Collegio di Ingegneria Meccanica, Aerospaziale, dell'Autoveicolo e della Produzione

Corso di Laurea Magistrale in Ingegneria Meccanica

Tesi di Laurea Magistrale

Applicazione e validazione di un modello Random Forest per la stima della massa di particolato in motori Diesel

Relatore

Prof. Daniela Anna Misul

Correlatore

Dott. Ing. Claudio Maino

Candidato

Alessandro Falai

Anno accademico 2018/2019

Indice

Abstract

La sempre maggior stringente necessità di ridurre le emissioni inquinanti ha portato ad uno studio sempre più approfondito delle tecniche di trattamento dei gas combusti prodotti da motori ad accensione spontanea. In particolare, viene analizzato il principale componente delle comuni polveri sottili, il soot.

L'obiettivo del presente elaborato è quello di realizzare un modello, implementabile in futuro in ECU (electronic control unit), che stimi in modo accurato la quantità di particolato prodotta da un motore Diesel. Tale modello, scritto in ambiente Python, è realizzato mediante strumenti matematici di Machine Learning, nonché algoritmi di apprendimento supervisionato.

Il materiale iniziale è costituito dall'insieme dei parametri di funzionamento, i quali corrispondono agli input del modello, e dai valori misurati di particolato, rappresentanti invece gli output, in determinati punti di lavoro di due motori Diesel (un 2.0 litri ed un Cursor 11 litri). Nello specifico, abbiamo 1112 punti rpm(giri motore) x Pme(Pressione media effettiva indicante il carico del motore) nel primo caso e 5260 punti rpm x Pme nel secondo, nel quale i parametri di funzionamento sono suddivisi tra misure effettuate a banco e stimate in centralina.

Le molte simulazioni condotte su diversi set di training, percentuale del dataset disponibile per l'apprendimento dell'algoritmo, e di testing, rimanente parte del dataset sui cui testare e produrre i risultati in termini di accuratezza e errore quadratico medio, hanno portato all'ottimizzazione di un codice solido e strutturato, in grado di individuare quali parametri di input influiscono maggiormente sui valori di particolato prodotti e quali, anche in seguito a errori di misura, risultano irrilevanti. Ciò infatti permette di ridurre il problema a quei parametri

motoristici caratterizzanti la formazione di soot tramite il fit dei dati, svincolandosi completamente dalla fenomenologia del caso.

Introduzione

Al giorno d'oggi, circa la metà della domanda globale di petrolio proviene dal settore dei trasporti nel suo insieme ed è previsto che tale trend tenderà ad aumentare nei successivi vent'anni. L'immenso parco mondiale di veicoli terrestri, che montano motori a combustione interna (MCI), contribuisce, in parte, al deterioramento della qualità dell'aria che respiriamo in seguito all'emissione di sostanze inquinanti, le quali comportano un aumento generale dei problemi di salute e una maggiore incidenza di malattie cardiocircolatorie, patologie respiratorie e tumori.

Un inquinante è una specie chimica che ha un effetto diretto sulla salute umana (cancerogeno o velenoso) e se ne distinguono di due tipi: *primari*, generati direttamente dal processo di combustione, e *secondari*. Gli inquinanti primari principali possono suddividersi in: monossido di carbonio CO, idrocarburi incombusti HC, ossidi di azoto NO_x e particulate matter PM (particolato); mentre quelli secondari si formano in un secondo momento a partire dal primario che può essere sottoposto ad una serie di reazioni chimiche in atmosfera. Evidenze scientifiche in campo sanitario hanno dimostrato come le particelle sospese, una miscela di polveri di diversa dimensione, origine e composizione che, essendo molto piccole, tendono a rimanere sospese in aria e sono prodotte principalmente dai motori ad accensione per compressione, comunemente chiamati Diesel, siano tra le maggiori cause di problemi alla salute. Infatti, le continue restrizioni adottate in campo normativo, a livello nazionale e internazionale, che pongono severi limiti sia sulla massa che sulle dimensioni delle particelle, hanno favorito la ripresa di ricerche destinate all'approfondimento dei meccanismi di formazione ed ossidazione del particolato. Come conseguenza principale dell'incremento delle limitazioni emissive imposte dalle norme, nel corso degli anni sono

state adottate nuove strumentazioni, con particolare riferimento alle tecnologie di *after-treatment*.

Il presente lavoro di tesi nasce dalla volontà di trovare una valida alternativa all'utilizzo della strumentazione OBD (*On-board diagnostic*), la quale prevede un sensore fisico, detto *soot sensor*, per la misura della quantità emessa di soot (frazione carboniosa del particolato) a valle del DPF (*Diesel Particulate Filter*), continuando un lavoro svolto precedentemente da un collega. Mentre i limiti di emissioni stabiliti dalla legislazione sono definiti in relazione ai cicli guida specificati, le emissioni in uscita motore possono variare significativamente a causa del comportamento del guidatore e delle diverse condizioni di carico del motore stesso. Allora l'idea è quella di applicare e validare un modello virtuale che stimi in maniera accurata la massa di *engine-out soot* così da avere, in seguito a misure di emissioni *tailpipe* (a valle del trattamento dei gas combusti) tramite sensori fisici, un controllo diretto sulla rigenerazione del filtro per il particolato. Per tale scopo, infatti, il modello, di tipo predittivo e che si avvale di un algoritmo di Machine Learning, chiamato Random Forest, andrebbe ad analizzare i principali parametri motoristici elaborati in ECU (*Engine Control Unit*).

Capitolo 1 – Accenni teorici

Per comprendere il seguente lavoro di tesi, si ritiene necessario fornire al lettore una breve introduzione riguardo il modello di combustione nei motori ad accensione spontanea e i vari meccanismi di formazione degli inquinanti che scaturiscono da un processo di ossidazione non ideale, strumenti essenziali per cogliere quegli aspetti che sono alla base del lavoro sperimentale svolto.

1.1 Combustione nei motori ad accensione spontanea

Nei motori ad accensione spontanea (o per compressione) vengono utilizzati combustibili con ritardi di accensione relativamente brevi, ovvero ad *alta reattività* (gasolio, biodiesel) e infatti, diversamente da quelli ad accensione comandata, il combustibile non potrà essere premiscelato con l'aria comburente e compresso senza che questo dia luogo a reazioni di combustione, pertanto, per avere un buon controllo sul processo di combustione, viene iniettato ad elevate pressione (500 ÷ 2200 bar) e velocità (≈ 100 m/s), permettendo così una atomizzazione del getto in una nube di finissime goccioline (con diametro d ≈ 10μm), direttamente nel cilindro qualche istante prima del termine della fase di compressione, corrispondente al punto morto superiore (PMS). Il combustibile, a contatto con l'aria comburente ad alte densità (20 ÷ 30 kg/m^3) e temperatura(≈900 K), vaporizza formando una miscela che si autoaccende spontaneamente senza la necessità di un innesco esterno. In questi motori la miscela, grazie all'elevata reattività del combustibile, è in grado di accendersi anche con rapporti di aria e combustibile relativamente lontani dalle condizioni

stechiometriche[1] (carenza di ossigeno), dando luogo ad un processo di deidrogenazione della molecola che porta alla formazione di scheletri carboniosi i quali, addensandosi gli uni sugli altri, danno luogo a particelle solide carboniose responsabili delle emissioni di particolato e del fumo nero caratteristici dei motori Diesel.

L' ossidazione dei combustibili, quali benzina e gasolio, è rappresentata dalla seguente reazione tra reagenti e prodotti:

$$C_xH_y + \left(\frac{y}{4} + x\right)(O_2 + \Psi\, N_2) \rightarrow\ xCO_2 + \frac{y}{2}H_2O + \Psi\ \left(x + \frac{y}{4}\right)N_2 \qquad (1)$$

dove:

- Ψ rappresenta il numero di moli di N_2 presenti all'interno dell'aria rispetto all'ossigeno (solitamente circa 3,77)
- x indica il numero di atomi di carbonio presenti nella molecola di combustibile (circa 7)
- y indica il numero di atomi di idrogeno presenti nella molecola di combustibile (circa 13)

Questa reazione è esattamente una *combustione stechiometrica e completa* dalla quale, come si nota, si ottiene una produzione di anidride carbonica, acqua e azoto (che dovrebbe rimanere incolume al processo di combustione). Purtroppo, nella realtà, una reazione di ossidazione avviene attraverso una serie di step intermedi durante i quali non tutte le molecole di idrocarburo possono completare il processo di ossidazione, ritrovandoli così nei gas combusti e causando quelle emissioni inquinanti precedentemente citate. La CO_2 è un prodotto diretto del processo sopra descritto e non è, quindi, considerato un

[1] Per combustione stechiometrica si intende un processo di ossidazione in cui i reagenti (idrocarburi e aria) si trasformano completamente in prodotti di reazione (CO_2, H_2O e N_2) secondo i *coefficienti stechiometrici* rappresentanti determinati rapporti molari.

inquinante in quanto non ha effetto sulla salute umana, però ha un impatto diretto su scala globale essendo un gas serra e, perciò, uno delle principali cause del riscaldamento globale.

Osservando un generico andamento schematico della pressione all'interno di un cilindro e la curva di HRR (Heat Release Rate) in funzione dell'angolo di manovella in figura, nella

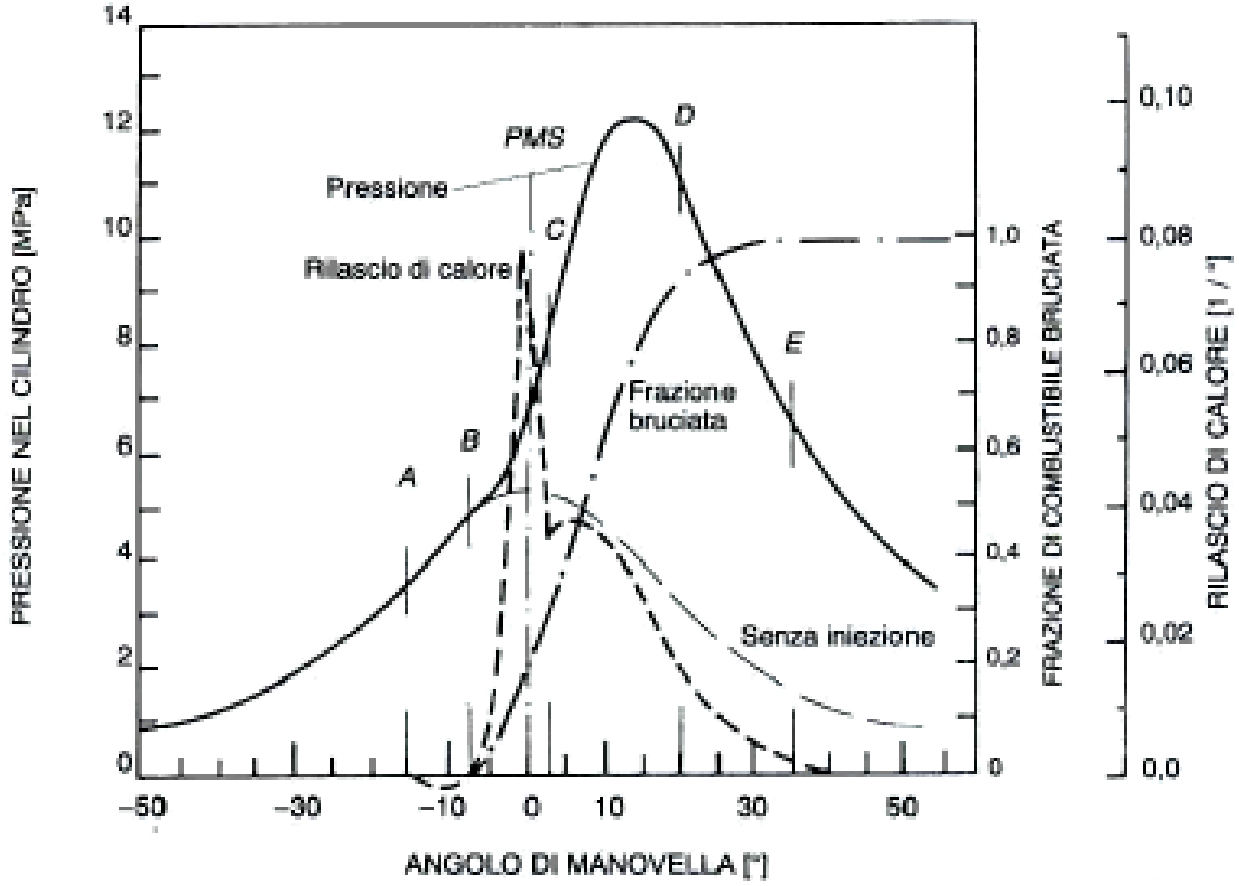

Figura 1.1 Andamento tipico in funzione dell'angolo di manovella della pressione nel cilindro di un motore ad accensione spontanea. Andamento della pressione con linea continua, frazione di combustibile bruciata con linea tratto-punto e rilascio di calore (*heat release rate*) con linea tratteggiata.

durata complessiva tra l'iniezione e la combustione, è possibile individuare quattro diverse fasi che caratterizzano la combustione [1]:

1. AB – *Ignition Delay*, ritardo di accensione
2. BC – *Premixed phase*, combustione in fase premiscelata

3. CD – *Mixing controlled phase*, combustione in fase diffusiva
4. DE – *Late combustion phase*, completamento della combustione

Andremo a vedere nel dettaglio le singole fasi individuando per ciascuna il contributo apportato al processo di combustione globale e sottolineando gli aspetti più importanti che lo caratterizzano.

Prima di ciò, si ritiene necessaria una breve considerazione riguardo alla quantità di combustibile che si accumula all'interno della camera di combustione del cilindro prima che il processo abbia inizio. Tale *accumulo* deve necessariamente essere tenuto sotto controllo, cioè mantenuto entro certi limiti, per smussare l'entità del picco di combustione premiscelata, intervenendo sull' injection rate (portata istantanea immessa in camera). Una diminuzione troppo accentuata di tale parametro potrebbe portare ad un degrado dell'efficienza del processo a causa dell'allungamento della combustione su un intervallo angolare di manovella maggiore e rischiando, quindi, di ritrovare parte del processo ossidativo in fase di espansione. Per questo inconveniente, le principali soluzioni risultano essere:

- *Injection rate shaping*: modulazione della portata iniettata mantenendola relativamente bassa nelle prime fasi dell'iniezione, in modo che si accumuli poco combustibile in camera, ed aumentandola nella fase successiva.
- *Iniezione pilota*: piccola quantità iniettata di combustibile (circa 5÷10% del totale) precedente l'iniezione principale. Studiando opportunamente la fase dell'iniezione pilota si può far si che questa prima quantità accumulata inizi a bruciare quando avviene l'avvio dell'iniezione principale, così da innalzare la temperatura localmente e accelerare il processo per il gasolio successivamente immesso in camera rendendo minore l'accumulo durante l'evento principale. Questo sistema permette di ridurre

notevolmente i picchi di combustione (di pressione),i quali sono caratteristici di quella che viene definita rumorosità di un motore Diesel.

