto

La manutenzione predittiva, in quanto uno dei pilastri dell'Industria 4.0., sta ricevendo un grande interesse da parte sia di accademici che di industrie. Da un lato, i primi mirano al miglioramento e l'integrazione di tecniche di apprendimento automatico, o di intelligenza artificiale, al fine di rendere gli algoritmi più robusti, efficaci, efficienti e sempre meno dipendenti dall'intervento umano. Dall'altro lato, le industrie guardano alla manutenzione predittiva come una grande opportunità sia per la riduzione dei costi derivante da una migliore gestione dei propri asset, sia come parte fondamentale di un più grande progetto di innovazione mirato alla creazione della cosiddetta "smart factory" (Mobley, 2002).

Il gap tra teoria e pratica è però ancora piuttosto rilevante. Le ragioni sono dovute principalmente alla mancanza di dati, o alla loro raccolta non propriamente funzionale allo scopo manutentivo. La maggior parte degli articoli esistenti in letteratura, che propongono diverse metodologie e tecniche con risultati sorprendenti dal punto di vista dell'accuratezza della predizione e dell'efficienza computazionale, considera sempre disponibili i dati relativi a tutte le possibili condizioni operative e di guasto. Questi vengono pertanto utilizzati per allenare i modelli per la predizione di guasti futuri. In altre parole, i metodi maggiormente utilizzati appartengono alla categoria di apprendimento supervisionato, approfondito già nel precedente report. Inoltre, anche nei casi in cui si ricorre all'apprendimento non supervisionato, resta comunque fondamentale avere disponibile un ampio dataset che porta alla creazione di modelli "statici", che quindi devono essere riallenati ogni qualvolta si presenti una nuova condizione, operativa e/o di guasto. Dal punto di vista industriale, questo rappresenta un grande limite per l'health management degli asset. In tali contesti, non si hanno nè tempo né risorse economiche da dedicare all'esecuzione di test volti alla raccolta dei dati. In alcuni casi, risulta addirittura impossibile portare a guasto un componente o un sistema, per le conseguenze potenzialmente catastrofiche che questo potrebbe comportare. Inoltre, risulta quasi impossibile raccogliere dati in continuo, data la grande mole che si genererebbe. Alla luce di tali problematiche, i "tradizionali" metodi di estrazione delle features e diagnostica, ampiamente studiati in letteratura, possono talvolta non risultare appropriati.

Una parte ancora non molto approfondita in letteratura dell'apprendimento automatico è quella che viene chiamata "Novelty Detection". Si tratta di approcci semi-supervisionati ed incrementali che, a partire da un numero limitato di condizioni note (ovvero modelli pre-allenati sulla base dei dati disponibili al momento dell'analisi), hanno l'obiettivo di rilevare un cambiamento nella struttura dei segnali raccolti e scoprire nuovi pattern, ovvero un insieme di dati rappresentativi di una nuova situazione. Tali approcci, che sebbene ancora all'esordio risultano molto promettenti, mostrano tutto il loro potenziale soprattutto nelle applicazioni streaming. Infatti, per ogni dato che gli arriva come input, riescono a definire se si tratta di una situazione già nota oppure no. Nel primo caso, grazie al modello pre-esistente è possibile definire a quale condizione il dato si riferisce; nel secondo caso, invece, è il modello stesso che viene modificato, in modo da includere la nuova condizione appena rilevata.

