

# UNIVERSITA' POLITECNICA DELLE MARCHE

---

## FACOLTA' DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

Manutenzione Preventiva per la Robotica e l'Automazione Intelligente



### ***PROGETTO B1: PHM 2022 Data Challenge***

<i>Studenti:</i>	<i>Docenti:</i>
<i>Angelone Mattia Domenico</i>	<i>Alessandro Freddi</i>
<i>Romanelli Marco</i>	
<i>Ali Waqar Badar</i>	

*Anno Accademico A.A.2022/2023*

# Sommario

1. INTRODUZIONE .....	4
2. MATERIALI E METODI .....	7
2.1 Dataset .....	7
2.2 Step Operativi .....	8
3. RISULTATI .....	10
3.1 Preparazione e caricamento del dataset .....	10
3.1.1 Caricamento dei file CSV .....	10
3.1.2 Pulizia del dato .....	11
3.2 Estrapolazione delle Features .....	13
3.2.1 Calcolo delle features nel tempo .....	13
3.2.2 Calcolo delle features in frequenza .....	15
3.2.3 Features .....	17
3.2.4 Ranking .....	19
3.3 Addestramento classificatori .....	20
3.4 Test del classificatore .....	21
4. CONCLUSIONI .....	24



## 1. INTRODUZIONE

La diagnostica di guasti è un processo importante per identificare e risolvere problemi in diversi sistemi tecnologici. La tecnologia dei classificatori multi classe offre un modo efficiente per diagnosticare i guasti in modo automatico. Un classificatore multi classe utilizza algoritmi di apprendimento automatico per analizzare i dati di input e assegnare l'anomalia ad una delle molte categorie predefinite.

La diagnostica di guasti con un classificatore multi classe è un processo che combina la tecnologia dell'apprendimento automatico con l'analisi dei dati per identificare i problemi in un sistema. Il classificatore multi classe è alimentato da una vasta quantità di dati sul funzionamento del sistema che sono stati raccolti in precedenza. Questi dati vengono utilizzati per addestrare l'algoritmo di apprendimento automatico a riconoscere pattern e relazioni tra le varie caratteristiche del sistema.

Quando viene rilevato un problema, il classificatore multi classe analizza i dati in entrata e utilizza le conoscenze acquisite durante l'addestramento per assegnare il problema a una categoria predefinita. Ad esempio, se un sistema di climatizzazione non funziona correttamente, il classificatore potrebbe identificare il problema come un problema di refrigerazione, un problema di ventilazione o un problema di controllo.

La diagnostica di guasti con un classificatore multi classe è molto utile perché offre una soluzione automatizzata e precisa per identificare i problemi. Questo metodo può risparmiare tempo e costi rispetto ai metodi di diagnostica tradizionali che possono essere più manuali e meno precisi. Inoltre, il classificatore può essere continuamente addestrato con nuovi dati per migliorare la sua precisione e mantenere la sua efficacia nel tempo.

Ecco i passi fondamentali per addestrare un classificatore multi classe:



1. **Misure.** Si misurano gli ingressi e le uscite mediante l'utilizzo di sensori. È importante che tali misure vengano prese in assenza e presenza di guasto ed in differenti condizioni operative. La difficoltà sta proprio nell'andare a raccogliere le misure in presenza di guasto e nelle differenti condizioni operative; tuttavia, se si ha la conoscenza ingegneristica si può fare una simulazione. In tal caso, mediante i sensori si raccolgono le misure senza guasto (sensor data) e poi mediante il modello di simulazione si raccolgono delle misure virtuali in cui si fanno comparire dei guasti (synthetic data). A questo punto i synthetic data si vanno ad aggregare con i sensor data.

2. **Filtraggio.** Un dataset non filtrato correttamente ha una bassa affidabilità. Il filtraggio permette di togliere il rumore e rimuovere gli outliers, ovvero quei valori distanti da ciò che è atteso.
3. **Estrazione feature.** Una feature, in generale, è una caratteristica che ha al suo interno l'informazione diagnostica. Nel caso della manutenzione preventiva la feature è tutto quello che nel tempo tende ad avere un andamento che cresce o decresce; quindi, è una grandezza derivata che ha delle proprietà caratteristiche che si possono utilizzare. Le feature possono derivare dalla conoscenza del modello (ad esempio dai residui) oppure dall'analisi del segnale (con cui si andranno a prendere i valori medi o le varianze in diverse finestre temporali).
4. **Addestramento predittore.** Supposto di aver realizzato delle feature utili vanno date in ingresso ad un predittore, che può essere un classificatore diagnostico, quindi quando c'è un'anomalia che supera una soglia definisce la tipologia dell'anomalia stessa per poi pilotare la manutenzione . Oppure si può effettuare prognosi ovvero stimare quanto tempo utile di vita il macchinario ha a disposizione, cioè fare la stima della RUL (Remaining Useful Life).
5. **Integrazione.** Dopo aver sviluppato l'algoritmo di diagnosi e prognosi si fa il deployment del modulo di predizione a bordo della macchina, in cui poi solo i dati aggregati si mettono in cloud e si fa su di essi analitica.

Ai fini del progetto sono stati implementati i primi 4 passi per l'addestramento di un classificatore multi classe in ambito diagnostico.



## 2. MATERIALI E METODI

### 2.1 Dataset

Le misure utilizzate per la realizzazione di questo progetto sono dati provenienti da rilevamenti di 3 sensori di pressione installati su una rocciatrice idraulica nella quale sono stati indotti guasti di diverse tipologie.

Una rocciatrice idraulica, anche nota come martello idraulico, è un attrezzo utilizzato per rompere rocce, calcestruzzo o asfalto utilizzando un'energia cinetica elevata generata da un pistone azionato da un sistema idraulico. È costituita da quattro sistemi principali:

- a) il sistema a percussione, di cui fa parte il pistone;
- b) il sistema di rotazione, in cui un motore idraulico ruota la barra di perforazione;
- c) Il sistema di smorzamento, in cui le onde di stress riflesse dalla roccia vengono dissipate come calore;
- d) il sistema di lavaggio, in cui un mezzo di lavaggio entra nella barra di perforazione.

Le misurazioni si sono concentrate sul sistema a percussione ed il sistema di smorzamento con l'utilizzo di 3 sensori posti in tre diverse parti del sistema:

Sensore	Frequenza di campionamento	Descrizione
Pin	50kHz	Pressione di percussione al raccordo di ingresso
Pdmp	50kHz	Pressione di smorzamento all'interno della camera esterna
Po	50kHz	Pressione nel volume dietro al pistone

Per ogni sensore è stato creato un dataset che contiene in ogni riga un ciclo di funzionamento con all'inizio una label che va ad indicare il tipo di guasto presente in quel ciclo di lavoro tra 11 classi di guasti:

Label	Lettera	Descrizione
1	NF	Nessun guasto
2	T	Barra di perforazione più spessa
3	A	Manca il sigillo A. Perdita dal canale ad alta pressione al canale di controllo
4	B	Manca il sigillo B. Perdita dal canale di controllo al canale di ritorno
5	R	Accumulatore di ritorno danneggiato
6	S	Barra di perforazione più lunga
7	D	L'orifizio dello smorzatore è più grande del solito
8	Q	Basso flusso nel circuito dello smorzatore
9	V	Danno alla valvola. Una piccola usura su uno dei sedili della valvola
10	O	L'orifizio sulla linea di controllo di uscita è più grande del solito

11	C	Il livello di carica nell'accumulatore ad alta pressione è basso
----	---	--

Ai fini dell'obiettivo di classificazione diagnostica dei guasti sono state adoperate 6 diverse rocciatrici dalle quale sono stati estrapolati 3 diversi dataset con le misurazione effettuate dai 3 sensori di pressione, ognuno dei quali è formato da 300-700 righe equivalenti ai diversi cicli di lavoro della macchina ai quali è stato applicato un label corrispondente al tipo di guasto presente.