1.1.1 Ignition delay

L'ignition delay, o ritardo di accensione, in un motore Diesel è definito come l'intervallo di tempo (nell'ordine dei ms) che intercorre tra l'inizio dell'iniezione (Start of Injection – SOI) e l'inizio della combustione (Start of Combustion – SOC) la quale non avviene in maniera simultanea per tutte le molecole di combustibile iniettate. Questo parametro riveste grande importanza non tanto per il fatto che sia di per sé nocivo ma in quanto responsabile della successiva fase di combustione a volume quasi costante. Quest'ultima e vantaggiosa ai fini del rendimento termico, ma dannosa dal punto di vista delle sollecitazioni e della rumorosità. Le cause fondamentali del ritardo possono essere suddivise in due macro-gruppi diversi per natura:

- Cause di natura fisica: fenomeni che modificano lo stato di aggregazione delle molecole di combustibile e permettono la miscelazione con l'aria;
- Cause di natura chimica: fenomeni di ossidazione che variano la struttura chimica del combustibile.

Ad ognuna delle cause citate è associato un intervallo di tempo, le quali sommate danno il tempo totale corrispondente al ritardo di accensione:

$$\tau = \tau_{fisico} + \tau_{chimico} \tag{2}$$

I due processi non sono mai rigorosamente in serie ma possono avere una certa sovrapposizione. Inoltre, l'aliquota fisica risulta essere predominante, essendo il combustibile ad elevata reattività, traducendosi in una alta velocità di reazione e, quindi, ridotto tempo dedicato alla chimica del fenomeno.

Entrando nel dettaglio della prima, vediamo che questa può essere suddivisa in sottogruppi che caratterizzano il cambiamento del getto di combustibile penetrante in camera di combustione:

- Atomizzazione (polverizzazione): il getto di liquido di combustibile deve trasformarsi in vapore così da potersi miscelare con l'aria. Tale mutazione avviene attraverso un processo di rottura della colonna liquida ed è governata principalmente dalle caratteristiche fisiche del combustibile, in particolare dalla tensione superficiale, e dall'intensità d'interazione aerodinamica con l'aria circostante, ad elevata densità, che tende a polverizzare il getto. Le forze aerodinamiche agiscono in un primo momento sulla superficie esterna del getto, provocandone una prima frantumazione (rottura o *break up primario*), e successivamente sulle gocce formatesi, le quali subiscono una progressiva trasformazione, perdono la loro forma sferica e di dividono in altre miriade di goccioline di dimensioni microscopiche (rottura o *break up secondario*).

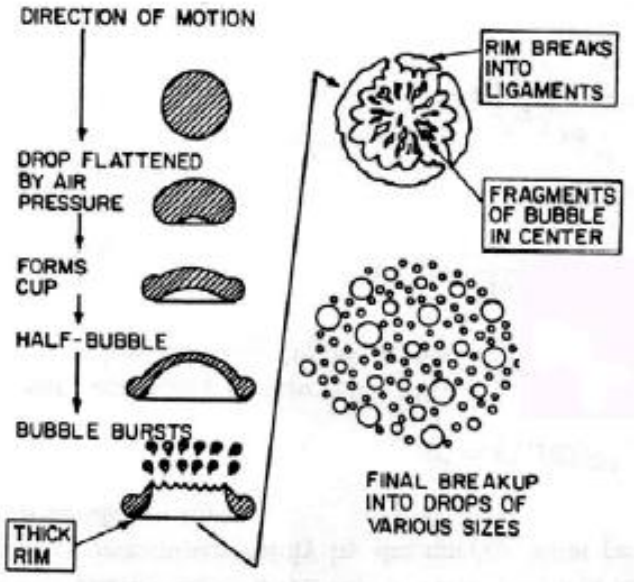

Figura 1.2 Processo di *break up* di una particella sferica che interagisce con l'aria ambiente

Questo processo è, quindi, controllato dalla contrapposizione delle forze aerodinamiche, che spingono verso l'atomizzazione del getto, e tensioni superficiali, che tendono mantenere le gocce coese. Il parametro che esprime l'opposizione dei due contributi è rappresentato dal rapporto di Weber:

$$We = \frac{\rho_g v^2 D}{\sigma}$$

Dove ρ_g rappresenta la densità del combustibile, v la velocità del liquido iniettato, D il diametro della goccia e σ la tensione superficiale che si oppone alla frantumazione. Viene preso il valore di $We = 12$ come limite al di sotto del quale non si ha polverizzazione della goccia.

- Evaporazione delle goccioline: all'avanzare della trasformazione, il combustibile all'interno della goccia si riscalda a causa del calore ricevuto dall'esterno, provocando contestualmente l'incremento della rapidità del processo di vaporizzazione. La conseguente riduzione delle dimensioni della goccia (evaporano gli strati superficiali) ed il suo rallentamento prodotto dalla resistenza aerodinamica causano una diminuzione del flusso di calore e un conseguente rallentamento del processo. Allora la rapidità del processo di evaporazione non è costante nel tempo ma raggiunge un picco per poi decrescere.
- Miscelamento dei vapori di combustibile con l'aria comburente: questa costituisce l'ultima fase dell'ignition delay, governa la successiva parte e dipende fortemente dal punto di funzionamento. Infatti, maggiore è la velocità di rotazione del motore e minori saranno i tempi a disposizione del processo e, così, maggiore dovrà essere la turbolenza in camera per accelerare il miscelamento tra vapori e aria e non avere cadute di rendimento.

Mentre i fenomeni chimici riguardano principalmente:

- reazioni di formazione di composti intermedi, quali radicali e perossidi: il combustibile ad alta reattività e capace di autoaccendersi con ritardi di ignizione contenuti è costituito da molecole lunghe e flessibili (alcani ad alto peso molecolare) che favoriscono la formazione di composti intermedi, quali perossidi e idroperossidi.
- fenomeni di *cracking* fortemente esotermici che precedono il processo di ossidazione.

1.1.2 Premixed Phase

Una volta terminato l'ignition delay dei nuclei iniettati (punto B, *Figura 1.1*), la combustione di questi causa un repentino innalzamento di temperatura e ciò permette una estrema accelerazione del processo ossidativo dell'intera quantità di combustibile accumulata che brucia quasi simultaneamente, cioè in maniera pressochè isocora, dando luogo ad una brusca impennata della pressione, come ben visibile dal tratto BC della figura. Tale incremento, seppur vantaggioso in termini di rendimento termodinamico, è responsabile, oltre della rumorosità precedentemente accennata, dell'insorgere di vibrazioni e sollecitazioni di entità importante, comportando strutture meccaniche più robuste che, traducendosi in elevate inerzie del motore, limitano i regimi di rotazione di un motore di questo tipo.

Durante tale fase si aggiunge un ulteriore inconveniente: il raggiungimento delle alte temperature finali crea condizioni idonee alla formazione degli ossidi di azoto (meglio noti

come NO_x). Infatti, la loro creazione non avviene durante la combustione premiscelata perché si hanno ancora condizioni di miscela ricca ($\lambda^2 = 0.25 \div 0.5$).

1.1.3 Mixing controlled Phase

Esaurita la rapida combustione dell'accumulo, il processo è regolato dalla velocità con cui le nuove frazioni di combustibile sono disponibili a bruciare, cioè dalla rapidità con cui evaporano, si miscelano con l'aria comburente ed ossidano. Solitamente in questa fase si raggiunge la massima pressione del ciclo (tratto CD della *figura 1.1*), e la rapidità con cui viene rilasciata l'energia può essere controllata agendo sull' injection rate. All'avanzare del processo, il nuovo combustibile iniettato trova sempre meno ossigeno disponibile per reagire a causa del progressivo aumento dei gas combusti all'interno della camera di combustione. In seguito a processi di deidrogenazione, condensazione e pirolisi, possono formarsi nuclei carboniosi incombusti, ovvero *soot*, costituiti da particelle solide contenenti un elevato numero di atomi di C (con H/C ≈ 0.1). Questa fase corrisponde alla *combustione diffusiva*, in cui viene bruciata circa il 90% di combustibile totale.

1.1.4 Late Combustion Phase

L'iniezione è ormai terminata, ma le reazioni chimiche procedono ancora esaurendosi in modo graduale. La combustione può coinvolgere i nuclei carboniosi che si sono formati nella

[2] Detta Dosatura Relativa definita come il rapporto tra la dosatura di lavoro α e quella stechiometrica α_{st} (essendo $\alpha = \frac{m_{air}}{m_{fuel}}$). Per $\lambda < 1$ siamo in condizioni di miscela ricca, per $\lambda > 1$ in condizioni di miscela povera e $\lambda = 1$ in condizioni stechiometriche.

fase precedente, permettendone l'ossidazione. Quest'ultima parte viene alimentata e promossa dai moti turbolenti che rimescolano i gas all'interno della camera, ma è necessario che ciò non si prolunghi in modo eccessivo così da non ridurre il rendimento del motore (per averlo alto occorre concentrare quanto più possibile il rilascio di calore nell'intorno del PMS).

Terminata la breve descrizione del processo di combustione, si procede analizzando le emissioni inquinanti prodotte dai motori Diesel, in particolare modo concentriamo l'attenzione su ciò che concerne cause e formazione del *soot*.

1.2 Emissioni di un motore Diesel

Il traffico stradale è sempre stato la principale causa di emissione di ossidi di azoto e fonte rilevante di polveri fini, nonché particolato, prodotti dai motori Diesel. Malgrado il netto incremento del traffico, nel corso degli anni le emissioni sono diminuite grazie alle innovazioni tecnologiche, necessarie per le sempre più stringenti limitazioni imposte dalla direttiva CEE europea, in ambito automobilistico. Nel 2016 le percentuali di emissioni provenienti dal traffico stradale rispetto a quelle complessive erano del:

- 50% circa per gli ossidi di azoto (NO_x)
- 12% circa per gli idrocarburi (HC)
- 20% circa per le polveri fini ($PM10$)

Tali valori risultano essere senza dubbio allarmanti, viste le cause che hanno sulla salute umana. Possiamo riportare alcuni dati sugli inquinanti principali, confrontandoli tra motori ad

accensione comandata e spontanea, in riferimento alle due categorie Euro 4 ed Euro 5[3] (Fonte dati: Arpa Lombardia, Inemar 2010).

COMBUSTIBILE	CATEGORIA EURO	PM2.5 (MG/KM)	PM10 (MG/KM)	NO_X (MG/KM)	CO (MG/KM)
BENZINA	Euro 4	15	27	55	299
	Euro 5	15	27	41	291
DIESEL	Euro 4	51	63	607	112
	Euro 5	16	28	437	94

Tabella 1 Confronto tra le emissioni inquinanti principali di un motore ad accensione comandata ed uno ad accensione spontanea. Sono stati presi in considerazione due standard europei (Euro 4 ed Euro 5)

In seguito a studi condotti sui motori Diesel, è stato visto come la presenza di idrocarburi incombusti (HC) e ossidi di azoto (CO) sia molto meno rilevante rispetto a quella di NO_X e *particulate matter (PM).*

1.2.1 Meccanismi di formazione degli inquinanti

Vecchi modelli per la descrizione del fenomeno di combustione erano basati esclusivamente sul segnale di pressione e di rilascio del calore e venivano avanzate ipotesi e supposizioni sul comportamento del getto di combustibile in camera. Grazie all'avvento di strumentazioni per l'analisi ottica si sono diffusi modelli più recenti basati su evidenze scientifiche. Il

[3] Gli standard europei sulle emissioni inquinanti sono una serie di limitazioni imposte sulle emissioni dei veicoli venduti degli Stati membri dell'Unione europea. Il periodo di immatricolazione dell'Euro 4 è dal 2005 al 2010, mentre quello dell'Euro 5 parte dal 2010.

processo di combustione tradizionale in un motore ad accensione per compressione può essere descritto dal modello concettuale di Dec, il quale ha condotto una sperimentazione su un motore ad accesso ottico [2]. Per capire la formazione delle particelle inquinanti, diventa fondamentale analizzare il comportamento e le trasformazioni del getto di combustibile tramite l'iniettore multi-foro. La penetrazione in fase liquida all'interno della camera è limitata dall'evaporazione del getto stesso, la quale tende ad avvenire rapidamente in seguito alle condizioni di alte pressione e temperatura. La pressione di iniezione viene mantenuta il più possibile costante. Seguendo l'evoluzione di uno dei getti, è possibile osservare ciò che succede dalla seguente *Figura 1.3* [3].

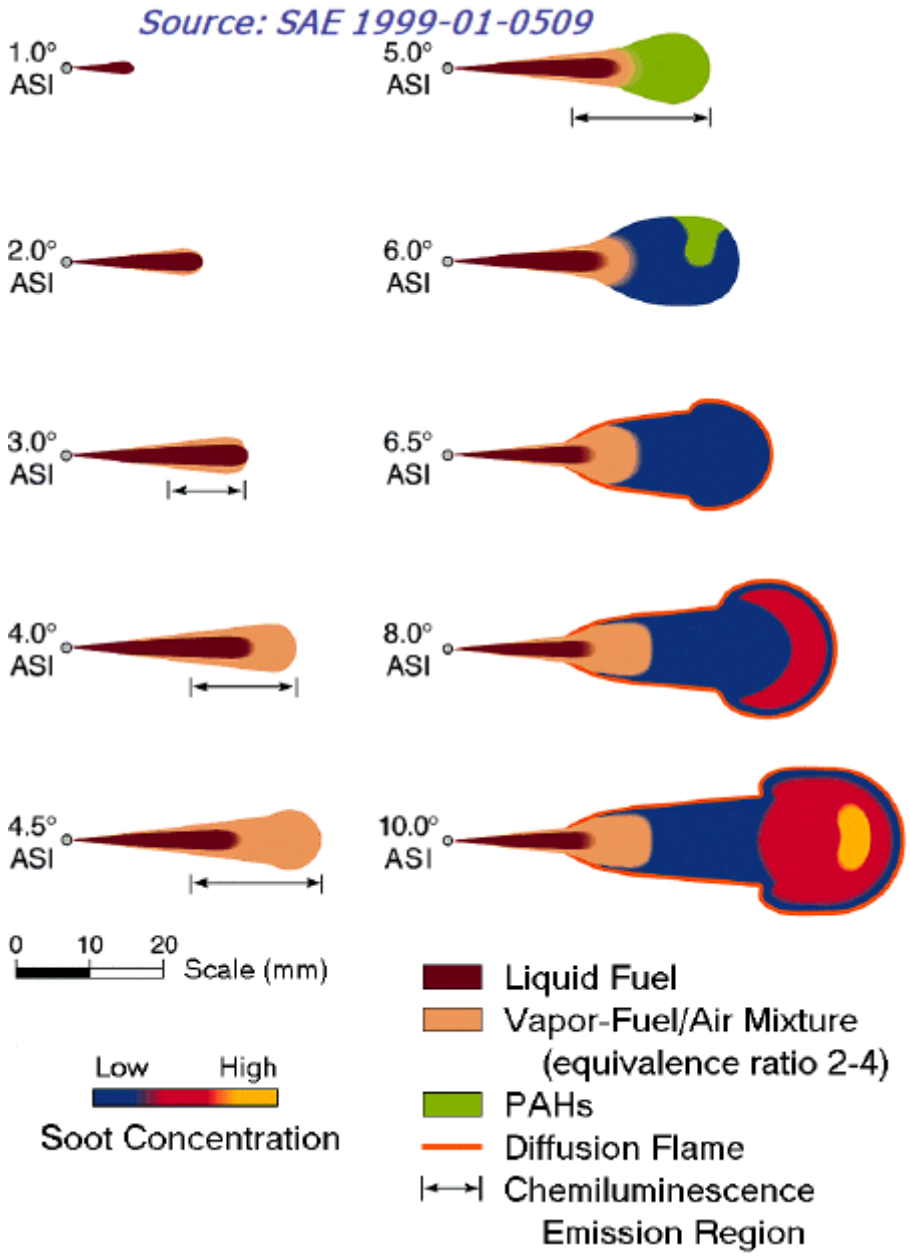

Figura 1.3 Sviluppo di un getto che entra in camera di combustione da 1° ASI fino a 10°ASI. Rappresentazione delle zone di formazione degli inquinanti all'aumentare dell'angolo di manovella.