Come già discusso nel precedente report, i modelli per la diagnostica e la prognostica, e anche quelli per la novelty detection, richiedono come input delle caratteristiche "rilevanti" estratte dai segnali. I segnali grezzi, infatti, sono in genere raccolti a frequenze dell'ordine del kiloHertz, che rendono i modelli di classificazione (diagnostica) non efficaci nel riconoscere e separare pattern differenti. Allo stesso modo, nel segnale grezzo non è quasi mai possibile

dedurre un andamento monotono (crescente o decrescente) che sia indicativo di un qualche processo di degradazione. Pertanto, non è possibile prevedere il comportamento futuro e quindi stimare la vita utile residua (prognostica). Al contrario, dividendo il segnale in segmenti di una opportuna lunghezza ed estraendo, per ogni segmento, delle informazioni sintetiche (features), ma con una piccola perdita di contenuto informativo, risulta più evidente sia la distinzione tra le diverse condizioni di salute che il processo di degradazione. Per questa ragione, la scelta e il calcolo delle features, così come dell'indicatore di salute, risulta di estrema importanza per un'accurata attività di diagnostica/prognostica. In ottica di novelty detection e apprendimento incrementale, tuttavia, i metodi tradizionali di estrazione delle features non risultano sempre adeguati, per due ragioni principali. Primo, i metodi di processamento dei segnali nei domini del tempo, frequenza e tempo-frequenza richiedono una conoscenza più o meno approfondita del comportamento del componente/sistema in analisi in ogni condizione di guasto. Si considerino ad esempio le vibrazioni generate da un cuscinetto a sfere, che può rompersi a seguito di creazione di cricche o a seguito di un processo di usura. Nel primo caso, in genere si utilizzano tecniche nel dominio del tempo, mentre nel secondo caso, le tecniche nel dominio tempo-frequenza risultano le più efficaci. Pertanto, la sola analisi dei segnali non è adeguata per la novelty detection, nei casi di sistemi complessi e nei casi in cui i comportamenti al guasto non sono del tutto noti. A tale scopo, invece, risultano particolarmente efficaci gli algoritmi per la riduzione della dimensionalità (Tang et al., 2014), che possono essere applicati sia dopo aver estratto un numero abbastanza consistente di features nei domini tempo, frequenza e tempo-frequenza, sia direttamente ai segnali grezzi. Il loro obiettivo è quello di ridurre lo spazio dimensionale del dataset, selezionando o estraendo automaticamente le features rilevanti. In entrambi i casi, gli algoritmi sono per la maggior parte supervisionati, ma esistono alcuni versioni non supervisionate o anche incrementali.

In questo report, verranno brevemente descritti due algoritmi non supervisionati per la selezione e l'estrazione delle features. Inoltre, verrà descritto una versione incrementale dell'algoritmo di estrazione delle features, che può essere applicato ai segnali in streaming e quindi efficace per l'identificazione di comportamenti non noti.

Algoritmi per la riduzione della dimensionalità

Gli algoritmi di estrazione delle features si dividono in due categorie: analisi o processamento dei segnali e riduzione della dimensionalità

Algoritmi per la selezione delle features (non supervisionata)

In (Wei et al., 2017), viene presentata una tecnica per la selezione delle features di tipo "adaptive", che si basa su un tipo di apprendimento non supervisionato e che viene applicato a un insieme di features precedentemente estratto attraverso tecniche di analisi dei segnali. Si tratta di un algoritmo che viene applicato su batch di dati e quindi può essere utilizzato nei casi in cui non si conoscono i comportamenti del componente/sistema nelle diverse condizioni di salute/guasto, al fine di selezionare le features più rilevanti per ognuna delle condizioni. L'algoritmo include due attività: la selezione automatica delle features basata sul calcolo del peso che queste assumono per ogni condizione, e l'eliminazione delle features ridondanti. L'output del primo step è costituito da un sottoinsieme di features, dette features sensibili (SFs), estratto dall'insieme complessivo delle features; il secondo step, invece, viene utilizzato per eliminare alcune features che possono risultare ridondanti e produce come risultato l'insieme delle features ottimali.