Il dataset finale è stato diviso in 2 sottogruppi:

- a) Training Data: dataset di 5 rocciatrici adoperati per l'addestramento del classificatore diagnostico
- b) Testing Data: dataset di una rocciatrice per il testing del classificatore addestrato attraverso il Training Dataset

## 2.2 Step Operativi

Il progetto verrà implementato tramite l'ambiente di programmazione MatLab con l'ausilio di due App: il Diagnostic Feature Designer ed il Classification Learner.

I passi per lo svolgimento del lavoro sono i seguenti:

1. Importazione del training dataset in MatLab e creazione di una Timetable comprendente tutti i dati di tutte le 5 rocciatrici divisi per sensori e campionati temporalmente.
2. Importazione della Timetable di training nel Diagnostic Feature Designer per l'estrapolazione delle features nel tempo ed in frequenza con relativo ranking.
3. Importazione delle features nel Classification Learner ed addestramento di vari classificatori diagnostici per scegliere il migliore in termini di accuracy.
4. Esportazione del modello del classificatore e testing attraverso il testing dataset con valutazione finale attraverso la distribuzione dei risultati.





## 3. RISULTATI

In questo capitolo saranno esplicitati i vari step operativi precedentemente elencati.

### 3.1 Preparazione e caricamento del dataset

Il caricamento del dataset è tra i passaggi più importanti di qualunque progetto, in quanto questa fase deve fornire i dati sul quale fare elaborazione.

L'aspetto fondamentale è l'ETL sui dati (Extract, Transform, Load): questo è un processo di integrazione dei dati che prevede la pulizia in primis e la trasformazione per renderli uniformi e coerenti, in modo tale da poterli caricare in un sistema di destinazione finale.

Nel nostro caso la fase di ETL come l'intero progetto è stata fatta con MatLab.

#### 3.1.1 Caricamento dei file CSV

##### CARICAMENTO FILE CSV DI TRAINING COME MATRICI

```
pdpm1=readmatrix("Data_Challenge_PHM2022_training_data\data_pdmp1.csv","Delimiter",',');  
pin1=readmatrix("Data_Challenge_PHM2022_training_data\data_pin1.csv","Delimiter",',');  
po1=readmatrix("Data_Challenge_PHM2022_training_data\data_po1.csv","Delimiter",',');
```

```
pdpm2=readmatrix("Data_Challenge_PHM2022_training_data\data_pdmp2.csv","Delimiter",',');  
pin2=readmatrix("Data_Challenge_PHM2022_training_data\data_pin2.csv","Delimiter",',');  
po2=readmatrix("Data_Challenge_PHM2022_training_data\data_po2.csv","Delimiter",',');
```

```
pdpm4=readmatrix("Data_Challenge_PHM2022_training_data\data_pdmp4.csv","Delimiter",',');  
pin4=readmatrix("Data_Challenge_PHM2022_training_data\data_pin4.csv","Delimiter",',');  
po4=readmatrix("Data_Challenge_PHM2022_training_data\data_po4.csv","Delimiter",',');
```

```
pdpm5=readmatrix("Data_Challenge_PHM2022_training_data\data_pdmp5.csv","Delimiter",',');  
pin5=readmatrix("Data_Challenge_PHM2022_training_data\data_pin5.csv","Delimiter",',');  
po5=readmatrix("Data_Challenge_PHM2022_training_data\data_po5.csv","Delimiter",',');
```

```
pdpm6=readmatrix("Data_Challenge_PHM2022_training_data\data_pdmp6.csv","Delimiter",',');  
pin6=readmatrix("Data_Challenge_PHM2022_training_data\data_pin6.csv","Delimiter",',');  
po6=readmatrix("Data_Challenge_PHM2022_training_data\data_po6.csv","Delimiter",',');
```

Questo primo passo consiste nel semplice caricamento dei singoli file CSV come matrici. Per semplicità non verrà approfondito.

### 3.1.2 Pulizia del dato

Questa seconda fase è fondamentale per ottenere dati uniformi e coerenti.

#### ELIMINAZIONE COLONNE DOVE SONO PRESENTI ELEMENTI NAN

```
pdp1=rmmissing(pdp1,2);
pin1=rmmissing(pin1,2);
po1=rmmissing(po1,2);
```

#### CREAZIONE VETTORE DEI FAULT CODE

```
faul_code_1 = pdpm1(:,1);
faul_code_2 = pdpm2(:,1);
faul_code_4 = pdpm4(:,1);
faul_code_5 = pdpm5(:,1);
faul_code_6 = pdpm6(:,1);
```

#### ELIMINAZIONE DELLA COLONNA DEI FAULT CODE DALLE VARIE MATRICI

```
pdp1(:,1) = [];
pin1(:,1) = [];
po1(:,1) = [];
```

#### TRASPOSIZIONE DELLE MATRICI

```
pdp1_T = transpose(pdp1);
pin1_T = transpose(pin1);
po1_T = transpose(po1);
```

#### CREAZIONE DI UNA TABLE INSERENDO IN OGNI ELEMENTO UNA COLONNA DELLA RELATIVA MATRICE

```
DATA = table();

for i = 1:7311
    DATA.Pdmp(i) = {array2timetable(pdp1_T(:,i),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.Pin(i) = {array2timetable(pin1_T(:,i),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.Po(i) = {array2timetable(po1_T(:,i),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.FaultCode(i) = {faul_code_1(i,1)};
end

for j = 1:7867
    DATA.Pdmp(i+j) = {array2timetable(pdp2_T(:,j),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.Pin(i+j) = {array2timetable(pin2_T(:,j),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.Po(i+j) = {array2timetable(po2_T(:,j),"VariableNames',{'Data'}','SampleRate',50000)};
    DATA.FaultCode(i+j) = {faul_code_2(j,1)};
end
```


Come è possibile vedere negli spezzoni di codice, è stata fatta una pulizia di dati mancanti per i 3 sensori di ogni rocciatrice :

- della prima rocciatrice sono stati eliminati 2.873.223 dati su un totale di 16.427.817;
- della seconda rocciatrice sono stati eliminati 2.973.726 dati su un totale di 17.228.730;
- della terza rocciatrice sono stati eliminati 2.848.875 dati su un totale di 16.204.401;
- della quarta rocciatrice sono stati eliminati 3.015.306 dati su un totale di 16.895.286;
- della quinta rocciatrice sono stati eliminati 1.195.359 dati su un totale di 6.846.147;

Successivamente è stata effettuata una estrapolazione della colonna contenente le label dei guasti per ogni CSV, una trasposizione delle matrici per poterle trasformare in un dato di tipo 'timetable', dove per ogni parametro misurato viene aggiunto il tempo di campionamento.

La fase finale è stata l'unificazione di tutti i dataset per ottenere un'unica table con i relativi parametri, label e tempo di campionamento.