Tutte le rappresentazioni sono classificate in base ad un angolo ASI (after start of injection), a partire dall'avvio idraulico dell'iniezione. Alcuni iniettori odierni sono elettromeccanici, ad azionamento indiretto: tra il comando elettrico e il momento in cui si apre l'iniettore c'è un ritardo. La prima fase corrisponde alla penetrazione del getto all'interno della camera fino

ad arrivare al 3° ASI in cui questa si ferma e diventa successivamente più piccola, contestualmente alla formazione di vapori di combustibile intorno al getto stesso. Il fenomeno di evaporazione determina la zona in cui non abbiamo più penetrazione del liquido e dipende dalle condizioni termodinamiche presenti in camera. Dopo appena 5° appare della chemiluminescenza: in corrispondenza della creazione della zona di miscela, comincia a provenire una debole luminosità dovuta non alla combustione, ma alle reazioni iniziali del combustibile con l'ossigeno formando alcuni radicali, senza un importante rilascio di energia e aumento di pressione. In questa fase si ha una combustione ricca e, in seguito alla discesa al di sotto del punto di infiammabilità, si ritrovano frammenti di combustibile: le molecole di combustibile non subiscono ossidazione ma rottura, per cui il legame C-C si modifica dando origine alla formazione di *idrocarburi policiclici aromatici* (PAH), i quali corrispondono ai precursori del *soot* (verde in figura). Col proseguire con il processo, i PAH si accrescono diventando particelle più o meno grandi: meccanismo di accrescimento (passaggio dal verde al blu). Successivamente si può osservare la fiamma diffusiva caratterizzata da un sostanzioso rilascio di energia chimica e un valore di dosatura circa unitario. Procedendo con l'iniezione, lo spray continua ad ingrandirsi ed aumenta la temperatura, continuando a creare condizioni adatte alla formazione di particelle carboniose, le quali risultano essere più numerose sulla punta del getto grazie al maggior tempo a disposizione e alle condizioni termodinamiche più elevate. Se la fiamma diffusiva va ad intersecare il *bowl* (pozzetto), ricavato sulla testa dello stantuffo, si spegne per lo scambio termico e per la carenza di ossigeno, provocando la mancata ossidazione delle particelle carboniose che ritroveremo allo scarico, ma solo in parte dopo aver partecipato al

soot burnout[4]. Considerando, quindi, l'intero sviluppo di un getto durante tutte le fasi della combustione, su può rappresentare la seguente *Figura 1.4* [4], rappresentativa di tutte le zone di formazione degli inquinanti.

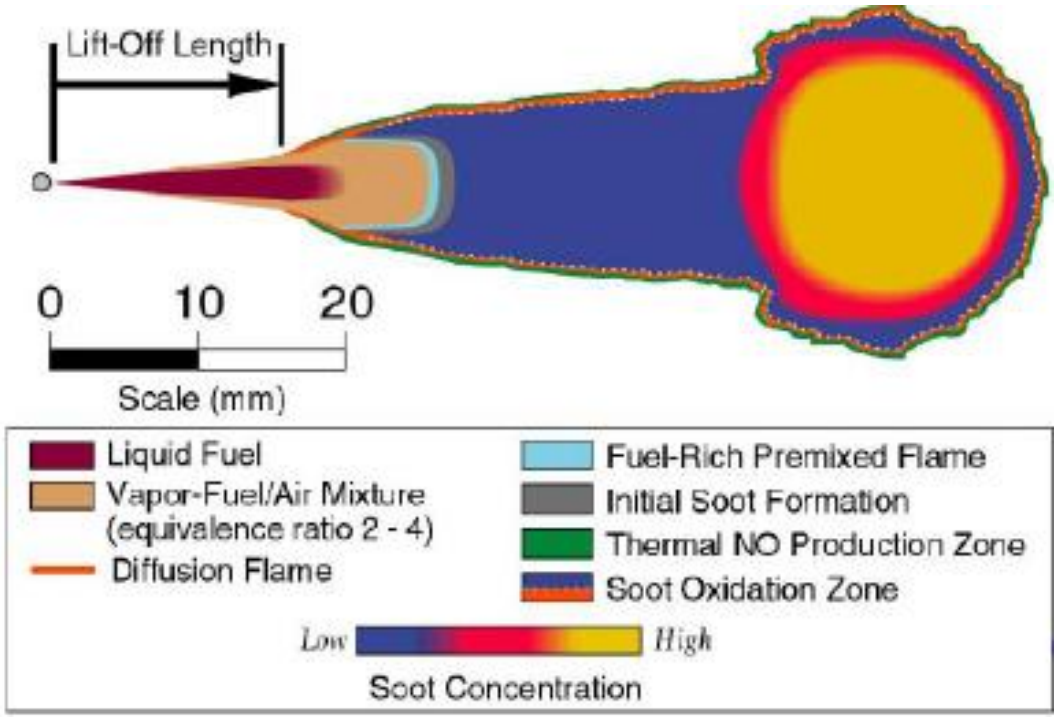

Figura 1.4 Rappresentazione finale delle zone di formazione degli inquinanti, durante tutte le fasi della combustione di un motore Diesel. (Lift-Off Length da *[5]*)

Come si può vedere, gli ossidi di azoto NO_x si formano in condizioni di alte temperature e, diversamente dal soot, in presenza di ossigeno. Infatti, la zona della fiamma diffusiva è quella migliore per la loro formazione.

[4] Le particelle carboniose, che non sono ossidate nel normale processo di combustione, possono incontrare una sostanziale presenza di ossigeno (dato che i motori Diesel lavorano con miscele povere) provocando, così, l'ossidazione del *soot*.

Ricapitolando, il soot inizia a formarsi appena è stata superata la combustione premiscelata (prodotti di combustione ricca come frammenti di combustibile, HC e PAH) e con temperature di circa 1600 K. Una volta arrivati a livello della fiamma diffusiva, aumentano le temperature (fino a 2700 K) e la concentrazione di ossigeno, grazie alla sua diffusione dall'esterno, così da permettere l'ossidazione del soot e di eventuali idrocarburi incombusti. Tali condizioni, però, facilitano la formazione di NO_x. Il giusto compromesso tra le due sostanze inquinanti, determinando un trade-off soot/NO_x, è descritto dal diagramma di Kamimoto-Bae, rappresentante una mappa di ϕ (l'inverso della dosatura relativa λ) in funzione della Temperatura, che esplica le zone delle due formazioni [6].

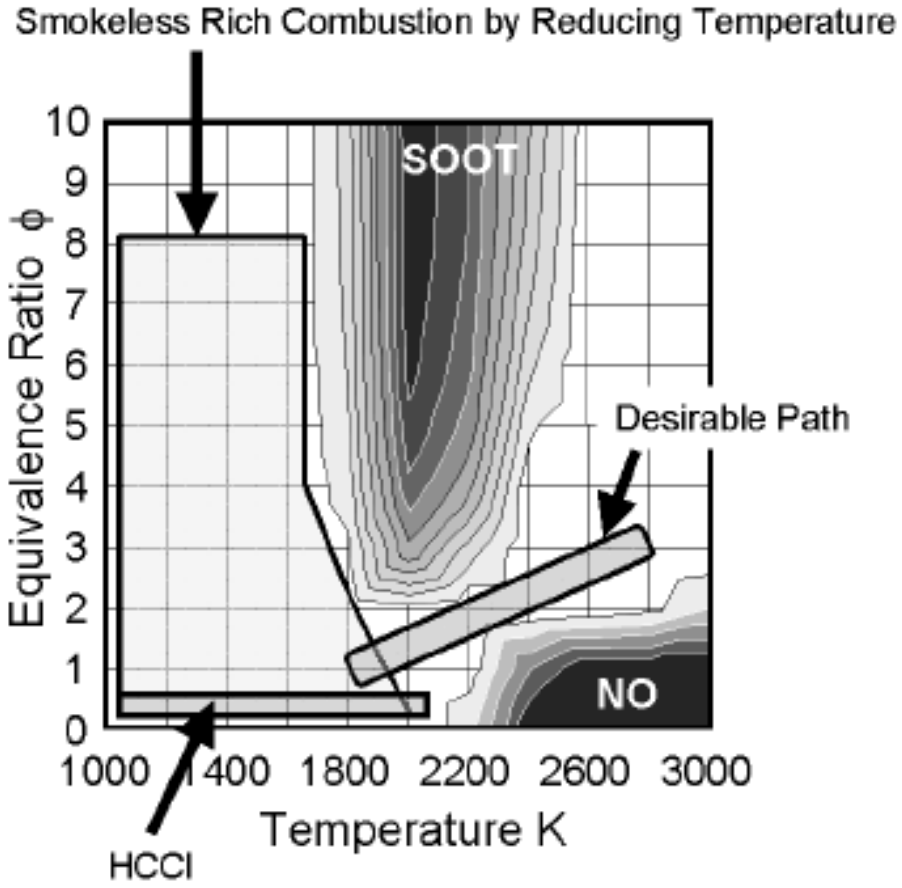

Figura 1.5 Diagramma di kamimoto-Bae rappresentante il trade-off tra soot e No_x. Sono visibili le varie zone: combustione ricca, campo dell'HCCI (Homogeneous Charge Compression Ignition), formazione soot, formazione NO_x, percorso ottimale.

La zona con combustioni di nuova generazione HCCI, ovvero con compressione di una miscela omogenea, è un obiettivo in quanto mostra l'assenza di entrambe le sostante inquinanti, ma riusciamo ad avvicinarci solo per carichi medio-bassi (condizione poco presente durante l'utilizzo di un motore a combustione).

1.2.2 Formazione degli NO_x

Nei motori ad accensione spontanea, nonostante si parli sempre di NO_x, gli ossidi di azoto sono formati per la maggior parte da NO (70÷80%) e per la restante da NO_2 [1]. Il meccanismo principale di formazione è quello termico, il quale avviene all'interno dei prodotti della combustione, e costituito dal modello di Zeldovich, secondo cui la formazione del monossido di azoto è descritto dalle seguenti reazioni:

$$N_2 + O \leftrightarrow NO + N \qquad (3)$$

$$N + O_2 \leftrightarrow NO + O \qquad (4)$$

$$N + OH \leftrightarrow NO + H \qquad (5)$$

La prima reazione è quella che richiede la più alta energia di attivazione, disponibile grazie alle alte temperature e presenza di ossigeno. Rispetto al processo di combustione, la formazione degli ossidi di azoto è lenta, ma con tempi compatibili con il ciclo di lavoro del motore. Questa reazione avviene nei prodotti della combustione e non nella fiamma diffusiva, rimanendo ad alta temperatura per molto tempo da permettere alle reazioni di avvenire. Il processo è fortemente influenzato dalla cinetica chimica, ma ciò esula dal

presente lavoro di tesi. Per quanto riguarda, invece, la formazione degli NO_2, si riferimento alle reazioni [3] :

$$NO + HO_2 \leftrightarrow NO_2 + OH \qquad (6)$$

$$NO_2 + O \leftrightarrow NO + O_2 \qquad (7)$$

Nei motori Diesel la prima reazione avviene sempre, mentre la seconda è facile che si interrompa a causa delle temperature più basse nei prodotti di combustione rispetto ai motori ad accensione comandata, quindi la riconversione in monossido di carbonio si arresta, permettendo la presenza di biossido di azoto allo scarico.

1.2.3 Formazione di PM

Con *particulate matter,* o particolato, si vuole indicare l'insieme delle particelle che vengono generate in fase di combustione e successivamente portate in sospensione dai gas di scarico, e formato da:

- una parte solida (SOL): costituita da elementi carboniosi e ceneri, è la componente principale e chiamata *soot*. Durante il processo di combustione, i precursori del particolato generano queste particelle carboniose che crescono e si aggregano. Inizialmente hanno strutture esagonali (dell'ordine dei nm) che possono impilarsi l'una sull'altra a dare strutture lamellari che talvolta si aggregano in particelle più o meno sferiche.

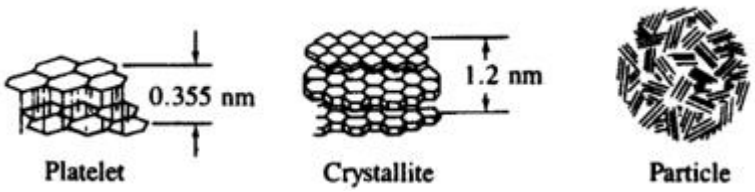

Figura 1.6 Fasi della struttura molecolare selle particelle carboniose

Tali particelle non sono compatte, ma presentano interstizi e spazi non trascurabili. Le ceneri sono invece sostanze incombustibili (dovuti a additivi metallici, ossidi di ferro) che si possono formare in camera di combustione o nei collettori di scarico.