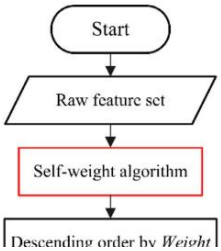
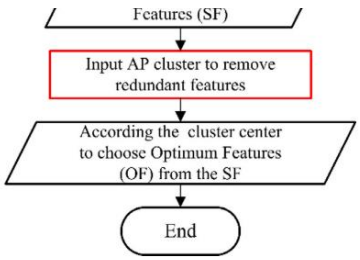


Figura 1 Algoritmo per l'estrazione delle features



Step 1. Questo step dipende soltanto dalle features stesse e attribuisce, quindi, per ogni condizione, il contributo di ogni feature senza conoscere la classe di appartenenza. Si supponga di avere un insieme di condizioni di guasto così definito: $C = \{f_{n,j},\ n = 1, \dots, N, j = 1, \dots, J\}_C$, dove $f_{n,j}$ è la *j*-esima feature dell'*n*-esima osservazione, J è il numero delle features e N è il numero di osservazioni relative a una certa condizione. Si definisce pertanto $K = N \times C$, come il numero complessivo di osservazioni disponibili. Indicando l'insieme complessivo delle features con $\{v_{k,j}\}_{k=1,\dots,K,j=1,\dots,J}$, l'algoritmo proposto prevede i seguenti passi:

1. Il valore $v_{k,j}$ viene normalizzato attraverso la seguente formula:

$$x_{k,j} = \frac{v_{k,j} - \min(v_{k,j})}{\max(v_{k,j}) - \min(v_{k,j})}, k = 1, \dots, K, j = 1, \dots, J$$

dove $x_{k,j}$ è il valore normalizzato della *k*-esima osservazione della *j*-esima feature

2. Viene calcolato il self-similarity factor $S_{mk,j}$, come segue:

$$S_{mk,j} = \left\| x_{m,j} - x_{k,j} \right\|^2, m, k = 1, \dots, K, j = 1, \dots, J$$

3. Viene calcolata la seguente matrice die pesi

$$W_j = \begin{bmatrix} S_{12,j} & \dots & S_{(K-1)K,j} \\ \dots & \dots & \\ S_{1K,j} & & \end{bmatrix}_{(K-1)\times(K-1)}$$

4. Viene infine calcolato il self-weight $Sw_j = media(W_j)$

I valori di W_j e Sw_j vengono automaticamente "aggiustati" sulla base del valore $x_{k,j}$. Infine, si ottengono tanti valori di Sw_j. Maggiore è la differenza tra le osservazioni appartenenti a diverse condizioni, maggiore è il valore Sw_j.

Step 2. Dopo aver ordinato le features in ordine decrescente di peso, si ottiene l'insieme delle features sensibili $x_{k,m}$. Dal momento che nello step 1 non vengono considerate le relazioni tra le features, è possibile che tale insieme contenga informazioni ridondanti. Pertanto, nel secondo step viene applicato un algoritmo di clustering, chiamato Affinity Propagation (AP), che serve per selezionare le features più rilevanti e contemporaneamente eliminare quelle ridondanti. Il processo si compone dei seguenti passi:

1. Si calcola la matrice trasposta dell'insieme delle features sensibili
$$X_{m,k}^T = \begin{bmatrix} x_{1,1} & \cdots & x_{1,k} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,k} \end{bmatrix}_{m\times k} = X_{m,k},\ m = 1, \dots, M, k = 1, \dots, K,$$
dove K è il numero di osservazioni e M è il numero di features sensibili selezionate durante lo step 1.
2. Viene calcolata la matrice di similarità S di $X_{m,k}$ attraverso il calcolo della distanza euclidea (negativa), come segue $S = -\left\| X_{r,m} - X_{r+i,m} \right\|^2$. Viene inoltre settata una preferenza $P = 2 * median(S)$
3. La matrice trasposta $X_{m,k}$ viene data poi in pasto all'algortimo AP per individuare il centro dei cluster, ovvero le features più rappresentative. Le features più similari tra loro vengono dunque inserite nello stesso cluster. Di conseguenza, le features ottimali sono quelle che assumono il ruolo di centro del cluster, mentre le altre features nello stesso cluster vengono considerate ridondanti e dunque eliminate.