Il risultato ottenuto:

 7311x4 [table](#)

	1 Pdmp	2 Pin	3 Po	4 faul_code_1
1	617x1 time...	617x1 time...	617x1 time...	2
2	617x1 time...	617x1 time...	617x1 time...	1
3	617x1 time...	617x1 time...	617x1 time...	1
4	617x1 time...	617x1 time...	617x1 time...	11
5	617x1 time...	617x1 time...	617x1 time...	5
6	617x1 time...	617x1 time...	617x1 time...	5
7	617x1 time...	617x1 time...	617x1 time...	2
8	617x1 time...	617x1 time...	617x1 time...	6
9	617x1 time...	617x1 time...	617x1 time...	6
10	617x1 time...	617x1 time...	617x1 time...	10
11	617x1 time...	617x1 time...	617x1 time...	2
12	617x1 time...	617x1 time...	617x1 time...	5
13	617x1 time...	617x1 time...	617x1 time...	5
14	617x1 time...	617x1 time...	617x1 time...	9
15	617x1 time...	617x1 time...	617x1 time...	5
16	617x1 time...	617x1 time...	617x1 time...	5
17	617x1 time...	617x1 time...	617x1 time...	11
18	617x1 time...	617x1 time...	617x1 time...	6
19	617x1 time...	617x1 time...	617x1 time...	8
20	617x1 time...	617x1 time...	617x1 time...	11

dove per ogni cella della matrice sono presenti i parametri del relativo sensore e tempo di campionamento.

	Time	1 Data
1	0 sec	0.0927
2	2e-05 sec	0.0268
3	4e-05 sec	-0.0311
4	6e-05 sec	-0.0765
5	8e-05 sec	-0.1103
6	0.0001 sec	-0.1247
7	0.00012 sec	-0.1244
8	0.00014 sec	-0.1067
9	0.00016 sec	-0.0736
10	0.00018 sec	-0.0277

## 3.2 Estrapolazione delle Features

Questa fase si focalizza sulla generazione di parametri contenenti l'informazione diagnostica, i quali vengono utilizzati appunto nella fase successiva per l'allenamento del classificatore diagnostico.

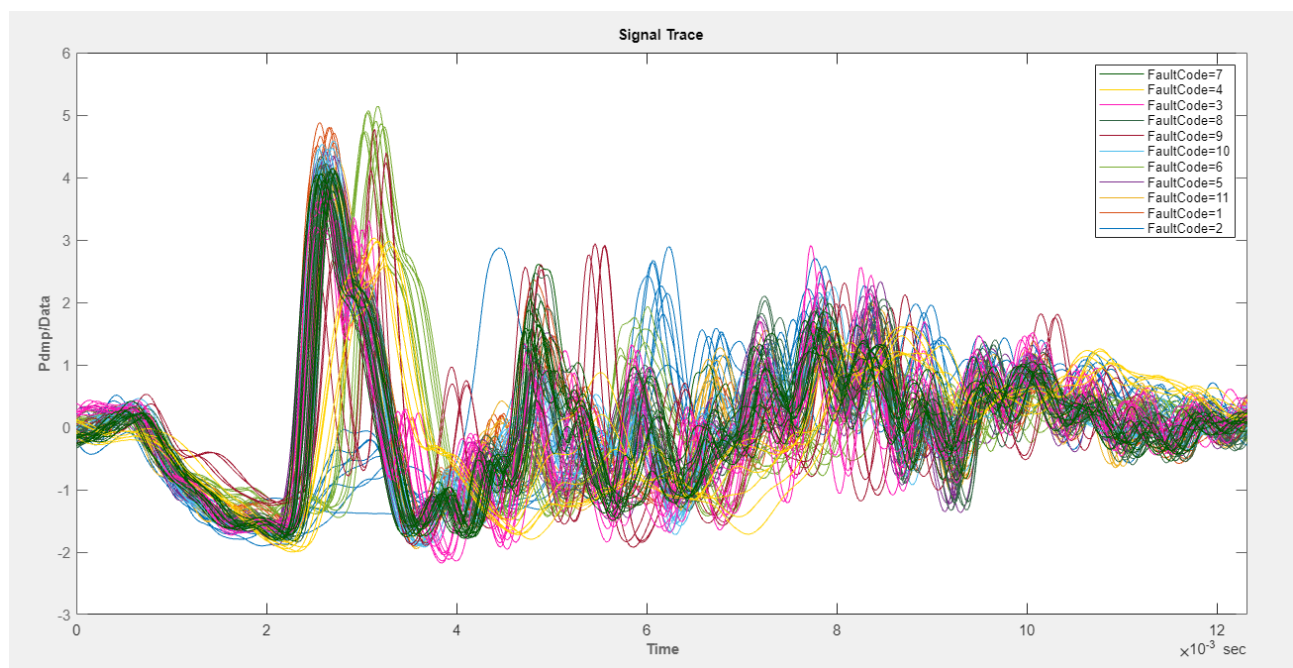
In questa fase è stata utilizzata un'app di MatLab : Diagnostic Feature Designer.

Questo tool ci permette di importare i dati diagnostici modificati precedentemente e di calcolare le relative features sia nel dominio del tempo sia nel dominio delle frequenze applicando inoltre, se necessario, un filtraggio nei due domini. Infine ci consente di classificare le features ed esportare quelle significative.

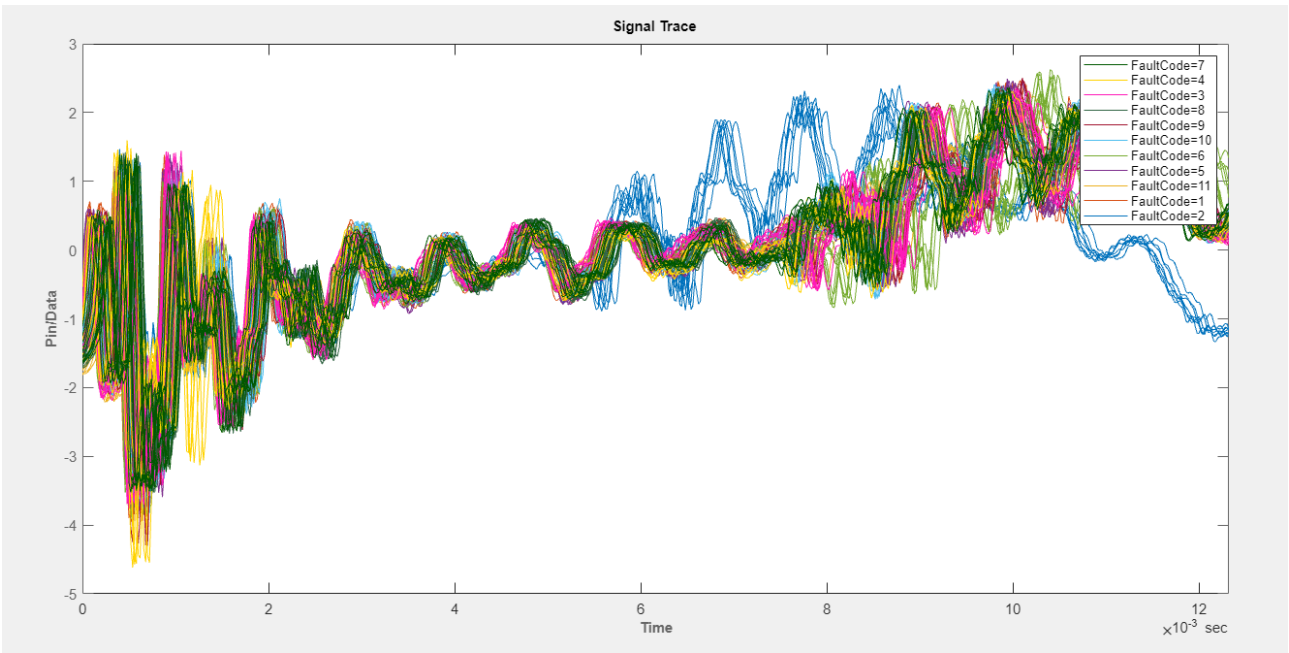
### 3.2.1 Calcolo delle features nel tempo

Per quanto riguarda l'andamento del segnale nel dominio del tempo, l'analisi è stata fatta su tutto il segnale senza dover applicare un filtraggio, diversamente dall'andamento nel dominio della frequenza.