- una parte solubile (SOF): costituita da composti organici (idrocarburi) che possono derivare dall'olio lubrificante o dal combustibile. La frazione organica maggiore deriva dal primo ed è ricca soprattutto di composti quali il C20.
- Solfati (SO_4): possono derivare da acqua o acidi solforici e si formano a valle del processo di combustione, quando le temperature diminuiscono, quindi nel condotto di scarico.

Le percentuali di queste tre categorie presenti nel particolato dipendono dalle condizioni di funzionamento del motore, come mostrato nella *Figura 1.7* di seguito [3], cioè dal carico e dalla velocità di rotazione.

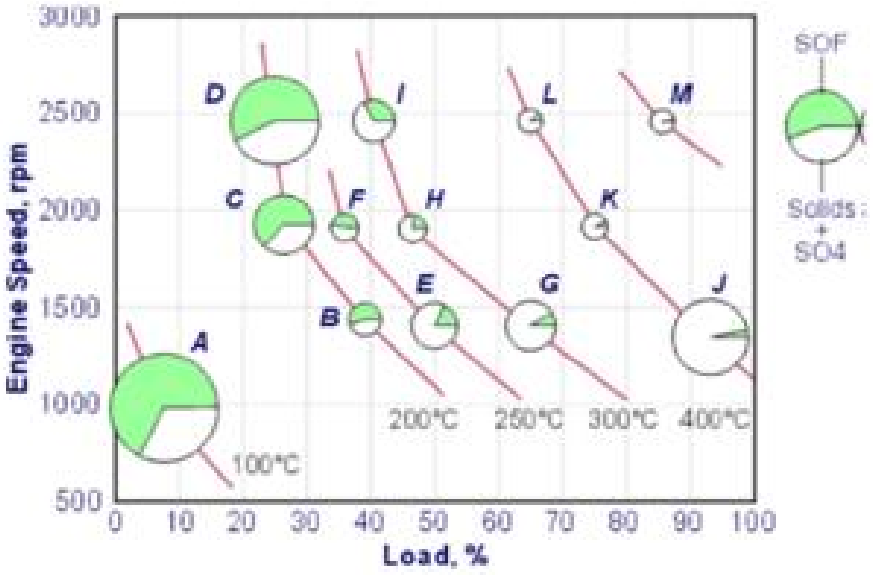

Figura 1.7 Variazioni di concentrazione delle parti costituenti il particolato al variare del carico motore (Load) e della velocità di rotazione del motore (Engine Speed).

Il problema delle emissioni di particolato è diventato sempre più importante per i produttori di mezzi di trasporto che montano motori Diesel poiché le ultime normative antinquinamento non richiedono più soltanto una limitazione delle emissioni in massa di PM, ma anche quella del numero di particelle emesse dal motore stesso. Infatti, le particelle di particolato possono avere dimensioni diverse, motivo per cui, le più piccole sono state trascurate per lungo tempo; tuttavia, recenti studi hanno dimostrato come anche queste siano dannose per la salute dell'uomo. In tal senso, può essere effettuata una classificazione delle particelle di PM riferita alla relativa grandezza:

- *Nuclei mode*: particelle ultrafini e nanoparticelle, le quali sono caratterizzate da una qualità diversa. Quelle più fini sono composti carboniosi circondate da elementi organici che sono stati adsorbiti dalla particella solida; la maggior parte sono molecole di SOF condensate a formare goccioline più grandi. Questo non avviene

durante la combustione, ma allo scarico o all'esterno quando le temperature sono particolarmente basse. Tali tipi di particelle sono molto instabili.

- *Accumulation mode*: le particelle fini di questo tipo sono per lo più tutte carboniose e, su di esse, durante la fase di scarico, si depositano dei composti organici
- *Coarse mode*: tale tipo si forma per usura delle parti di scarico e sono le particelle più grandi. Sono basse sia in numero che in massa, quindi hanno una bassa incidenza, ma alcune di queste possono formarsi anche a valle del filtro antiparticolato.

In *Figura 1.8* viene riportato l'andamento esemplificativo della concentrazione normalizzata di particelle di PM in funzione del diametro: ciò che si evince è che la distinzione tra numero di particelle di PM e massa totale di PM, effettuata nelle normative attuali, è rilevante, in quanto le curve dimostrano che è particolarmente differente il riferimento all'uno o all'altro. La distribuzione relativa al numero di particelle mostra che da uno scarico di un motore ad accensione per compressione, verranno emesse un numero elevato di particelle molto piccole e leggere; viceversa, la distribuzione relativa alla massa di PM mostra che, nonostante siano emesse in numero decisamente inferiore, le particelle in *accumulation mode* o *coarse mode* sono quelle di maggior peso e grandezza.

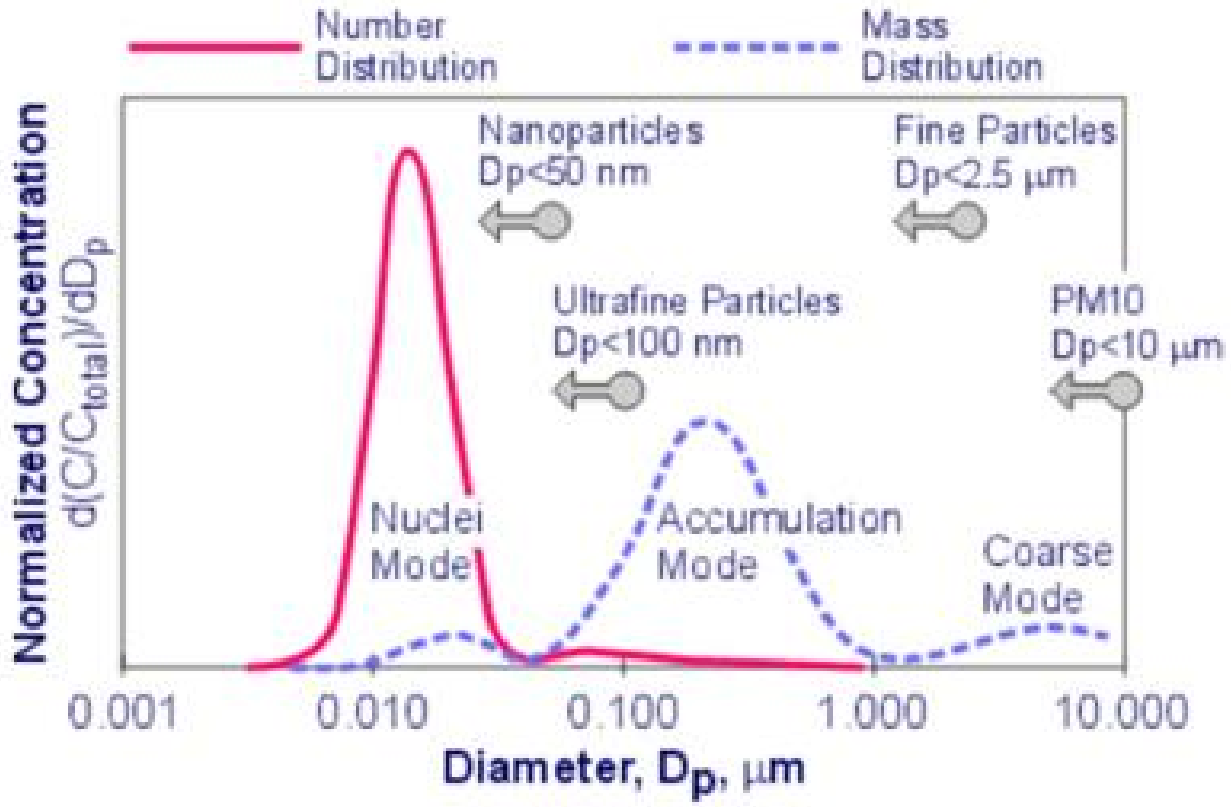

Figura 1.8 Distribuzione in massa e numero di particelle di PM

Per il presente lavoro di tesi si ritiene necessario fare un breve cenno riguardo l'*after-treatment* nei motori Diesel, esplicando esclusivamente ciò che riguarda il trattamento del particolato prima dell'immissione in atmosfera.

1.3 Cenno DPF

I gas combusti uscenti dai motori ad accensione per compressione, prima dell'immissione in atmosfera, vengono trattati tramite un sistema di dispositivi atti a ridurre le emissioni inquinanti, facendo rientrare i valori nei limiti imposti dalle normative. Tuttavia, per il presente lavoro, sarà fornita una descrizione per ciò che riguarda l'abbattimento del PM, ovvero il *Diesel Particulate Filter* (DPF). Tale filtro è un filtro meccanico costituito da un certo numero

di canali ciechi, in particolare risultano ciechi verso l'entrata o l'uscita in maniera alternata, così da forzare il flusso di gas a passare attraverso le pareti dotate di una certa porosità.

Figura 1.9 Schema dei canali ciechi di un DPF *[3]*

La separazione tra gas e particolato avviene meccanicamente e può avvenire in due modi:

- Letto profondo: le particelle hanno diametri inferiori rispetto ai pori della matrice e, passando attraverso questa, grazie ad azioni elettrostatiche ne rimangono imprigionate;
- Filtraggio a torta: le particelle, anziché passare attraverso una matrice porosa poiché hanno diametri leggermente più grandi, si depositano sulla matrice stessa e, conseguentemente, quelle successive si depositeranno sopra le precedenti a cascata.

Tale processo, però, è destinato ad avere una fine dal momento che il filtro necessita di essere liberato dal particolato accumulato per non otturarsi. Tale è proprio la rigenerazione del filtro antiparticolato tramite ossidazione del PM, caratteristica imprescindibile dei sistemi di trattamento dei gas combusti. I DPF possono essere classificati in base al tipo di rigenerazione:

- Periodica: è una rigenerazione forzata che utilizza come agente ossidante l'O_2 e a cui dobbiamo fornire calore. Dato che l'utilizzo di ossigeno comporta alte temperature (650°), viene utilizzato un catalizzatore per abbassarle;
- Continua: viene utilizzato come agente ossidante la NO_2 e, poiché ossida a basse temperature, non necessita del catalizzatore.

Non è possibile stabilire il momento della rigenerazione andando a misurare la caduta di pressione del filtro, dato che tutti i termini della caduta variano, aumentando all'aumentare della portata. Infatti, la presente ricerca volge allo studio di un modello che, andando a prevedere la formazione di *soot* in uscita motore e sapendo quanto ne viene emesso in atmosfera, fornisce la possibilità alla ECU di capire quando dover rigenerare.

Capitolo 2 – Introduzione al Machine Learning

Al giorno d'oggi gli apprendimenti delle macchine, e più in generale le reti neurali, sono ampiamente diffusi nei campi della robotica, della medicina e del network, mentre rimane un argomento del tutto attuale in campo automobilistico, in cui, solo da pochi anni, hanno iniziato ad essere investigati per apportare migliorie in termini di sicurezza, controllo ed emissioni delle vetture.

Per Machine Learning (ML), o apprendimento della macchina, si intende la scienza (e l'arte) di programmare i computers in modo tale che possono apprendere dai dati [7].

Nel presente lavoro di tesi viene utilizzato un codice, scritto in ambiente Python, che sfrutta un algoritmo di Machine Learning, nello specifico chiamato *Random Forest* proprio per la sua struttura a foresta e caratterizzato da alberi e foglie, ognuno con il suo specifico compito. Vedremo più tardi nel dettaglio come è costituito e come lavora tale algoritmo.

2.1 Classificazione degli algoritmi

Esistono diversi tipi di sistemi di Machine Learning e possiamo classificarli in categorie basate sui seguenti criteri:

- Possono o non possono essere allenati con la supervisione dell'uomo e tra i quali si hanno: *supervised, unsupervised, semisupervised* and *Reinforcement Learning*;
- Possono o non possono imparare incrementalmente durante l'applicazione;
- Se lavorano semplicemente comparando i nuovi dati a quelli già conosciuti, oppure rilevano i modelli tramite i dati di *training* per poi costruire un modello predittivo.

Tali criteri non sono esclusivi ma possono essere combinati l'un l'altro a comporre un modello più strutturato. Per il presente lavoro, saranno brevemente citati solo quelli

appartenenti alla prima categoria e poi spiegato nel dettaglio solo uno di questi perché utilizzato nel codice del modello.

I sistemi di ML possono essere classificati secondo il tipo e la quantità di supervisione che possono ottenere durante l'allenamento.

2.1.1 Supervised Learning

I tipi di algoritmi di apprendimento automatico di maggior successo sono quelli che automatizzano i processi decisionali generalizzando da esempi noti. In questa impostazione, che è nota come apprendimento supervisionato, l'utente fornisce all'algoritmo coppie di input e output desiderati e l'algoritmo trova un modo per produrre l'output desiderato dato un input. In particolare, l'algoritmo è in grado di creare un output per un input che non ha mai visto prima senza l'aiuto di un essere umano.

- Un tipico compito del *supervised Learning* è costituito dalla Classificazione, in cui l'algoritmo è allenato su diversi dati conosciuti, ciascuno con la loro classe[5] di appartenenza, per poi, successivamente, imparare a classificare quelli nuovi.

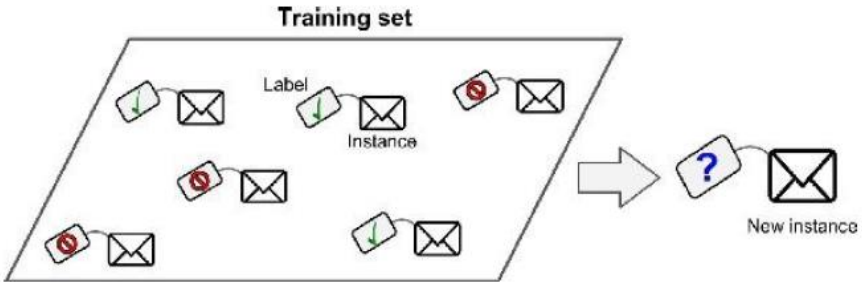

Figura 2.1. Un training set dotato di labels e un nuovo dato da classificare. (Unsupervised Learning) *[7]*

[5] La classe è l'insieme dei dati aventi proprietà comuni. Tale concetto è semantico e dipende strettamente dall'applicazione [8].

- Un diverso tipo si basa sulla regressione in cui l'algoritmo stesso deve prevedere un certo valore numerico di *target*, essendo dato un certo *set di features*, chiamate predittori. Per allenare tale sistema è necessario fornire, in generale, molti dati contenenti sia predittori che valori obiettivo, detti *labels*, che possono essere, quindi, considerati rispettivamente *input* e *output* del sistema. Quindi, risolvere un problema di regressione corrisponde ad apprendere una funzione approssimante delle coppie *input-output* date.