Algoritmi per l'estrazione delle features (non supervisionata)

A differenza dei metodi di selezione delle features, i metodi per l'estrazione delle features possono essere applicati direttamente ai segnali grezzi. In genere, vengono utilizzati per segnali diversi dalle vibrazioni, che vengono campionati a frequenze molto più basse. Uno dei metodi più utilizzati per l'apprendimento automatico delle features è l'analisi delle componenti principali (PCA) (Zhu et al., 2018). Dato un dataset X, di dimensione m, la PCA trova un insieme di vettori ortonormali di dimensione p<m, chiamati componenti principali (PCs), che massimizza la varianza del dataset quando questo viene proiettato in un sottospazio racchiuso tra questi componenti. Di base, se abbiamo uno spazio bidimensionale e vogliamo proiettarlo in uno spazio a una sola dimensione, la PCA cerca la direzione del vettore e la posizione del punto su tale vettore, espresso da un coefficiente, in modo che l'errore di proiezione sia minimo. (Fig.1). A tale scopo, viene calcolata la matrice di covarianza del dataset, da cui vengono estratti gli autovalori e gli autovettori. Gli autovettori sono proprio i component principali, mentre gli autovalori dei corrispondenti autovettori rappresentano la varianza espressa da quel componente principale. I componenti principali vengono selezionati in modo che la varianza cumulativa da loro descritta è maggiore di una certa percentuale (in genere compresa tra il 90 e il 99%).

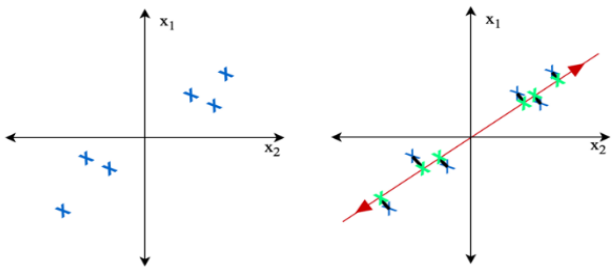


Figura 2 Riduzione della dimensionalità attraverso la PCA

Estrazione delle features per applicazioni streaming

PCA incrementale (non supervisionata)

In (Lippi & Ceccarelli, 2019), è stata introdotta una versione incrementale della PCA, che quindi non solo può essere utilizzata per l'apprendimento non supervisionato delle features, ma anche per la novelty detection (in streaming), in quanto, per ogni punto del dataset che si rende disponibile valuta i componenti principali sulla base del dataset disponibile fino a quel punto più il punto corrente. Come affermato dagli autori stessi, l'algoritmo proposto fornisce gli stessi risultati della versione batch. Inoltre, l'algoritmo contiene anche la normalizzazione delle variabili, eseguita anche questa in streaming. Di base, la differenza tra la versione batch e quella incrementale sta nel calcolo della matrice di covarianza, che diventa ricorsivo. I passi dell'algoritmo sono i seguenti:

1. Vengono calcolate la media $\bar{x}_{n(j)}$ e la deviazione standard $\bar{\sigma}_{n(j)}$ per ogni variabile j $(j = 1, \dots, m)$ sulle prime n osservazioni disponibili, al fine di calcolare la matrice standardizzata Z_n come segue

$$Z_n = \begin{bmatrix} x_1 - \bar{x}_n \\ \dots \\ x_n - \bar{x}_n \end{bmatrix} \Sigma_n^{-1} \quad (1)$$

dove $\Sigma_n \equiv diag(\sigma_n)$ è una matrice quadrata $m \times m$

2. Viene calcolata la matrice di covarianza Q_n della matrice delle features X_n come segue

$$Q_n = \frac{1}{n-1} Z_n^T Z_n \quad (2)$$

3. Viene calcolata la matrice diagonale standardizzata Q_n attraverso la matrice degli autovettori C_n

$$Q_n = C_n^{-1} \begin{bmatrix} \lambda_1 & & \\ & \dots & \\ & & \lambda_m \end{bmatrix} C_n \quad (3)$$

Dove gli autovalori λ_i sono ordinati in ordine decrescente.