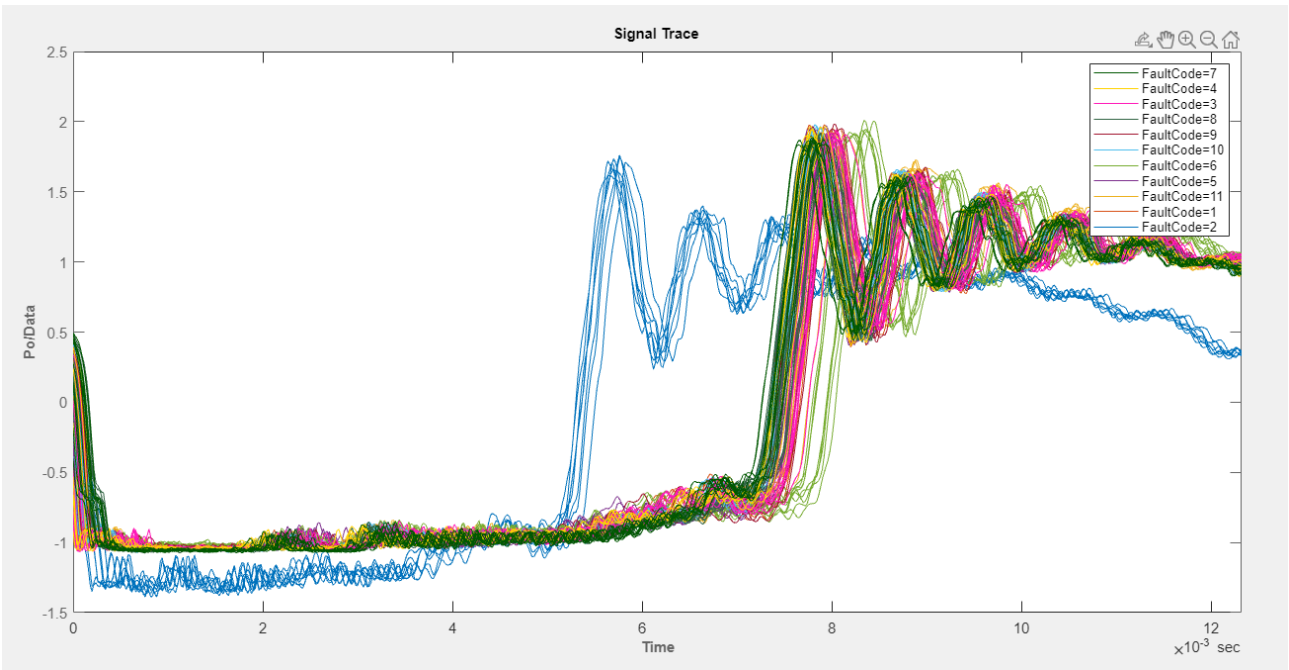
Andamento nel tempo del sensore Pdmp, con relativo raggruppamento dei guasti:



Andamento nel tempo del sensore Pin, con relativo raggruppamento dei guasti:



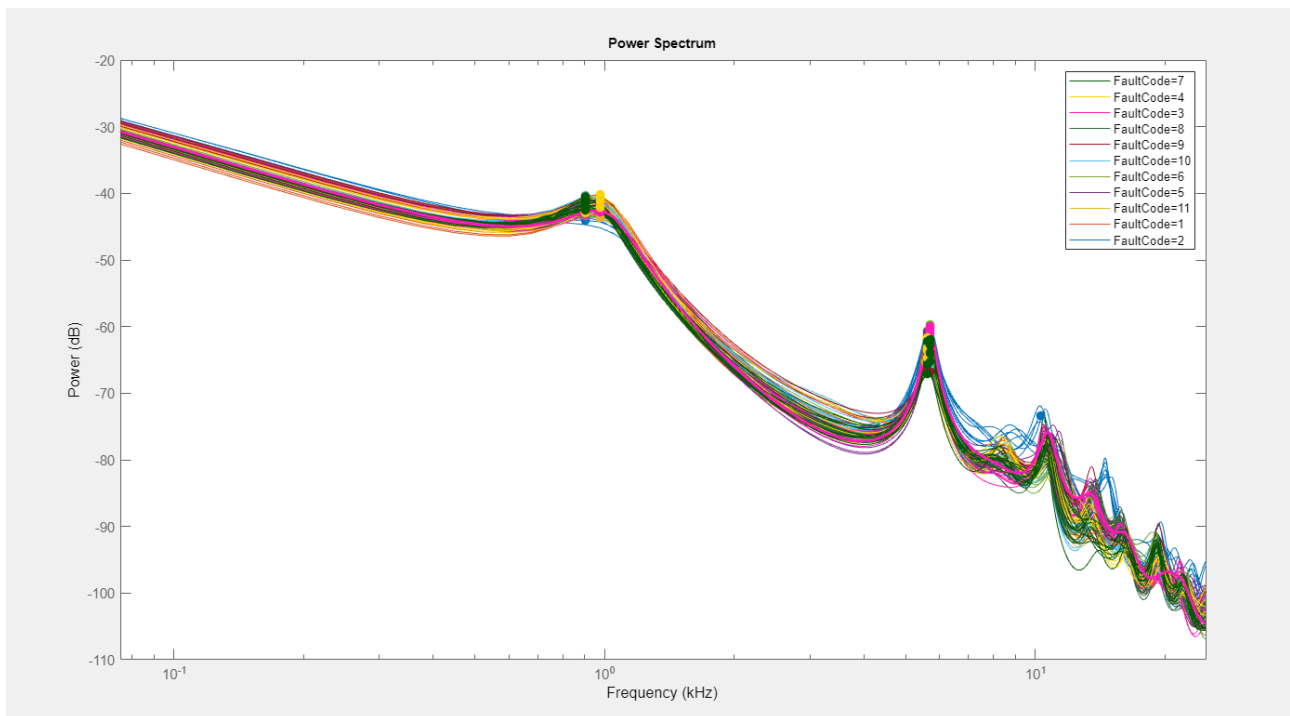
Andamento nel tempo del sensore Po , con relativo raggruppamento dei guasti:



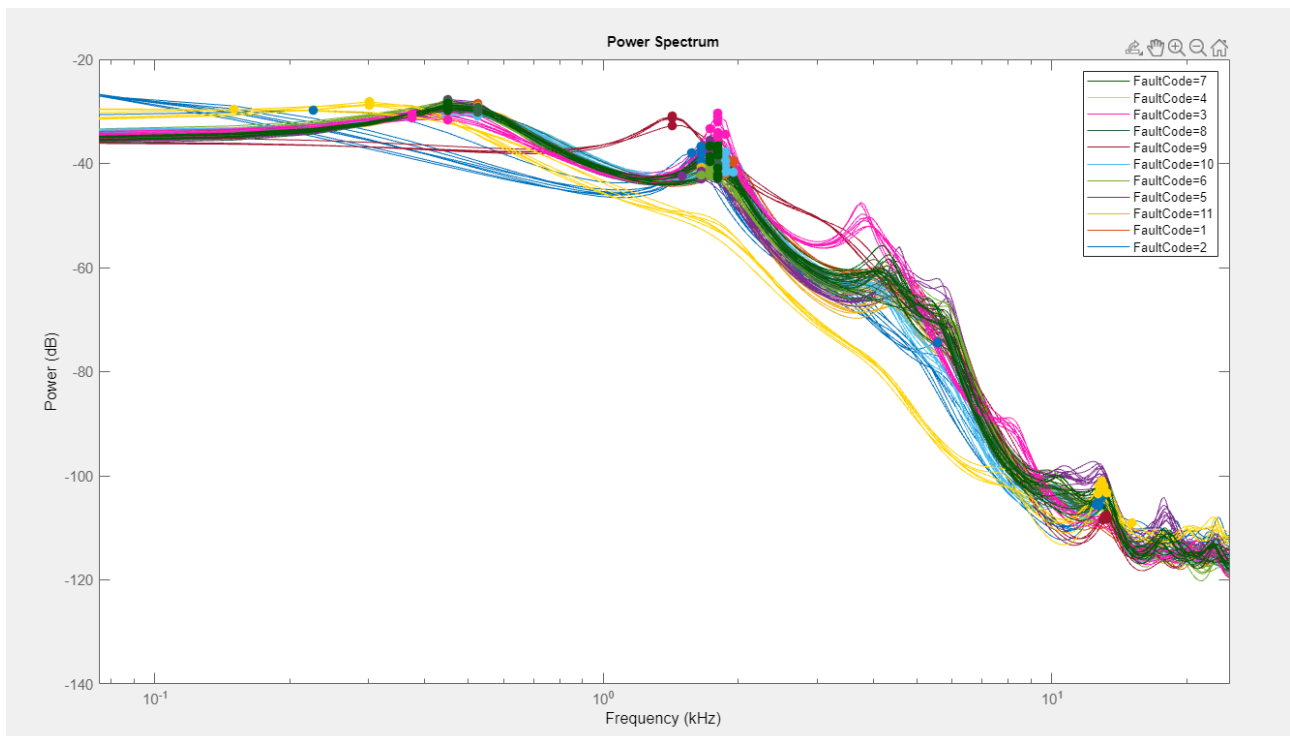
### 3.2.2 Calcolo delle features in frequenza

Per il calcolo delle features nel dominio in frequenza, è stato applicato un filtraggio per pulire il segnale attraverso un'auto regressive model, impostando una frequency grid da 0 a 25 khz ed un model order pari a 20.

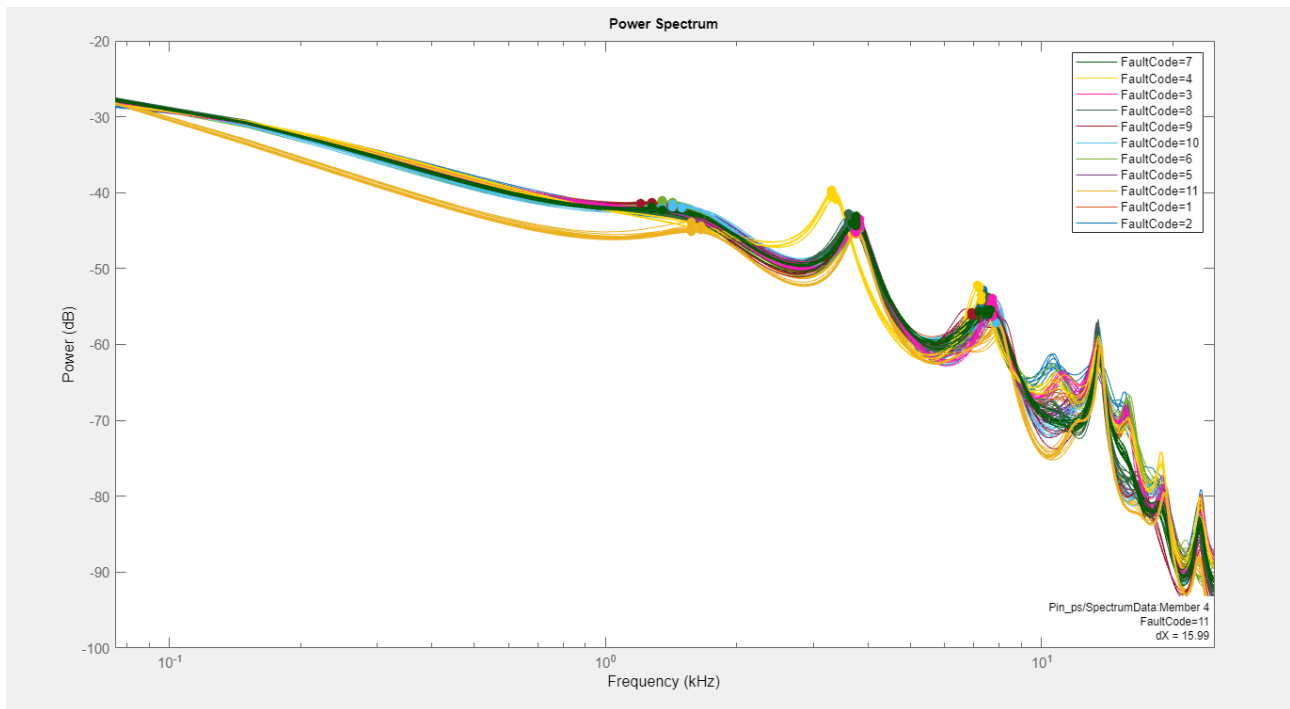
Andamento in frequenza del sensore Po , con relativo raggruppamento dei guasti:



Andamento in frequenza del sensore Pdmp , con relativo raggruppamento dei guasti:



Andamento in frequenza del sensore Pin , con relativo raggruppamento dei guasti:



Nei grafici è stato notato che sono presenti dei picchi più rilevanti di altri, come è possibile vedere nelle immagini precedenti: per questo motivo le features sono state calcolate in una finestra di frequenza compresa tra 0,5KHz e 25KHz prendendo solo i primi 3 picchi fondamentali.



### 3.2.3 Features

Dai passi precedenti abbiamo ottenuto per ogni sensore 10 features calcolate nel dominio del tempo:

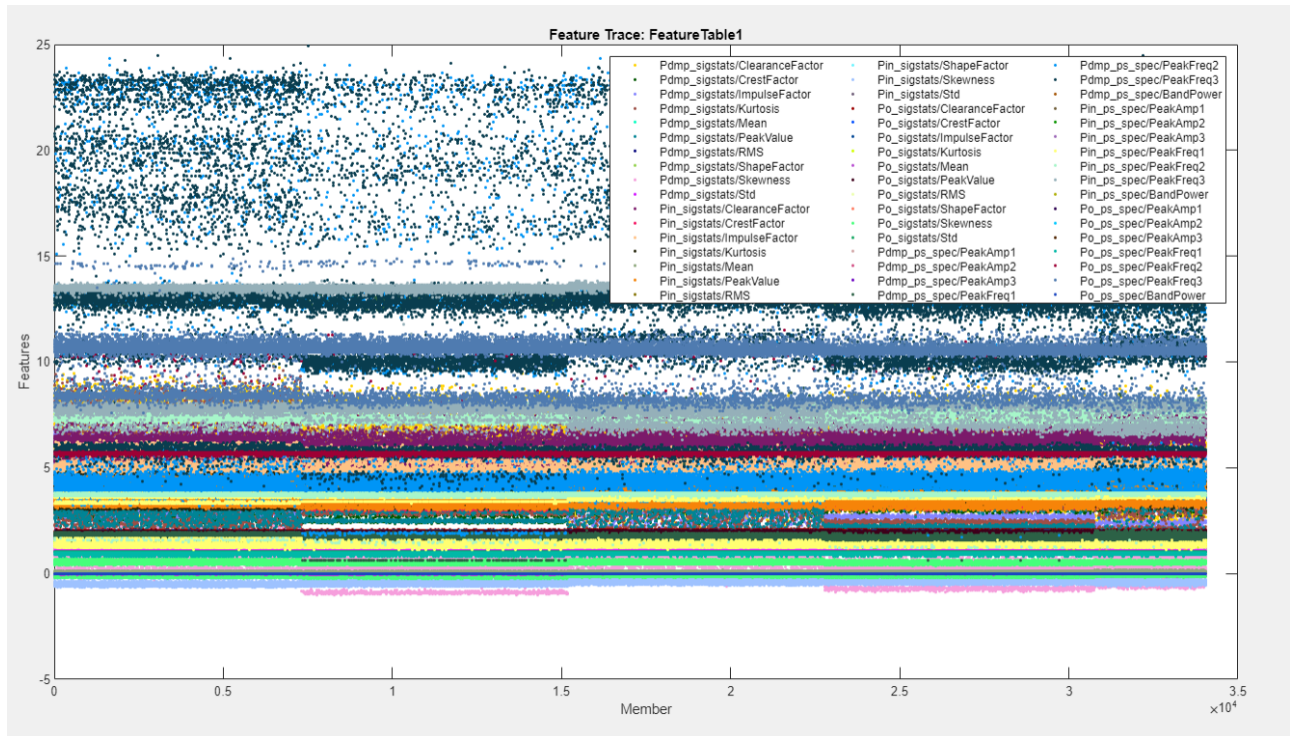
- a) Media
- b) Deviazione Standard
- c) Root Mean Square (RMS): la radice quadrata della media dei valori al quadrato
- d) Shape factor: RMS diviso per la media del valore assoluto
- e) Kurtosis: lunghezza delle code della distribuzione di un segnale
- f) Skewness: asimmetria della distribuzione di un segnale
- g) Peak value: valore assoluto massimo del segnale
- h) Impulse Factor: confronta l'altezza di un picco con il livello medio del segnale
- i) Crest Factor: valore di picco diviso per il valore RMS
- j) Clearance Factor: valore di picco diviso per il valore medio quadratico delle radici quadrate delle ampiezze assolute