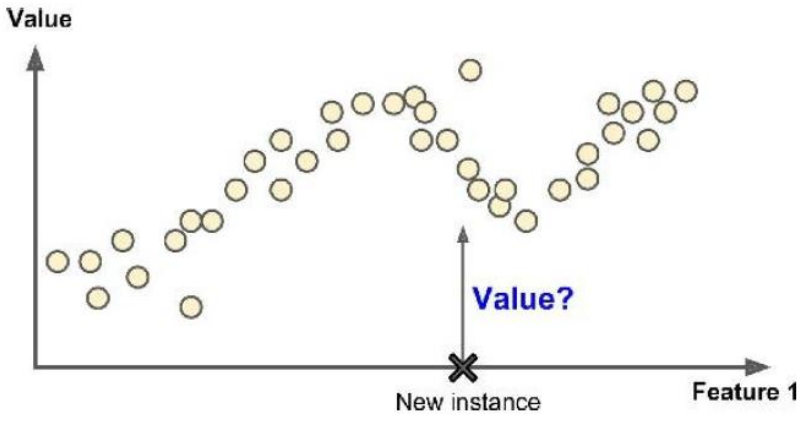

Figura 2.2. Regressione dei dati conosciuti e predizione di una nuova istanza (Unsupervised Learning) *[7]*.

Si può notare come alcuni algoritmi di regressione possono essere utilizzati per la classificazione, così come può valere il viceversa. I più importanti e conosciuti algoritmi di apprendimento supervisionato, dei quali tratteremo solo quello di interesse per il presente lavoro, sono:

- K-Nearest Neighbors
- Linear Regression

- Logistic Regression
- Support Vector Machines
- Decision Trees and Random Forests
- Neural networks

Tra questi, come precedentemente accennato, analizzeremo solo il caso degli alberi decisionali, *Decision trees*, e *Random Forests.*

2.1.2 Unsupervised Learning

Negli algoritmi di apprendimento non supervisionato il *training set* non è etichettato, cioè non contiene i valori di output tramite i quali allenare il modello e, quindi, non sono note le classi dei *pattern*[6] utilizzati per l'addestramento.

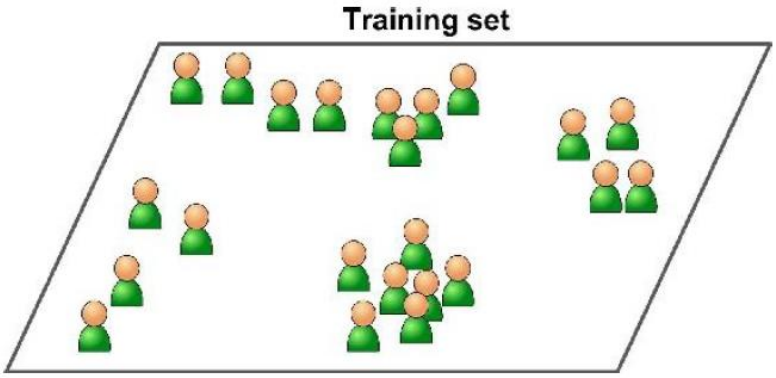

Figura 2.3. Esempio di Training set non etichettato (senza output). Unsupervised Learning

Tra i più importanti algoritmi di questa tipologia possiamo trovare:

[6] Si utilizza spesso il termine *pattern* per riferirci ai dati ed è preferibile non tradurlo in quanto termine maggiormente indicativo rispetto alla sua stessa traduzione.

- *Clustering*: individua gruppi (cluster) di pattern con caratteristiche simili in cui le classi del problema non sono note e i pattern non etichettati (senza output). Spesso il numero si cluster non è noto a priori, ma quelli individuati nell'apprendimento possono poi essere usati come classi. La natura non supervisionata del problema li rende più complicati i quelli di classificazione.

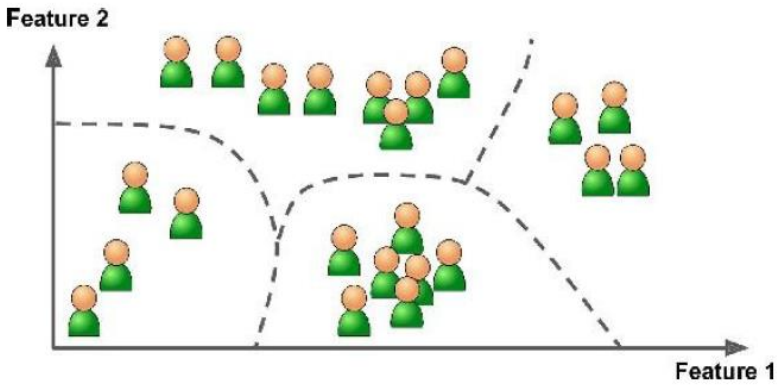

Figura 2.4. Esempio di *Clustering*

- Riduzione di dimensionalità: riduce il numero di dimensioni dei pattern in input, senza perdere troppe informazioni. L'operazione comporta una perdita di informazione, ma l'obiettivo è mantenere quelle importanti per il caso, dipendenti quindi dall'applicazione. Risulta quindi molto utile per rendere trattabili problemi con dimensionalità molto elevata, scartando informazioni ridondanti e/o instabili, permettendo quindi al codice di calcolare più velocemente grazie al minor spazio su disco occupato dai dati e, talvolta, avere prestazioni più alte.

2.1.3 Semisupervised Learning

Gli algoritmi di apprendimento semi-supervisionato lavorano con *training set* solo parzialmente etichettati e la distribuzione dei pattern senza output può aiutare ad ottimizzare la regola di classificazione. Molti algoritmi di questo tipo sono combinazioni tra quelli *supervised* e *unsupervised*.

2.1.4 Reinforcement Learning

Un algoritmo di Reinforcement Learning ha come obiettivo l'apprendimento di un comportamento ottimale a partire dalle esperienze passate. L'acquisizione della conoscenza è dedicata ad un agente, il quale osserva l'ambiente circostante ed esegue azioni per modificarlo, provocando passaggi da uno stato all'altro e ricevendo delle ricompense, dette *rewards* che possono essere anche dei *penalties* in senso di ricompense negative. L'obiettivo è apprendere l'azione ottimale in ciascuno stato, in modo da massimizzare la somma delle ricompense ottenute nel lungo periodo.

Si ritiene importante sottolineare un aspetto che riguarda il ML in generale: dal momento che lo scopo principale è selezionare un algoritmo di apprendimento ed allenarlo su un certo *set* di dati, la selezione risulta estremamente importante in quanto potremmo avere un "*bad algorithm*" e "*bad data*", nel senso che possono non essere compatibili, oppure si ha a disposizione solo un esiguo numero di dati per l'allenamento, o addirittura l'algoritmo scelto non è compatibile con l'applicazione stessa.

2.2 Training e Valutazione Prestazioni

In genere, il comportamento di un algoritmo di Machine Learning è regolato da un set di parametri che lo caratterizzano. L'apprendimento consiste nel determinare il valore ottimo di questi stessi parametri.

Quindi, dato un training set *Train* e un insieme di parametri, la funzione obiettivo $f(Train, \mathrm{param})$ può indicare [8]:

- L'ottimalità della soluzione da massimizzare;
- L' errore o perdita (detta *loss-function*) da minimizzare.

Tale funzione può essere ottimizzata esplicitamente, con metodi che operano a partire dalla sua definizione matematica (come per esempio il calcolo alle derivate parziali), oppure implicitamente, utilizzando metodi euristici che modificano i parametri in modo coerente con la funzione stessa.

La maggior parte degli algoritmi richiede di definire, prima dell'apprendimento vero e proprio, il valore dei cosiddetti *iperparametri* e, una volta scelti opportunamente, si esegue l'apprendimento per ciascuno, andando a prendere quelli che hanno fornito le prestazioni migliori. Esempi di iperparametri possono essere il numero di neuroni in una rete neurale, il grado di un polinomio usato in una regressione, il tipo di loss-function, il numero di alberi e foglio in un algoritmo Random Forest, e così via.

Andiamo allora a dare una definizione di quelle che vengono considerate le prestazioni dell'algoritmo e a definire i parametri che le caratterizzano.

2.2.1 Valutazione delle Prestazioni

Le prestazioni di un algoritmo possono essere valutate utilizzando direttamente la funzione obiettivo per quantificare le prestazioni. In genere, però, è preferibile usare una misura direttamente legata alla semantica del problema, ovvero parametri matematici del campo della statistica e probabilità. Anche in questo caso, si può fare una distinzione dipendentemente dal tipo di sistema, cioè discreto o continuo.

- Discreto: in un problema di Classificazione, l'accuratezza (da 0% a 100%) è la percentuale di pattern correttamente classificati, mentre l'errore è il complemento di questa.

$$Accuratezza = \frac{pattern\ correttamente\ classificati}{pattern\ classificati} \qquad (8)$$

$$Errore = 100\% - Accuratezza \qquad (9)$$

- Continuo: in un problema di Regressione viene valutato il *root mean square error* (*rmse*), cioè la radice della media dei quadrati degli scostamenti tra il valore vero e il valore predetto. In poche parole, questo parametro fornisce un'idea di quanto il sistema, con la sua predizione, si allontana dalla realtà dei dati ed è espresso dall'equazione (10):

$$rmse = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(pred_i - true_i)^2} \qquad (10)$$

Anche se *rmse* è generalmente preferibile come misura delle prestazioni in un algoritmo di regressione, talvolta si può utilizzare un'altra funzione chiamata *mean absolute error* (*mae*), chiamata anche *average absolute deviation*:

$$mae = \frac{1}{N}\sum_{i=1}^{N}|pred_i - true_i|$$

Quindi, sia *rmse* che *mae* sono due modi per misurare la distanza tra due vettori, quello delle predizioni e quello dei valori di target.

2.2.2 Training, Validation and Test

Il miglior modo per sapere come un modello si adatterà ai nuovi dati è provarlo sulle istanze stesse. Per far ciò il dataset fornito all'algoritmo viene diviso in due parti: *Training* e *Testing,* a sua volta il primo viene suddiviso in *training set* e *validation set:*

- *Training:*
    - *training test:* insieme di pattern tramite il quale addestrare l'algoritmo trovando il valore ottimo dei parametri;
    - *validation test:* insieme di pattern su cui il sistema tara gli iperparametri;
- *Testing:* insieme di pattern su cui l'algoritmo valuta le prestazioni finali.

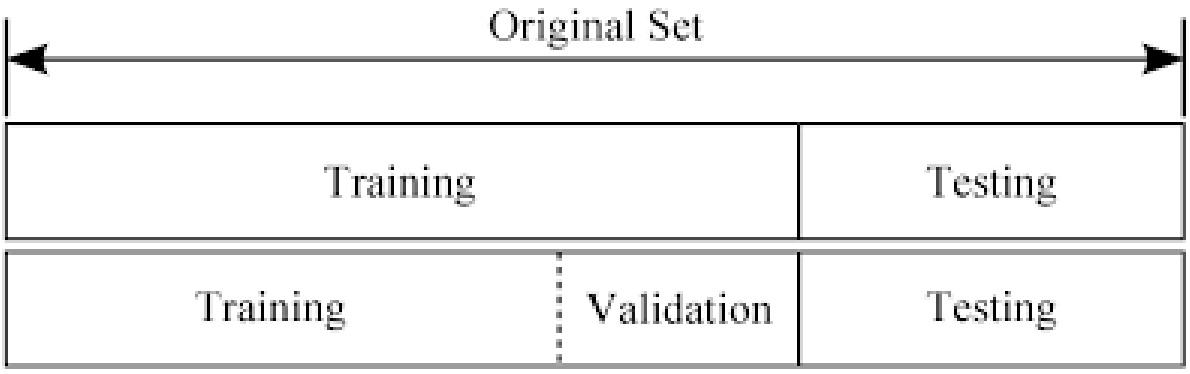

Figura 2.5. Schema di suddivisione del Dataset completo fornito all'algoritmo.

Per evitare di utilizzare troppi dati di training nel *validation set* e averne, quindi, a sufficienza per l'allenamento dell'algoritmo, una comune tecnica è quella della *cross-validation*: l'insieme dei dati di training viene suddiviso in sottoinsiemi complementari, dopo di che il modello è allenato su una parte di questi e validato sulla rimanente. Tale operazione è ripetuta per diverse combinazioni dei sottoinsiemi. Una volta definita quella che restituisce le prestazioni migliori e selezionati gli iperparametri, il modello finale viene allenato su il *training set* completo e, successivamente, applicato al *test set* per fornire l'errore generalizzato[7].

Uno dei primi obiettivi da perseguire durante l'addestramento è la convergenza sul *train set*. Se consideriamo un classificatore in cui l'addestramento prevede un processo iterativo, si ottiene convergenza quando:

- L'output della loss-function ha andamento decrescente rispetto al numero di iterazioni;
- L'accuratezza ha andamento crescente rispetto al numero di iterazioni.

[7] Valutando l'algoritmo sul *test set* si ha una misura del tasso di errore della predizione rispetto al caso di target. Tale è chiamato proprio *errore generalizzato,* o *out-of-sample error*.

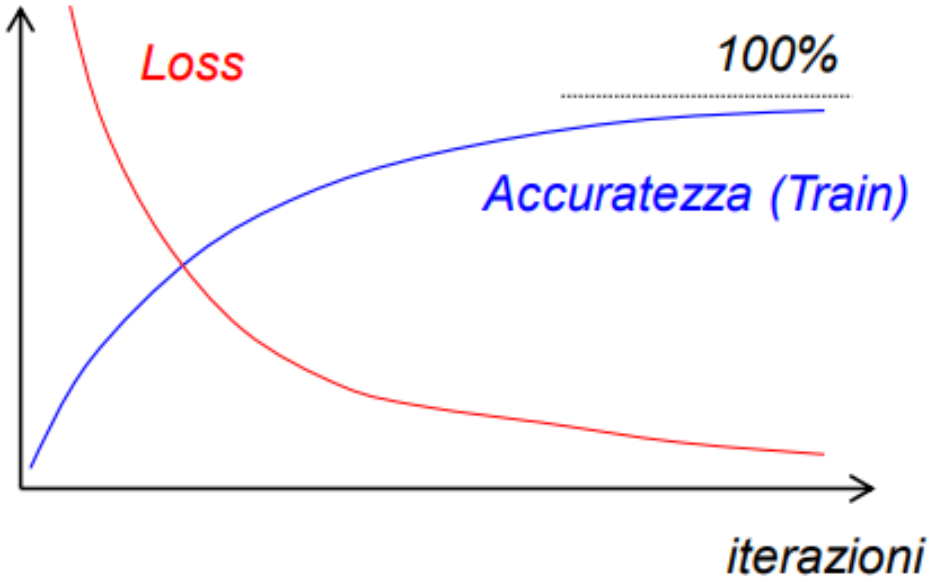

Figura 2.6. Andamenti qualitativi del *Loss* e dell'*accuratezza* per un modello ben costruito.