4. Infine, l'evoluzione nel tempo dei valori dei componenti principali fino all'osservazione n viene calcolata come segue:

$$PC_n = Z_n C_n \quad (4)$$

5. All'osservazione $n+1$, la media e la deviazione standard vengono aggiornate e la matrice standardizzata Z_{n+1} viene calcolata come segue

$$Z_{n+1} = \begin{bmatrix} Z_n \Sigma_n + \Delta \\ y \end{bmatrix} \Sigma_{n+1}^{-1} \quad (5)$$

dove $y = x_{n+1} - \bar{x}_{n+1}$, Δ è una matrice $n \times m$ matrix costruita ripetendo n volte il vettore the vector $\delta = \bar{x}_n - \bar{x}_{n+1}$

6. Viene quindi calcolata la matrice di convarianza nQ_{n+1} come segue

$$nQ_{n+1} = Z_{n+1}^T Z_{n+1} \quad (6)$$

Che dipende soltanto dalla matrice di covarianza calcolata al punto precedente e il vettore di features $\boldsymbol{x}_{n+1}$ corrispondente all'osservazione $n+1$.

7. Infine, la matrice aggiornata Q_s viene utilizzata per calcolare il valore *n*-esimo values dei componenti principali attraverso l'Eq. (4).

Esempio di applicazione

Di seguito si riporta un esempio di applicazione della PCA incrementale. Il dataset di partenza è stato estratto da una macchina industriale su un orizzonte temporale di circa un anno. L'obiettivo dell'analisi è il riconoscimento, in real-time del cambio della condizione operativa implementata in macchina, la quale è determinata da un insieme di 11 segnali di temperatura, raccolti ad una frequenza di 1 Hz. Come si vede in Figura 3. Nel periodo di analisi si sono implementati due setting diversi.

Period	Setting
From 2017-10-20 to 2017-11-03	1
From 2017-12-04 to 2018-09-17	2

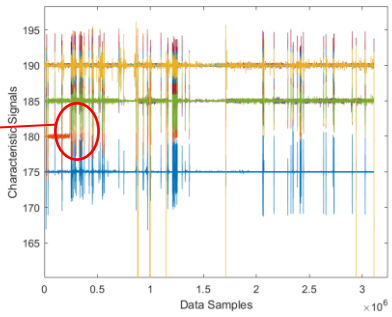


Figura 3 Dataset di partenza

A questo dataset è stata quindi applicata la PCA incrementale, al fine di estrarre un sottoinsieme di features che fosse capace di rivelare il cambio da una condizione all'altra. In Figura 4a si riporta l'andamento dei 4 componenti principali estratti, che come si vede in Figura 4b contengono il 90% della varianza. Da questa stessa figura, inoltre, si vede come la PCA incrementale fornisca gli stessi risultati della versione tradizionali in batch. Questi componenti principali, ogni volta che vengono aggiornati al fine di includere la nuova osservazione disponibile, sono stati utilizzati come input di un algoritmo di clustering ricorsivo. Questo algoritmo, sulla base dei componenti principali, è in grado di riconoscere il cambio di condizione operativa e, contemporaneamente, assegnare le osservazioni corrispondenti alla stessa condizione allo stesso cluster (Figura 5).