Successivamente sono state estratte 7 features nel dominio delle frequenze settando il numero di picchi per i quali generare le features pari a 3:

- a) Band power: la potenza del segnale nella banda di frequenza selezionata
- b) Peak amplitude: una caratteristica basata sull'ampiezza dei picchi
- c) Peak frequency: una caratteristica basata sulla frequenza dei picchi.

Avendo settato il numero di picchi pari a 3, il peak amplitude ed il peak frequency sono stati calcolati per ogni singolo picco.

In totale sono state estrapolate 51 diagnostic features ricavate dai tre sensori al fine di addestrare un classificatore diagnostico.



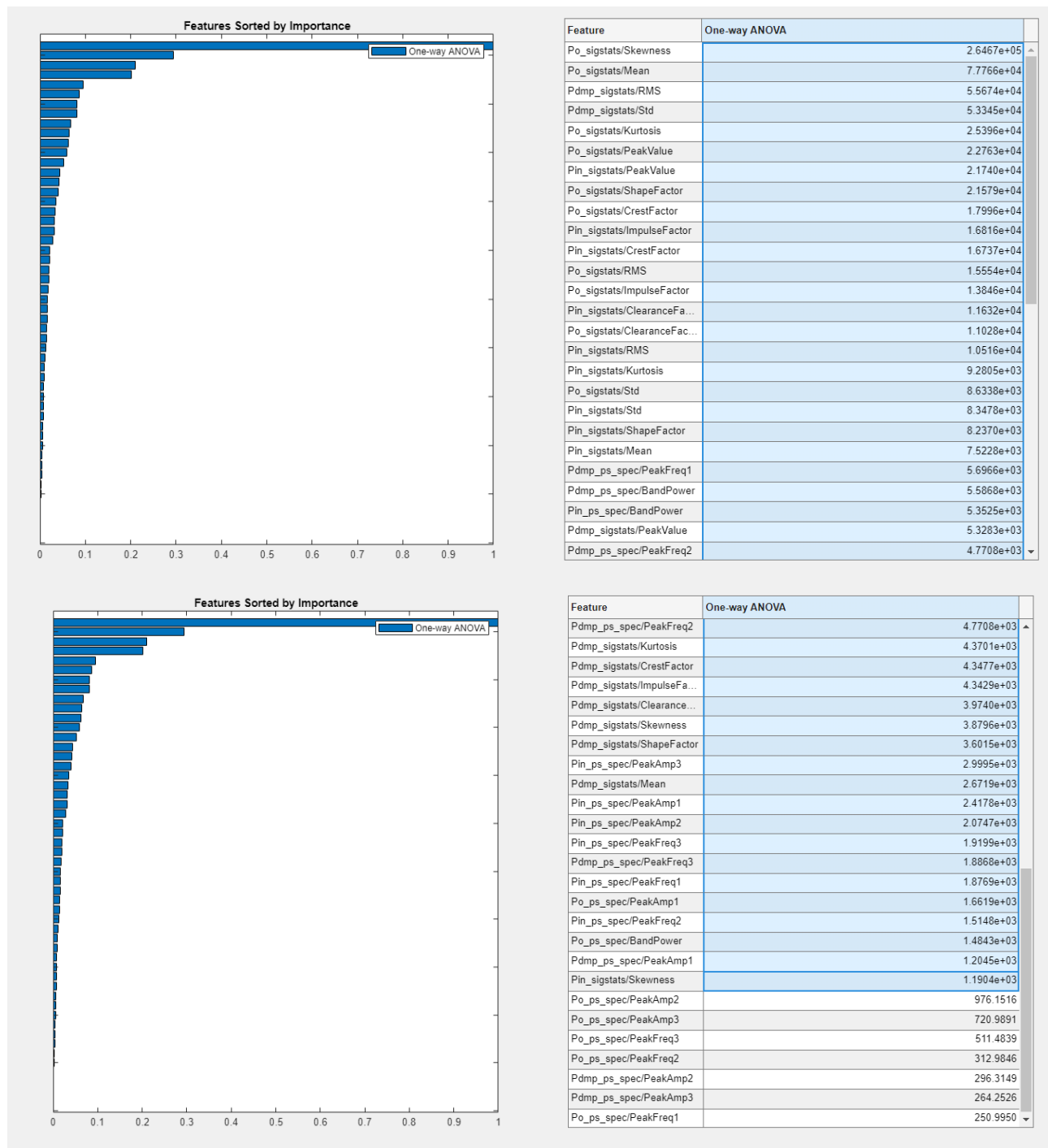
	FaultCode	Pdmp_sigstats/ClearanceFactor	Pdmp_sigstats/CrestFactor	Pdmp_sigstats/ImpulseFactor	Pdmp_sigstats/Kurtosis	Pdmp_sigstats/Mean	Pdmp_sigstats/PeakValue	Pdmp_sigs
1	2.0000	3.5766	2.6518	3.1554	2.3017	-0.0010	2.6686	
2	1.0000	7.5691	4.2290	6.0059	6.8003	-0.0170	4.5481	
3	1.0000	8.4477	4.4799	6.5869	7.9757	-0.0110	4.7995	
4	11.0000	7.0257	3.9864	5.5587	5.8020	-0.0162	4.2258	
5	5.0000	6.0030	3.5824	4.8277	4.5169	-0.0202	3.8124	
6	5.0000	6.2378	3.8031	5.1284	5.1353	-0.0215	4.0320	
7	2.0000	4.0225	2.8551	3.4623	2.6310	0.0012	2.8708	
8	6.0000	8.7444	4.5043	6.7157	8.1191	-0.0401	4.8974	
9	6.0000	7.5593	4.4228	6.1814	7.4208	-0.0248	4.8182	
10	10.0000	8.3750	4.3217	6.3409	6.7957	-0.0239	4.5850	
11	2.0000	3.5158	2.5650	3.0730	2.2558	0.0001	2.5795	
12	5.0000	7.2607	3.9627	5.6411	6.1334	-0.0275	4.2179	
13	5.0000	6.4048	3.7135	5.0815	4.7620	-0.0192	3.9704	
14	9.0000	7.0724	4.1075	5.6081	4.6860	-0.0271	4.3966	
15	5.0000	6.0392	3.7573	4.9777	4.7642	-0.0271	4.0093	
16	5.0000	6.0471	3.6400	4.8760	4.5008	-0.0226	3.8773	
17	11.0000	6.2493	3.7095	5.0136	4.7716	-0.0244	3.9230	
18	6.0000	8.1290	4.4548	6.4536	7.8058	-0.0148	4.8590	
19	8.0000	6.4052	3.8409	5.1843	4.8768	-0.0073	4.1063	
20	11.0000	6.4099	3.8629	5.1929	5.1435	-0.0204	4.0967	
21	6.0000	8.8769	4.6468	6.9048	8.8986	-0.0204	5.0468	
22	8.0000	5.9418	3.6109	4.8165	4.4191	-0.0097	3.8378	
23	3.0000	5.7918	3.6163	4.7855	4.4356	-0.0330	3.8884	
24	11.0000	6.7296	3.9258	5.3879	5.4790	-0.0165	4.2041	
25	4.0000	3.9555	2.8246	3.4412	2.8220	-0.0500	2.9834	
26	7.0000	6.4015	3.7348	5.0744	4.6342	-0.0061	3.9466	
27	11.0000	6.3697	3.7519	5.0800	4.7024	-0.0215	4.0342	