Se gli andamenti non sono quelli mostrati in *Figura 2.6* significa che molto probabilmente saranno presenti; per esempio:

- Se il *loss* non decresce (o oscilla) il sistema non converge: il metodo di ottimizzazione non è efficace, gli iperparametri sono fuori range, il tasso di apprendimento è inadeguato, possono esserci errori di implementazione, e così via.
- Se il *loss* decresce ma l'accuratezza non cresce, probabilmente è stata scelta una loss-function errata.
- Se l'accuratezza non si avvicina al 100% sul *Train*, i gradi di libertà del classificatore non sono sufficienti per gestire la complessità del problema.

Di seguito sono brevemente spiegati i due errori più comuni e importanti che possono verificarsi durante l'addestramento del modello.

2.2.3 Overfitting e Underfitting del Training Data

L'obiettivo ultimo di un algoritmo di Machine Learning è quello di massimizzare l'accuratezza sul *Test.* Se si ipotizza che i risultati ottenuti sul *validation set* siano rappresentativi del *testing*, allora dobbiamo massimizzare l'accuratezza della parte di validazione. Un modello per far sì che sia ben costruito, allenato e validato deve avere una generalizzazione[8] adatta al tipo di problema.

Dopo l'allenamento, l'algoritmo di apprendimento raggiungerà uno stato in cui sarà in grado di predire gli output per i pattern ancora non visionati (fase di test), cioè sarà in grado di generalizzare. Tuttavia, soprattutto nei casi in cui l'apprendimento è stato effettuato troppo a lungo o dove c'era uno scarso numero di esempi di allenamento, il modello potrebbe adattarsi a caratteristiche che sono specifiche solo del training set, ma che non hanno riscontro nel resto dei casi.

Quando un modello ha un numero eccessivo di gradi di libertà, o parametri che lo caratterizzano, rispetto al numero di osservazioni, raggiunge un elevata accuratezza sul *Train* ma non sulla validazione (scarsa generalizzazione): si parla di *Overfitting* di Train.

Risulta, quindi, buona norma partire con pochi gradi di libertà, controllabili attraverso gli iperparametri, e via via aumentarli monitorando accuratezza su *Train* e *Validation*.

Un esempio di *Overfitting* può essere osservato dalla seguente *Figura 2.7* [9], dove si può notare un confronto tra una regressione lineare e una interpolazione polinomiale.

[8] Si può definire *generalizzazione* la capacità di trasferire l'elevata accuratezza raggiunta sul Train alla fase di validazione.

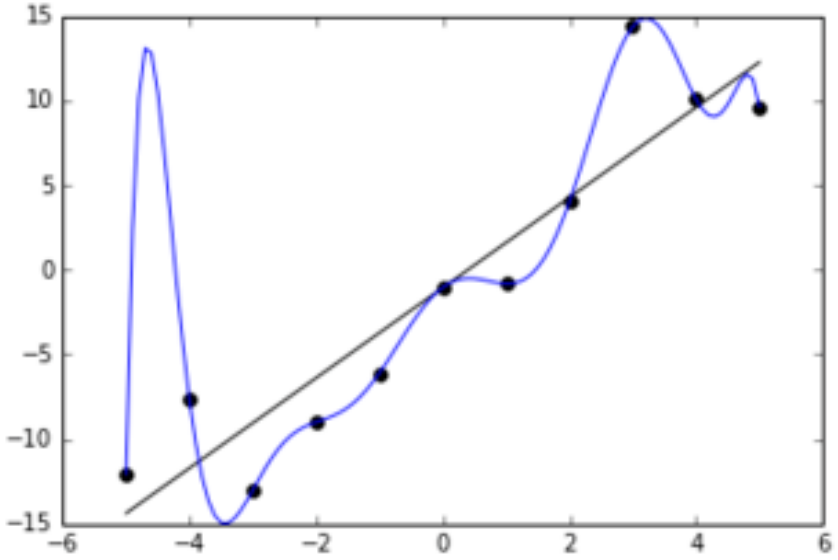

Figura 2.7. Esempio di Overfitting di una funzione polinomiale (linea blu). In questo caso una funzione lineare fornirebbe più alte prestazioni, costituendo una migliore generalizzazione, nonostante quella polinomiale si adatti in modo perfetto ai dati.

Opposto all'*Overfitting* abbiamo l'*Underfitting*, il quale si verifica quando il modello è troppo semplice per apprendere dalla struttura dei dati forniti. Infatti, ciò può accadere quando i parametri, che definiscono l'algoritmo, sono troppo pochi rispetto alla mole di dati di apprendimento. Le principali strategie per risolvere tale problema possono essere:

- Selezionare un modello più potente, con più parametri;
- Fornire migliori features all'algoritmo;
- Ridurre i vincoli del modello.

Capitolo 3 – Modello predittivo di particolato

Se nei capitoli precedenti si è voluto mostrare una breve panoramica riguardo la teoria che sta alla base del presente lavoro di tesi, introducendo la parte fenomenologica della combustione nei Diesel e le emissioni di inquinanti nel primo e una breve descrizione dell'apprendimento automatico delle macchine grazie ad algoritmi di Machine Learning nel secondo, in questo capitolo si vuole iniziare ad entrare nello specifico del vero e proprio lavoro sperimentale effettuato rispetto allo *state of art* [10].

Partendo da un precedente studio condotto per la previsione della massa di particolato, è stato sviluppato e validato un modello semi-empirico [10] con approccio DoE (Design of Experiment), che combina alcuni parametri relativi alla fisica ed alla chimica del problema e sfrutta alcuni parametri con metriche non direttamente misurate ed altri con metriche direttamente misurate, in cui sono stati identificate le variabili di input, del motore considerato (Euro 5, Diesel 2.0 L), più influenti sulla formazione e ossidazione di PM e introdotto un modello matematico esponenziale. L'intento della presente ricerca è quello di applicare e validare un modello di apprendimento automatico ML che sia svincolato totalmente dall'aspetto fenomenologico della formazione dell'inquinante e sia in grado di sottolineare i parametri motoristici più influenti sulla formazione di *soot*, tramite datasets di parametri appartenenti a metriche diverse: *Engine Control Unit (ECU)* e *test bench* (banco prova).

Il presente modello, come già accennato in precedenza, sfrutta un algoritmo di Machine Learning ad apprendimento supervisionato denominato *Random Forest*, il quale è costituito da un insieme di algoritmi più semplici, *Decision Trees*, che possono svolgere compiti sia di classificazione che di regressione (quest'ultimo corrispondente al nostro caso), ed è

implementato in un ambiente di sviluppo integrato *open source* in linguaggio informatico Python. A tale scopo, viene utilizzata una libreria *open source* di apprendimento automatico per questo linguaggio di programmazione, chiamata Scikit-Learn e all'interno della quale sono presenti gli algoritmi precedentemente citati, e progettata per lavorare con altre librerie come NumPy e SciPy che contengono la maggior parte delle funzioni utilizzate nel codice del modello predittivo. Andremo a vedere, prima di tutto, la struttura di tale codice per poi passare all'analisi e ai risultati.

3.1 Decision Trees

Il *Decision Tree* è una delle più comuni tecniche di classificazione e regressione usate al giorno d'oggi [11] in ambito Machine Learning. La *Figura 3.1* mostra un esempio di albero decisionale in forma flowchart composto da blocchi decisionali, i nodi (*nodes*), dai quali nascono a destra e sinistra i rami (*branches*) che portano ad altri blocchi decisionali, e blocchi di terminazione, foglie (*leafs*), dove sono raggiunte le decisioni finali [12]. Questo algoritmo è in grado di prendere un set di dati non familiari ed estrarre una serie di regole tramite le quali rendere l'uomo capace di comprendere il problema e i risultati, ed ha i seguenti vantaggi e svantaggi:

- Vantaggi:
    - Basso tempo di computazione
    - Facile comprensione dei risultati appresi
    - Può trattare *features* irrilevanti
- Svantaggi:
    - Incline all' Overfitting

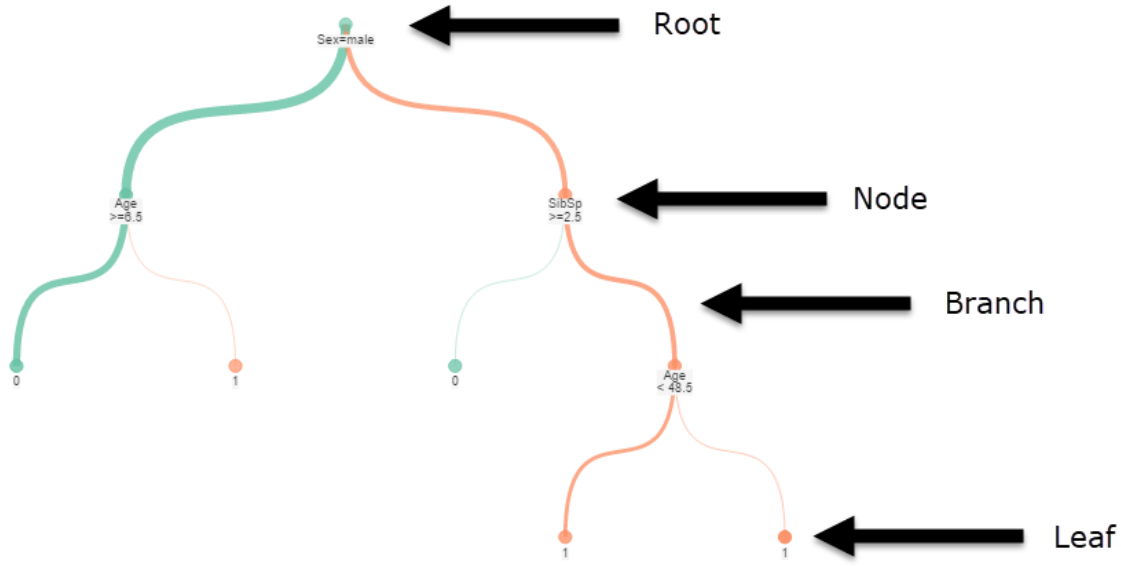

Figura 3.1. Esempio di Decision Tree in forma flowchart per un problema di classificazione.

Riprendendo la struttura dell'albero le varie parti contengono specifiche variabili, secondo le quali viene costruito l'albero stesso:

- Nodo: rappresenta una variabile come feature (o attributo);
- Ramo verso un nodo figlio: rappresenta un possibile valore per la feature;
- Foglia: valore predetto per la variabile target.

Per la costruzione dell'albero decisionale si parte fornendo all'algoritmo il dataset di punti contenenti tutti i valori associato ad ogni attributo, o *feature*, e i valori di target. I dati vengono suddivisi ogni volta nei nodi secondo i valori assunti dalle variabili *features*. Per determinare ciò, si prova ogni feature e si misura quale split fornisce i risultati migliori. Dopo di questo, viene diviso il dataset in sottoinsiemi, i quali sono fatti passare attraverso i primi rami del primo nodo decisionale: se i dati analizzati rispettano la stessa condizione imposta alla feature viene raggiunto il nodo foglia, altrimenti si prosegue verso un altro nodo decisionale in cui vengono divisi secondo la condizione nel nuovo nodo. Ogni nodo foglia finale definisce

una certa regione, entro le quali saranno poi valutate le nuove variabili obiettivo. Un esempio di un *decision tree* per la regressione è rappresentato in *Figura 3.2* [13].

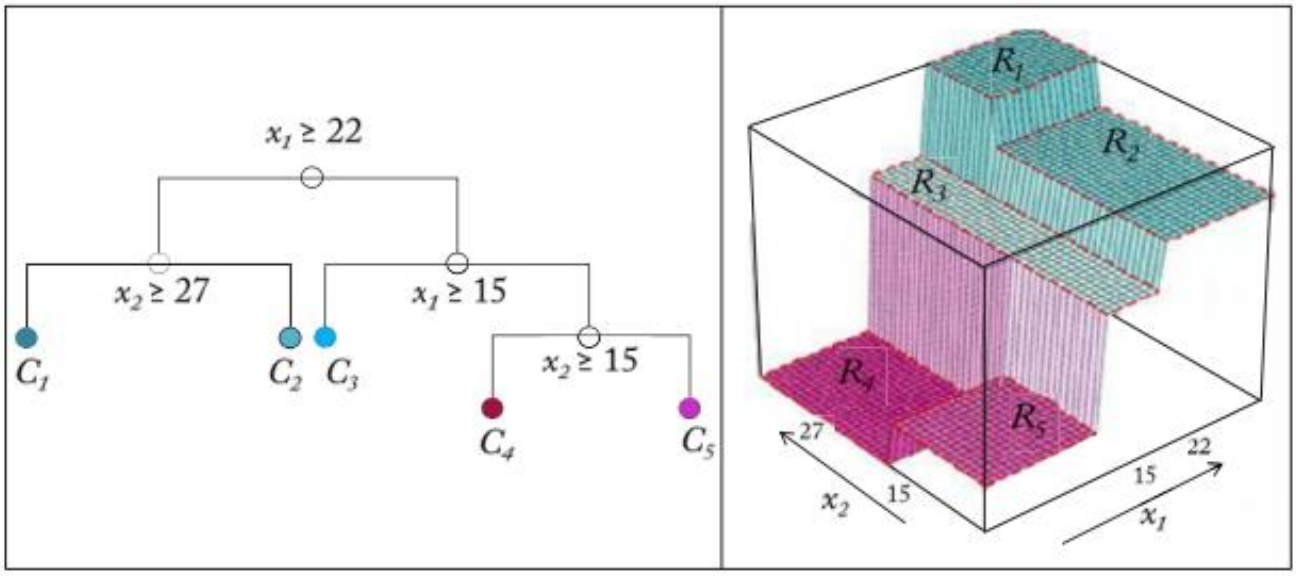

Figura 3.2. Esempio di albero decisionale di regressione e le corrispondenti regioni di suddivisione dello spazio degli input.

I *Decision Tree* adottano veramente poche assunzioni rispetto al training dataset (diversamente dai modelli lineari, per esempio, che ovviamente assumono dei dati lineari). Pertanto, se non vincolata, la struttura dell'albero si adatterà ai dati di training nel modo migliore possibile, cadendo molto probabilmente in *overfitting*. A tal proposito, si distinguono i modelli in:

- *Nonparametric*: i molti parametri disponibili non sono determinati a priori sul training, così che la struttura è libera di adattarsi perfettamente ai dati;
- *Parametric*: possiede un numero di parametri predeterminati, così i gradi di libertà sono limitati e, conseguentemente, viene ridotto il rischio di overfitting.