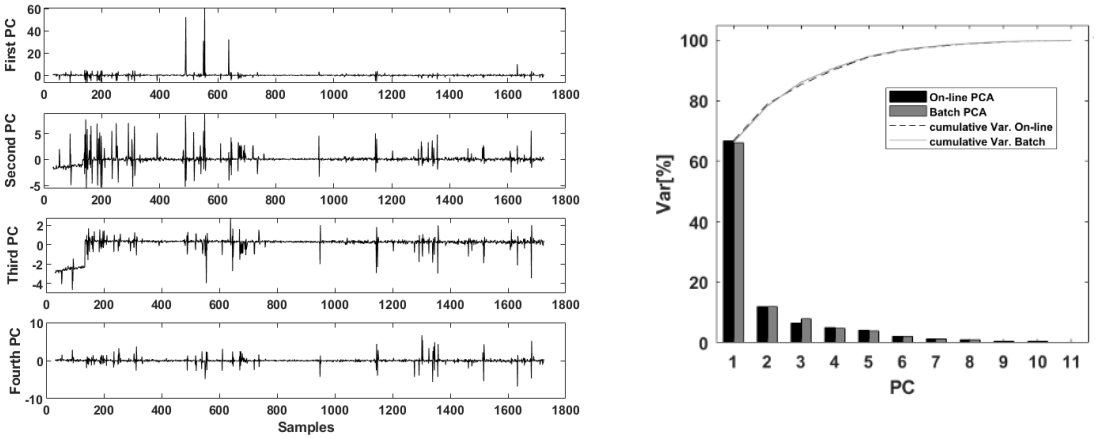


Figura 4 a. Componenti Principali; b. Confronto fra PCA in batch e PCA incrementale

Per dimostrare l'efficacia della PCA, lo stesso algoritmo di clustering è stato applicato all'insieme completo delle features (gli 11 segnali di temperatura). Come si vede in Tabella 1, nel caso di utilizzo di PCA, l'algoritmo riesce a riconoscere con una latenza inferiore il cambio di stato e non produce falsi positivi (cambi di stato non verificatisi nella realtà).

Tabella 1. Risultati del clustering con e senza PCA.

PCA	Cambio rilevato (numero dell'osservazione)	Cambio reale (numero dell'osservazione)

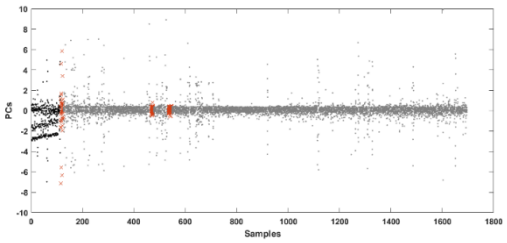


Figura 5 Risultati del clustering con PCA incrementale

Estrazione dell'indicatore di salute

Le features estratte con gli algoritmi descritti nei precedenti paragrafi sono principalmente utilizzate per la diagnostica, per analisi in batch, o per la novelty detection, per applicazioni streaming. Tuttavia, per la prognostica, che ha l'obiettivo di calcolare la vita utile residua del componente, bisogna calcolare una feature monotona, che prende il nome di indicatore di salute e che è in grado di rivelare il processo di degradazione. A tal scopo, non esistono modelli di tipo non supervisionato, in quanto risulta ancora necessaria la conoscenza del comportamento del componente al fine di identificare sia il segnale, tra quelli disponibili, che è in grado di descrivere la degradazione e sia quale features assume un andamento monotono. È possibile, tuttavia, una volta identificato l'indicatore, calcolarlo in streaming, estraendolo ogni volta che il segmento di segnale (della lunghezza che si sarebbe scelta nelle analisi batch) si rende disponibile.

Ad esempio, si riporta l'estrazione dell'energia nel dominio tempo-frequenza, attraverso la tecnica Empirical Mode Decomposition (EMD), già descritta nel precedente report. La prima scelta da compiere è la lunghezza dei segmenti di segnale. Si tratta di una scelta arbitraria, che spesso dipende dalla velocità di rotazione della macchina, ma anche dalla latenza

dell'algoritmo utilizzato o dal numero di osservazioni che si vogliono avere. Una volta scelto tale parametro, ogni volta che un intero segmento si rende disponibile, viene applicata la EMD e calcolata l'energia degli IMF così ottenuti.