### 3.2.4 Ranking

Le 51 features precedentemente calcolate sono state classificate attraverso un metodo statistico dell'analisi della varianza: One way Anova.

Attraverso questo ranking sono state scelte 44 features in base ad una valutazione euristica: sono state selezionate quelle con ordine di grandezza paragonabile (in questo caso la grandezza è il risultato del test One way Anova).

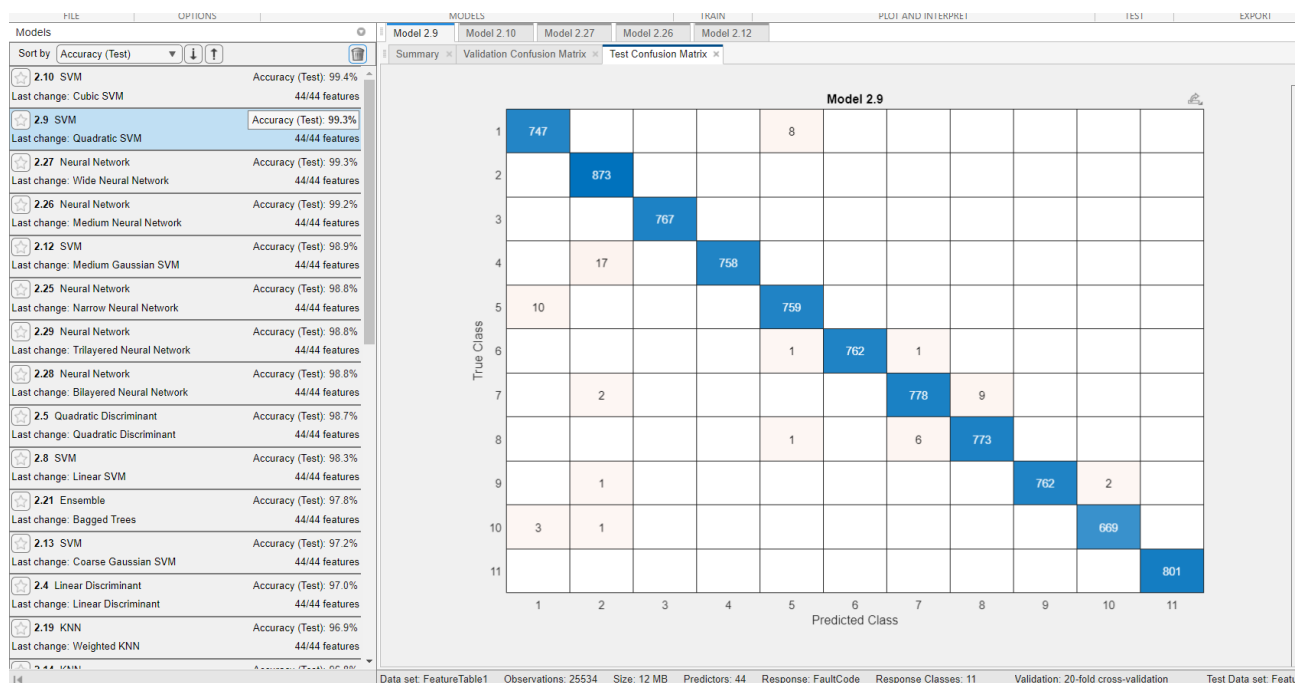
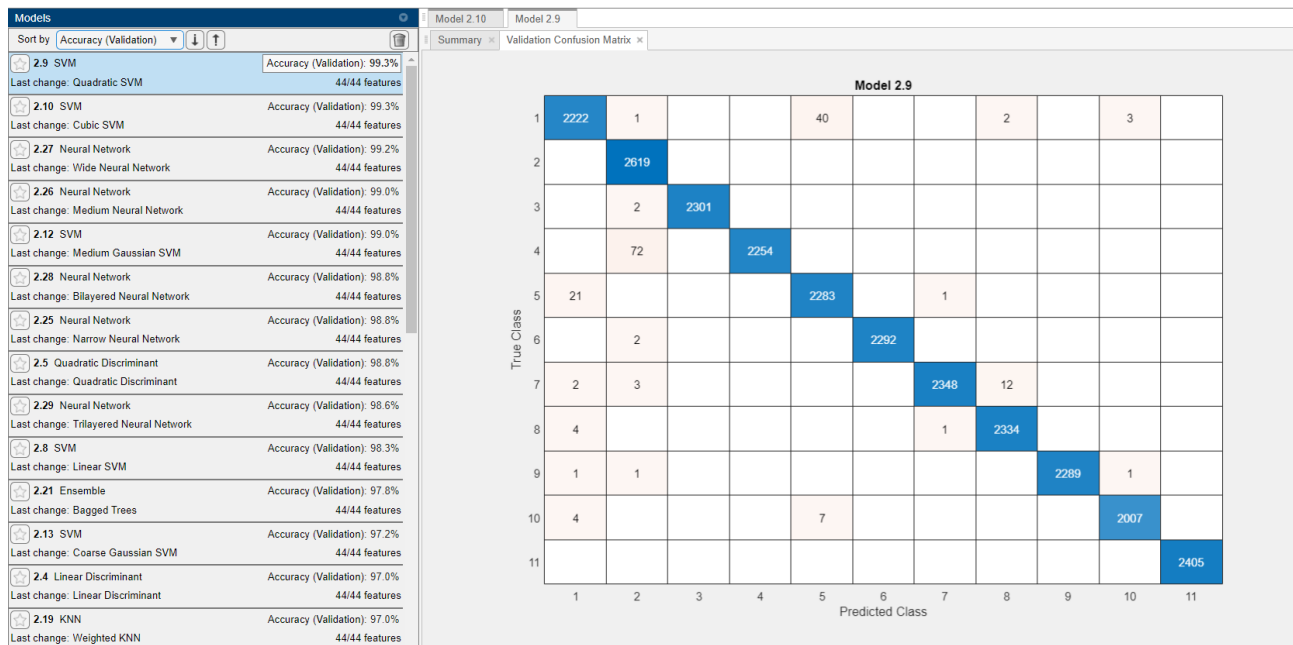
Effettuata questa scelta le feature sono state esportate per poter addestrare un classificatore diagnostico.