Come è noto a questo punto, per evitare questo problema è necessario limitare la libertà del Decision Tree durante l'allenamento. Tale processo è chiamato regolarizzazione degli iperparametri e, nonostante dipenda dall'algoritmo utilizzato, si può almeno limitare la massima profondità dell'albero. In Scikit-Learn, i principali `iperparametri`[9] che controllano la crescita della struttura e della forma del Decision Tree sono [7]:

- `max_depth`: indica la massima profondità del Decision Tree, definita in livelli.;
- `min_samples_split`: numero minimo di istanze che un nodo deve avere prima di essere diviso;
- `min_samples_leaf`: numero minimo di istanze che un nodo foglia deve avere;
- `min_weight_fraction_leaf`: è equivalente al precedente ma espresso come frazione sul numero totale di istanze pesate;
- `max_leaf_nodes`: massimo numero di nodi foglia;
- `max_features`: massimo numero di *features* valutate per dividere ogni nodo.

Incrementando quelli che esprimono un valore minimo o diminuendo quelli che esprimono un valore massimo si può regolarizzare il modello.

Risulta chiaro, quindi, come i Decision Trees siano semplici da comprendere e interpretare, facili da utilizzare, versatili e potenti. Tuttavia, hanno delle limitazioni, alcune già citate. Infatti, impostando dei valori di features nei nodi, realizzano delle condizioni al bordo ortogonali, visibile anche in *Figura 3.2*, e ciò li rende del tutto sensibili a piccole variazioni del training set, come per esempio una sua rotazione. Pertanto, dal momento

[9] Viene utilizzato, per gli iperparametri elencati, un carattere diverso da quello utilizzato nella scrittura del testo e corrispondente al carattere usato in un codice. Questo perché tali iperparametri, con i nomi riportati, sono direttamente utilizzati in linguaggio Python.

che l'algoritmo di training usato da Scikit-Learn è stocastico[10], si possono avere modelli diversi anche lavorando sullo stesso dataset di training (a meno che non si imposti un valore all'iperparametro `random_state`). Tale problema viene chiamato *Instability*, e può essere limitato fortemente dal famoso algoritmo *Random Forest*, il quale fa una media tra le predizioni su tanti alberi, come vedremo nel prossimo pagarafo.

3.2 Ensemble Learning & Random Forest

Quando si vuole ottenere la predizione di un certo evento, risulta più efficace l'utilizzo di un insieme di predittori piuttosto che il miglior predittore singolo, ottenendo quasi sempre previsioni migliori. Il gruppo di predittori viene chiamato *ensemble* e, di conseguenza, questa tecnica è denominata *Ensemble Learning.* Un algoritmo che sfrutta questa metodologia è chiamato *Ensemble method* [7]. Un gruppo di Decision Trees, ognuno con un diverso sottoinsieme casuale del training set, che forniscono una previsione finale è chiamato *Random Forest* ed è uno dei più potenti algoritmi di Machine Learning, nonostante la sua semplicità.

Come già accennato, per ottenere una buona previsione finale occorre avere una serie di predittori che usano diversi algoritmi di training. Un altro modo altrettanto efficace è quello di utilizzare lo stesso algoritmo di apprendimento per ogni predittore, ma quest'ultimi allenati su diversi sottoinsiemi del training set. Si può distinguere l'assegnazione dei sottoinsiemi ai singoli predittori in due categorie:

[10] Viene selezionato in modo casuale un set di features da valutare ad ogni nodo.

- *Bagging* (*bootstrap aggregating*) [14]: con ricambio;
- *Pasting:* senza ricambio.

Entrambi permettono ai dati di training di essere selezionati molte volte attraverso molti predittori. Il processo di selezione è mostrato nella *Figura 3.3.*

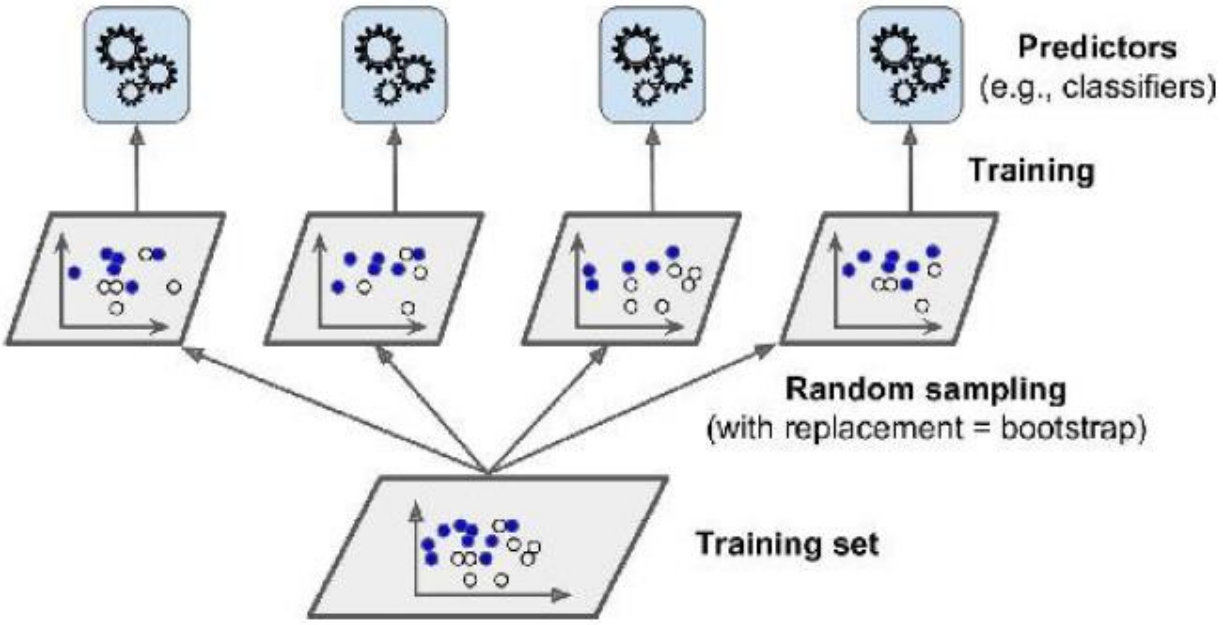

Figura 3.3. Campionamento di un Training set con bagging o pasting e training

Una volta che tutti i predittori sono stati allenati, l'ensemble può fornire la previsione per un nuovo dato semplicemente aggregando quella di tutti i predittori stessi. Come si può notare, i predittori sono allenati tutti in parallelo e ciò permette alla macchina di usare diversi core della CPU; anche la predizione può essere fatta in parallelo.

Ovviamente, come ci possiamo aspettare, la previsione di un gruppo di predittori risulterà molto più generalizzante rispetto al singolo, evitando così l'overfitting: l'ensemble possiede una varianza più piccola.

L'algoritmo *Random Forest* è, quindi, un insieme di Decision Trees generalmente allenati con il metodo di bagging (o qualche volta pasting) e può essere utilizzato per problemi discreti, classificazione, o continui, regressione, quest'ultimo costituente il presente caso. Questo algoritmo aggiunge un'ulteriore casualità, oltre all'assegnazione dei sottoinsiemi del training set ai singoli predittori: invece di ricercare la migliore feature come condizione per suddividere un nodo, trova la migliore feature in un sottoinsieme casuale dell'insieme globale delle feature stesse. Questo conduce ad una maggior diversità dei singoli alberi decisionali.

Una caratteristica fondamentale e di grande utilità, appartenente al Random Forest, è costituita dalla *Feature Importance*: funzione tramite la quale questo tipo di algoritmi misurano facilmente l'importanza relativa di ogni feature. Scikit-Learn trova l'importanza di una singola feature semplicemente calcolando quanto i nodi dell'albero, che utilizzano tale feature, riducono le impurità sulla media di tutti gli alberi decisionali. In altre parole, è una media pesata dove ogni peso di un nodo è uguale al numero di dati di allenamento associati al nodo stesso. Questa tecnica viene attuata successivamente all'allenamento, e i valori di importanza associati ai features sono tali che la loro somma sia uguale a uno. Passiamo a descrivere il funzionamento e la struttura del presente algoritmo.

3.3 Struttura e funzionamento del modello

Il modello predittivo lavora su dataset Excel di punti corrispondenti alle condizioni di funzionamento di un dato motore, nonché una mappa di punti definiti da due variabili:

- Velocità di rotazione del motore [rpm];
- Carico del motore espresso in *pme* (pressione media effettiva).

Per ogni punto di funzionamento è riportato un elevato numero di variabili corrispondenti ai parametri motoristici, che caratterizzano il processo di combustione e le emissioni conseguenti, i quali vengono misurati a banco (*test bench*) oppure stimati dalla centralina ECU (Engine Control Unit). Avendo a che fare con due diversi motori Diesel, la struttura dei dataset e le variabili impiegate saranno approfondite successivamente.

Per creare la foresta di alberi decisionali, Random Forest appunto, e quindi per generare ogni singolo predittore, è necessario fornire all'algoritmo un dataset di training, contenente sia gli input che gli output del sistema, e, successivamente, un dataset di testing, anch'esso contenente input e output, per generare i risultati in termini di accuratezza ed errore quadratico medio *rmse*.

3.3.1 Training

Dopo l'importazione dei dati da Excel si procede con il suddividere i dati in una parte dedicata all'allenamento e una al testing, tramite la funzione di Scikit-Learn:

```
train_test_split(X, y, test_size=test_size)
```

dove la variabile X corrisponde agli input e y agli output. Quindi questa funzione crea quattro variabili, ovvero gli input e gli output associati al training e gli input e gli output associati al testing, come si può vedere nella *Figura 3.4.*

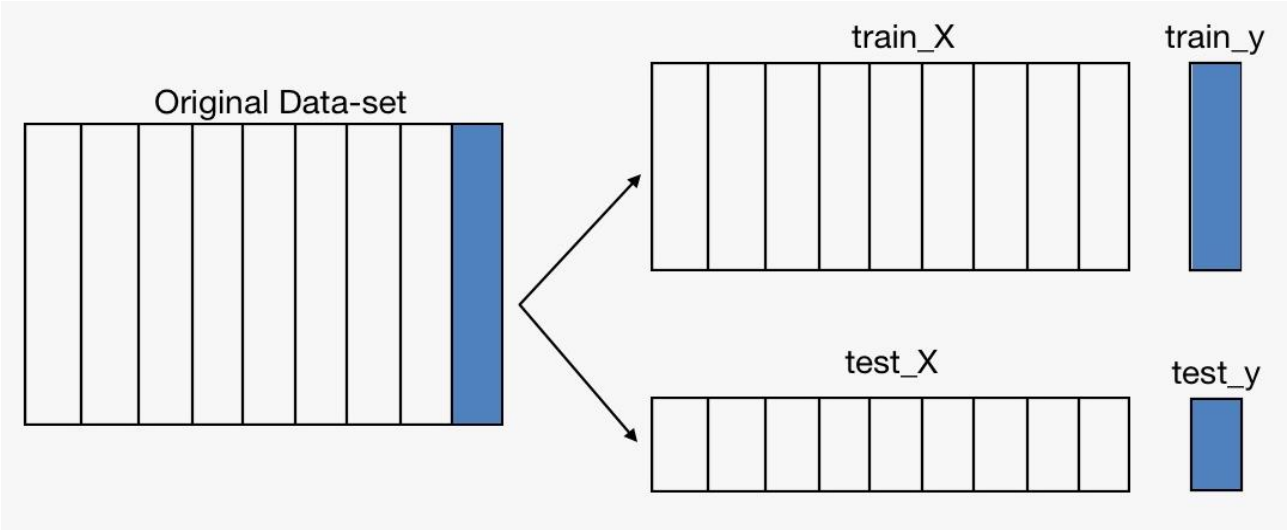

Figura 3.4. Suddivisione del Dataset originale in una parte di training e una di testing

Dopo aver preparato il dataset, una delle tecniche di Scikit-Learn più utilizzate per valutare un modello decisionale durante questa fase di allenamento è la cosiddetta *K-fold cross-validation*: suddivide in modo casuale il training set in K sottoinsiemi, chiamati *folds,* e successivamente allena e valuta l'algoritmo in K iterazioni, dedicando K-1 *folds* per l'allenamento e 1 *fold* per la validazione, sempre diversi ad ogni iterazione, come si può notare dalla *figura 3.5*. Nel nostro caso è stato adottando un valore di K pari a 10, il che vuol dire avere il training set suddiviso in dieci *folds* e dieci iterazioni.

Figura 3.5. K-fold Cross-validation con K=10.

L'algoritmo va a vedere quale delle dieci iterazioni utilizzate riporta i risultati migliori (risultati in `score` e non in errore quadratico medio *rmse*) e il risultato è costituito da un array contenente dieci valori di scores.

A questo punto risulta necessario sintonizzare l'algoritmo con i valori migliori dei propri parametri; questa operazione viene chiamata *Tuning*. Un modo potrebbe essere quello di provare manualmente diversi valori degli iperparametri dell'algoritmo, cambiandolo di volta in volta fino a trovare la combinazione migliore. Ovviamente questo risulta un lavoro lungo e tedioso, dal momento che esistono funzioni di Scikit-Learn che svolgono tale compito automaticamente. Tale è proprio conosciuta come `GridSearchCV`. Tutto ciò che dobbiamo fare è assegnare più valori agli iperparametri che verranno provati e, conseguentemente, la `GridSearchCV` ne troverà la combinazione migliore, utilizzando la cross-validation, cioè allenando sul training set e validando sul validation set.

Gli iperparametri usati all'interno del codice, i quali controllano lo sviluppo della struttura della *Random Forest* e di ogni singolo Decision Tree, sono stati scelti riferendosi alla letteratura attualmente presente [15] e risultano essere decisamente importanti perché permettono di accrescere la potenza predittiva del modello o di realizzarne uno più veloce. Gli iperparametri utilizzati da Scikit-Learn e nel presente codice sono:

- `n_estimators`
- `max_features`
- `min_samples_leaf`
- `n_jobs`
- `random_state`
- `max_depth`
- `min_samples_split`

Dove `n_estimator` rappresenta il numero di predittori (Decision Trees) che l'algoritmo costruisce prima di fare le operazioni di votazioni e predizioni e, in generale, un più alto numero di alberi incrementa le performance e permette delle predizioni più stabili a scapito di un maggior tempo computazionale; `max_features` è il massimo numero di features che Random Forest considera per suddividere un nodo; `min_sample_leaf` rappresenta il numero minimo di istanze richieste nei nodi foglia, quest'ultimi costituenti la base dell'albero; `n_jobs` indica alla macchina quanti processori (core) le è permesso usare ed è sempre impostato al valore -1, che significa nessun limite di utilizzo dei core; `random_state` è utilizzato per rendere replicabile l'uscita del modello, il quale produrrà sempre gli stessi risultati per un certo valore di questo parametro e se lavora con gli stessi iperparametri e training data; `max_depth` rappresenta la profondità di ogni albero della foresta definita in livelli e, in generale, più profondo è un Decision Tree e maggiori split possiede e maggiori

informazioni riesce a catturare; `min_samples_split` rappresenta il numero minimo di istanze richieste per suddividere un nodo interno e può variare considerando almeno un dato per nodo fino a prendere tutte le istanze in ogni nodo; incrementando tale parametro, ogni albero della foresta diventa più vincolato. L'impostazione dei migliori iperparametri è necessaria per evitare problemi di *overfitting* e *underfitting* [16], precedentemente spiegati, e non è mai una ricerca semplice e richiede molto spesso un elevato tempo computazionale, dato che la `GridSearchCV` deve investigare tutte le possibili combinazioni e trovare l'ottimale su un set diviso in tante iterazioni quanto è il valore K della cross-validation.