Esempio di applicazione

Si consideri il segnale mostrato in Figura 6a, che rappresenta il comportamento di un cuscinetto durante un test accelerato (Nectoux et al., 2012), fino alla rottura. In letteratura, è stato dimostrato che per segnali di vibrazioni raccolti dai cuscinetti, l'analisi in tempo-frequenza è quella più efficace per l'estrazione dell'indicatore di salute. Pertanto, in questa applicazione, si è considerata una delle tecniche maggiormente utilizzate, l'EMD. Questa è stata applicata ogni secondo, quindi 25600 osservazioni. In Figura 6b, sono mostrati sia gli IMF ottenuti dall'ultimo segmento di segnale analizzato, sia l'andamento dell'energia nel tempo. Il risultato è lo stesso che si otterrebbe applicando l'EMD al dataset batch e calcolando il valore dell'energia per ogni segmento di segnale.

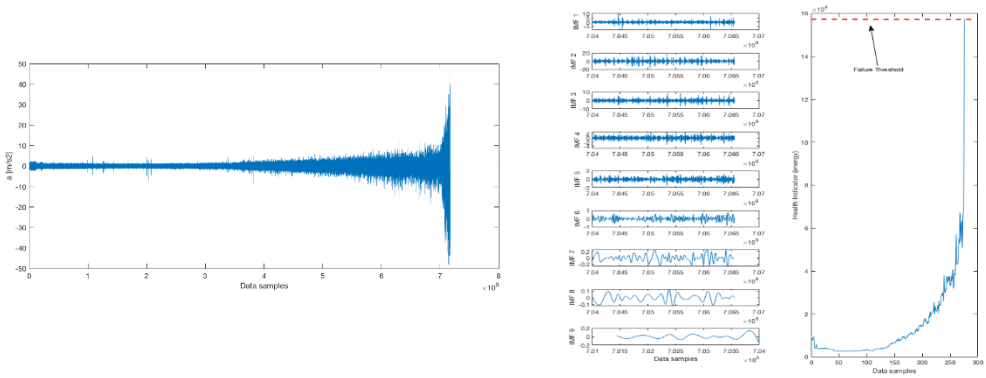


Figura 6 a. Segnale di vibrazione di un cuscinetto fino alla rottura; b. IMF calcolati attraverso l'EMD applicata in streaming e andamento dell'indicatore di salute

Referenze

Lippi, V., & Ceccarelli, G. (2019). Incremental principal component analysis: Exact implementation and continuity corrections. *ICINCO 2019 - Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics*, *1*, 473–480. https://doi.org/10.5220/0007743604730480

Mobley, R. K. (2002). Role of Maintenance Organization. In *An Introduction to Predictive Maintenance* (pp. 43–59). https://doi.org/10.1016/b978-075067531-4/50003-8

Nectoux, P., Gouriveau, R., Medjaher, K., Ramasso, E., Chebel-morello, B., Zerhouni, N., Varnier, C., Nectoux, P., Gouriveau, R., Medjaher, K., Ramasso, E., Chebel-morello, B., Nectoux, P., Gouriveau, R., Medjaher, K., Ramasso, E., Morello, B., Zerhouni, N., & Varnier, C. (2012). PRONOSTIA: An experimental platform for bearings accelerated degradation tests. *EEE International Conference on Prognostics and Health Management, PHM'12*, *1–8*.

Tang, J., Alelyani, S., & Liu, H. (2014). Feature Selection for Classification: A Review. *Data Classification: Algorithms and Applications*, 37–64. https://doi.org/10.1.1.409.5195

Wei, Z., Wang, Y., He, S., & Bao, J. (2017). A novel intelligent method for bearing fault diagnosis based on affinity propagation clustering and adaptive feature selection. *Knowledge-Based Systems*, *116*, 1–12. https://doi.org/10.1016/j.knosys.2016.10.022

Zhu, Y., Zhang, X., Wang, R., Zheng, W., & Zhu, Y. (2018). Self-representation and PCA embedding for unsupervised feature selection. *World Wide Web*, *21*(6), 1675–1688. https://doi.org/10.1007/s11280-017-0497-2