### 3.3 Addestramento classificatori

Attraverso le 44 features generate nello step precedente, è stato possibile allenare i classificatori. In questa fase è stata utilizzata un'ulteriore app a disposizione di MatLab: il Classification Learner. Esportando direttamente la table delle features dal Diagnostic Feature Designer al Classification Learner, è stato possibile allenare tutti i classificatori messi a disposizione dal tool di MatLab ed in un secondo momento sono stati valutati i migliori.

Poiché il testing dataset fornito per il progetto è sprovvisto di fault label, per una valutazione ottimale del training è stato diviso il dataset delle features in due parti: il 75% è stato utilizzato per il training dei classificatori ed il restante 25% è stato usato per un testing preliminare prima del vero testing sui testing dataset.



Abbiamo notato che ogni classificatore ha avuto una percentuale di Accuracy molto alta. Poiché le differenze percentuali tra i migliori addestratori sono molto marginali, si è optato per scegliere il classificatore suggerito dal Classification Learner, ossia SVM Quadratic con una validation accuracy del 99.3% ed una test accuracy del 99.3%.

### 3.4 Test del classificatore

Il dataset che è a disposizione ha le label dei guasti mancanti. Questo sta a significare che il test fatto può essere considerato approssimato non avendo avuto a disposizione la possibilità di fare un confronto tra la label predetta e quella reale. Nello specifico è stato esportato il modello del classificatore precedentemente allenato.

Una volta esportato gli è stato dato in input il dataset di test, ossia una table di features ricavate dai testing dataset seguendo gli stessi passi analizzati precedentemente. La features table per il test comprende le medesime features usate per il training del classificatore.

#### CARICAMENTO E MANIPOLAZIONE FILE CSV DI TESTING PER CREAZIONE DELLA TEST TABLE

```
pdp3=readmatrix("Data_Challenge_PHM2022_testing_data\data_pdp3.csv","Delimiter",";");
pin3=readmatrix("Data_Challenge_PHM2022_testing_data\data_pin3.csv","Delimiter",";");
po3=readmatrix("Data_Challenge_PHM2022_testing_data\data_po3.csv","Delimiter",";");

pdp3=rmmissing(pdp3,2);
pin3=rmmissing(pin3,2);
po3=rmmissing(po3,2);

pdp3(:,1) = [];
pin3(:,1) = [];
po3(:,1) = [];

pdp3_T = transpose(pdp3);
pin3_T = transpose(pin3);
po3_T = transpose(po3);
```

```
DATA3 = table();

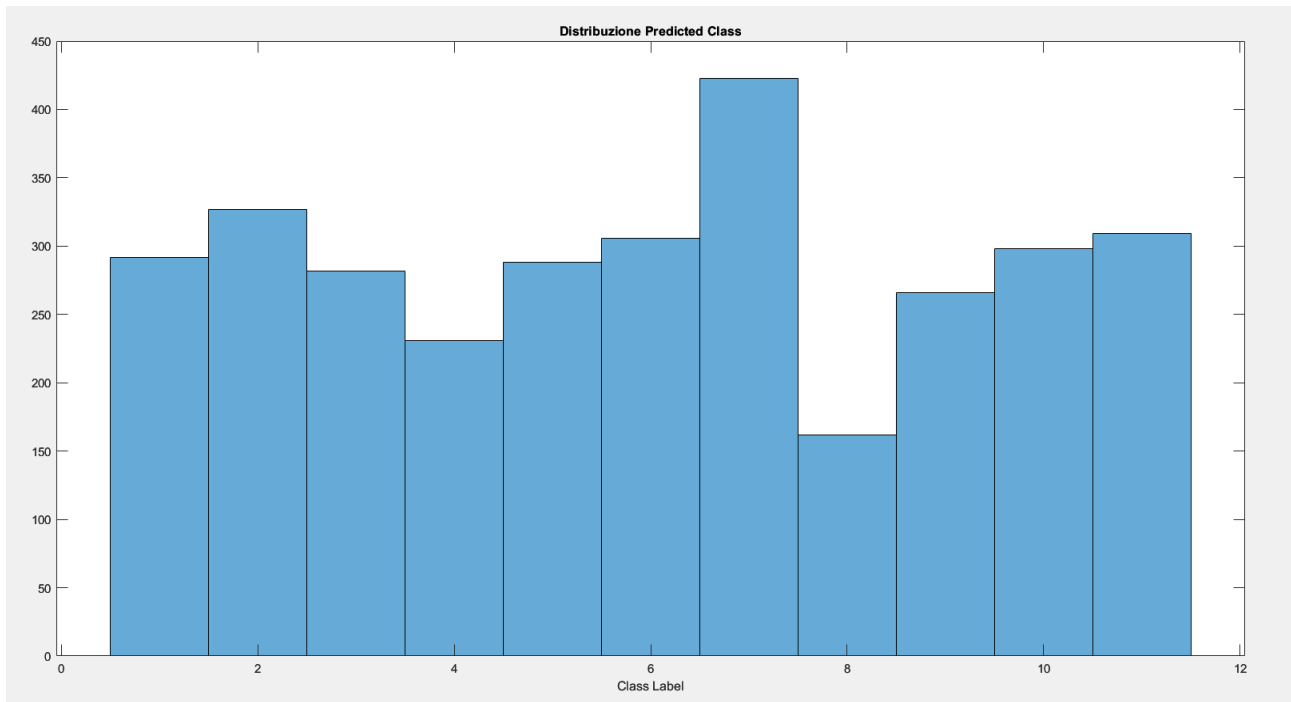
for i = 1:3184
    DATA3.Pdmp(i) = {array2timetable(pdp3_T(:,i),"VariableNames',{'Data'},'SampleRate',50000)};
    DATA3.Pin(i) = {array2timetable(pin3_T(:,i),"VariableNames',{'Data'},'SampleRate',50000)};
    DATA3.Po(i) = {array2timetable(po3_T(:,i),"VariableNames',{'Data'},'SampleRate',50000)};
end
```

#### TEST DEL MODELLO SUL DATASET REALE DI TESTING

```
yfit = TrainedModel.predictFcn(FeatureTable1);
```

#### CREAZIONE ISTOGRAMMA PER LA VALUTAZIONE DELLA DISTRIBUZIONE DEI RISULTATI

```
histogram(yfit,'EdgeColor','k')
title('Distribuzione Predicted Class')
xlabel('Class Label')
```



Non avendo le fault label, l'unica metrica per valutare i risultati di testing del classificatore allenato è la valutazione della distribuzione dei risultati di testing assicurandoci di ottenere un'omogeneità dei risultati tra le 11 classi di fault in quanto un probabile errore che avremmo potuto riscontrare nel classificatore è che esso avrebbe classificato solo alcune classi di fault ignorandone altre.



## 4. CONCLUSIONI

Ciò che si può evincere da tale progetto è la qualità nello sviluppare un classificatore ottimale, tale per cui è possibile automatizzare il processore di riconoscimento di un fault. La tecnologia dei classificatori multi-classe utilizzata ha avuto risultati ottimi dal punto di vista tecnico e di efficienza diagnosticando con percentuali alte la correttezza del tipo di guasto. I dati messi a disposizione per l'addestramento autonomo sono stati all'altezza delle aspettative per poter allenare un algoritmo di apprendimento automatico per riconoscere i vari pattern dei fault. Ciò che non è stato possibile effettuare e che potrebbe essere un future work è un test finale su un dataset di macchinari differenti e con relative label. Ciò consentirebbe di fare un'analisi più approfondita e di estrapolare un valore di accuracy di tale classificatore con un margine di errore pari quasi allo zero.