Le righe di codice utilizzate per le varie fasi di training sono riportate di seguito.

```
rfr = RandomForestRegressor(n_jobs=-1)
params = {'n_estimators': [ ],
              'max_features': [ ]
              'max_depth': [ ],
              'min_samples_leaf': [ ]
              'min_samples_split': [ ]}
first_grid = GridSearchCV(rfr, params, cv=10)
first_grid.fit(X_train, y_train)
```

Figura 3.6. Funzioni utilizzate in Python per il Training dell'algoritmo. Immagine realizzata con JupyterLab.

Risulta possibile visualizzare la migliore combinazione degli iperparametri, chiamata anche migliore estimatore, con la funzione `best_estimator:`

```
first_grid.best_estimator_
```

e i risultati in termini di score e deviazione standard.

3.3.2 Feature importance

Successivamente a quanto visto sopra, viene allenata la *Random Forest* utilizzando la migliore combinazione degli iperparametri trovati attraverso la `GridSearchCV`. Ciò risulta propedeutico per una delle più utili funzioni di questo algoritmo in cui viene creata una classifica dei features in base a quanto riducono le impurità all'interno dei nodi, cioè alla loro importanza. Questa tecnica, come già precedentemente spiegato, è rappresentata dalla *Feature importance*, tramite la quale vengono calcolate le importanze relative dei features, ovvero la loro somma restituisce uno. La funzione di Scikit-Learn utilizzata è `feature_importances_` e la riga di codice risulta essere:

```
rfr_bestE.feature_importances_
```

Questa tecnica permette, quindi, di ridurre la dimensione del problema, selezionando solo un certo numero di features tali per cui la somma dell'importanza di queste non sia maggiore al 90% del totale. A questo punto viene ricreato il dataset di training tenendo in considerazione solo i features importanti e, successivamente, riallenato l'algoritmo su questo nuovo training set.

3.3.3 Testing e plotting

Ultimo passaggio è stato quello di applicare l'algoritmo al dataset di test, considerando anche qui le sole features rilevanti, riportando i risultati delle simulazioni in termini di accuratezza e errore quadratico medio.

Inoltre, sono stati creati dei grafici per quanto riguarda la Feature importance, uno rappresentante la cumulata dei valori e l'altro realizzato a barre indicanti i valori di importanza di ogni feature, e un file Excel riportante tutti i risultati e dataset utilizzati, per ogni simulazione.

Capitolo 4 – Analisi Dati e Risultati

Nel presente capitolo saranno presentati i Dataset relativi a due motori Diesel di diversa taglia ed utilizzati per l'applicazione e la validazione dell'algoritmo di Machine Learning precedentemente spiegato. Un'analisi necessaria e propedeutica prima di procedere con le simulazioni è quella costituente lo studio fenomenologico del caso, cioè determinare quali sono i parametri del motore che influiscono sulla formazione del particolato; questo si traduce in un'individuazione dei dati che costituiscono gli *input* del Dataset in ingresso. Andiamo, quindi, a definire quali sono le grandezze con cui lavorerà il modello per determinare la predizione finale.

4.1 Applicazione del modello - motore Diesel 2.0L

Il primo motore di riferimento è un motore GM-E[11] Euro 5, le cui principali specifiche possono essere ritrovate in *Tabella 2* [17]*;* i punti motore analizzati sono relativi a dei *key-points* per i quali erano state ottimizzate le strategie di iniezione, come è possibile vedere in *Tabella 3.* I dati sperimentali sono stati acquisiti ad un banco di test dinamico di ICEAL-PT (Internal Combustion Engine Advanced Laboratory at the Politecnico di Torino), in attività di ricerca con GMPT-E (General Motors PowerTrain-Europe) per la calibrazione del motore [10].

[11] General Motors Euro 5 Diesel [10].

Engine type	2.0 L “Twin-Stage” Euro 5
Displacement	1956 cm^3
Bore x stroke	83.0 mm x 90.4 mm
Connecting rod length	145 mm
Compression ratio	16.5
Valves per cylinder	4
Turbocharger	Twin-stage with valve actuators
Fuel injection system	Common Rail 2000 bar piezo
Specific power and torque	71 kW/l to 205 Nm/l

Tabella 2. Specifiche motore GM-E di riferimento

Key-point	No. of tests
1500 x 2 PPM	147
1500 x 5 PPM	129
1500 x 5 PMA	131
2000 x 2 PPM	137
2000 x 5 PPM	134
2000 x 5 PMA	146
2500 x 8 PMA	130
2750 x 12 PMA	158

Tabella 3. Key-points, strategia di iniezione scelta e relativo numero di misurazioni effettuate. PPM *pilot pilot main*, PMA *pilot main after*.

Il Dataset completo, quindi, è così composto:

- 1112 punti motore: *rpm* (round per minute) x *pme* (pressione media effettiva);
- I punti si suddividono secondo due strategie di iniezione:
  - PMA: *pilot main after*
  - PPM: *pilot pilot main*
- Per ogni punto di funzionamento del motore si hanno 15 variabili misurate:
  - 14 parametri costituenti gli input (*features*)
  - 1 parametro costituente l'output della previsione (particolato)

Per l'analisi dei dati si sono considerati tre Dataset su cui applicare l'algoritmo e valutarne, così, la robustezza in base ai risultati ottenuti:

- Dataset completo: tutti i 1112 punti descritti precedentemente.
- Dataset PMA: sottoinsieme del completo con 565 punti di funzionamento con strategia di iniezione *pilot main after*
- Dataset PPM: sottoinsieme del completo 547 punti di funzionamento con strategia di iniezione *pilot pilot main*.

Risulta necessario spiegare in breve cosa sono e perché si utilizzano le strategie di iniezione. Esse corrispondono ai frazionamenti dell'iniezione totale di combustibile in parti più piccole, dipendentemente dal tipo, e sono estremamente utili per ridurre il rumore di combustione, legato ai picchi di pressione in camera, e il livello di emissioni di inquinanti. Nel presente caso possiamo fare distinzione tra le principali parti costituenti l'iniezione totale [1]:

- *Pilot*: elevato anticipo di iniezione, riduce il rumore di combustione all'avviamento ed ai bassi regimi riducendo il ritardo di accensione e, quindi, l'accumulo di combustibile. Si ottiene una combustione più progressiva, per cui, limitando le temperature massime raggiunte in camera, si riduce la formazione di NO_x. Risulta una piccola frazione (≈5%) del totale di combustibile iniettato.
- *Main*: è l'iniezione principale e serve a generare potenza dal motore (effetto utile).
- *After*: iniezione con breve ritardo rispetto alla principale e risulta utile per alzare leggermente la temperatura in camera di combustione e creare, così, un ambiente favorevole all'ossidazione del *soot*.

I parametri motoristici misurati, costituenti i *features* dell'algoritmo, risultano essere suddivisi in *input* e *output*:

Input	**Unità di misura**	
Speed	rpm	velocità di rotazione del motore
p_rail	bar	pressione di iniezione del combustibile (pressione nel rail)
p_boost	bar	pressione dell'aria nell' intake manifold (collettore di aspirazione)
SOI_main	Deg before PMS	inizio dell'iniezione della main
q_pil_1	mm^3/cyc/cyl	quantità di combustibile iniettata durante la prima *pilot*
q_pil2	mm^3/cyc/cyl	quantità di combustibile iniettata durante la seconda *pilot*

q_pil_tot	mm^3/cyc/cyl	quantità di combustibile totale della *pilot*
q_after	mm^3/cyc/cyl	quantità di combustibile iniettata durante la *after*
q_tot	mm^3/cyc/cyl	quantità totale di combustibile iniettato
q_air	mg/cyc/cyl	quantità di aria iniettata in camera
lambda relativo	-	rapporto tra α e α_{stec}
EGR valve position	%	posizione della valvola che controlla il flusso di gas esausti ricircolati (EGR)
Swirl actuator position	%	indica il grado di *swirl*
T_gas_intake	°C	Temperatura del gas all'ingresso motore

Tabella 4. Input di riferimento del modello.

Output	Unità di misura	
PM experimental	mg/cyc	quantità misurata di *particulate matte*

Tabella 5. Output di riferimento del modello.

Analizziamo adesso i risultati conseguenti alle simulazioni effettuate per ogni Dataset sopra elencato, riportanti gli iperparametri del *Random Forest*, le performance in termini di accuratezza ed errore quadratico medio e i tempi computazionali e i parametri più rilevanti tramite *feature importance*, relativamente a diversi valori di `train_test_split` adottati per analizzare la risposta del modello. Infatti, una tecnica di validazione, in grado di stabilire la robustezza del codice, è quella di analizzare l'andamento dell'accuratezza in funzione della dimensione dei dati di training e testing forniti.

4.1.1 Risultati – Dataset completo

I primi risultati ottenuti riguardano, quindi, il Dataset completo con tutte le caratteristiche che definiscono la struttura del *Random Forest* e dei *Decision Tree* che lo costituiscono, visibili nella *Tabella 6*.

Risultati Random Forest						
Complete Dataset						
train_test_split	test_size		20%	40%	60%	80%
Best Hyperparameters	N_estimators		500	500	500	100	20
	Max_features		5	5	5	5	7
	Max_depth		9	9	9	8	7
	min_sam_leaf		1	1	1	1	1
	min_sam_split		3	3	2	2	2
Variables Importance (90%)	Admissible variables		7	7	7	7	7
Score	BestP training score		0,979	0,981	0,977	0,975	0,965
	BestP training RMSE		0,00007	0,00008	0,00008	0,000095	0,00017
	Test final score		0,922	0,916	0,874	0,733	0,811
	Test final RMSE		0,00043	0,00032	0,00049	0,00098	0,00065
t_comput [s]	GrisSearchCV		8000	7000	6000	7300	7000
	Re-fitting for feature_importances		1	1	1	1	1
	FinalScore_test		2	2	2	1	1

Tabella 6. Risultati Random Forest relativamente al Dataset completo del motore GM-E Euro 5 Diesel 2.0 L.

Per lo studio è stato considerato, come si può notare, valori del *size* del dataset di training pari a 80%, 60%, 40% e 20% del totale (il *size* del testing è il complemento a 100 visibile in tabella). Possiamo osservare come una riduzione della dimensione del training dataset, con contemporaneo aumento di quella del test dataset, porti ad una diminuzione delle

performance (accuratezza e rmse), dovuto al fatto che il modello si allena su una quantità di dati minore.

Per quanto riguarda invece la *feature importance*, come risulta dalla *tabella 6*, il modello ha riportato i parametri che hanno avuto maggiore influenza sui valori di previsione del particolato (output), come riportato in *Tabella 7*.

Feature Importance - COMPLETE DATASET			
Test_size 20%		Test_size 40%	
1.	lambda [-]	1.	lambda [-]
2.	qtot [mm^3/cyc/cyl]	2.	qtot [mm^3/cyc/cyl]
3.	pboost [bar]	3.	pboost [bar]
4.	qair [mm^3/cyc/cyl]	4.	qair [mm^3/cyc/cyl]
5.	speed [RPM]	5.	pf [bar]
6.	pf [bar]	6.	Tgas(intake) [°C]
7.	Tgas(intake) [°C]	7.	speed [RPM]
Test_size 60%		Test_size 80%	
1.	lambda [-]	1.	lambda [-]
2.	pboost [bar]	2.	pboost [bar]
3.	Tgas(intake) [°C]	3.	Tgas(intake) [°C]
4.	qtot [mm^3/cyc/cyl]	4.	qtot [mm^3/cyc/cyl]
5.	pf [bar]	5.	pf [bar]
6.	qair [mm^3/cyc/cyl]	6.	qair [mm^3/cyc/cyl]
7.	speed [RPM]	7.	Swirl actuator pos [%]

Tabella 7. Feature importance prodotto dall'algoritmo.

Risulta evidente come la dosatura, con cui il motore si trova a lavorare, è il parametro predominante qualunque sia la *size* di allenamento del codice. Ciò non deve stupire in

quanto, insieme alla quantità di combustibile, alla quantità e alle condizioni termodinamiche dell'aria, la pressione del combustibile e la velocità di rotazione, la dosatura definisce la condizione di lavoro del motore per il quale si ha un determinato livello di particolato prodotto.

È stato condotto uno studio analogo con dimensione del dataset di training molto più grande, dato che, in applicazioni reali, è possibile avere una grandissima quantità di dati misurati. Si riportano, allora, i risultati per valori di `test_size` pari a 1%, 3% e 5%, cioè un training *size* rispettivamente di 99%, 97% e 95%, in *Tabella 8.*

Risultati Random Forest				
Complete Dataset				
train_test_split	test_size	1%	3%	5%
Best Hyperparameters	N_estimators	500	100	350
	Max_features	7	6	6
	Max_depth	12	8	10
	min_sam_leaf	1	1	1
	min_sam_split	4	4	3
Variables Importance (90%)	Admissible variables	7	7	7
Score	BestP training score	0,984	0,973	0,983
	BestP training RMSE	0,00006	0,000092	0,000058
	Test final score	0,956	0,94	0,915
	Test final RMSE	0,00015	0,00072	0,00064
t_comput [s]	GrisSearchCV	10000	10000	10000
	Re-fitting for feature_importances	10	10	10
	FinalScore_test	10	10	10

Tabella 8. Risultati per un Training dataset molto più ampio.

Come si poteva ben prevedere, i risultati di questo caso risultano essere migliori, con accuratezze molto più elevate ed errori più bassi, proprio grazie al fatto che il modello si allena su una quantità di dati più elevata, risultando in grado di predire i valori di particolato in maniera più precisa.

Feature Importance - COMPLETE DATASET		
Test_size 1%	Test_size 3%	Test_size 5%
1. lambda [-]	1. lambda [-]	1. lambda [-]
2. pboost [bar]	2. pboost [bar]	2. qtot [mm^3/cyc/cyl]
3. qtot [mm^3/cyc/cyl]	3. qtot [mm^3/cyc/cyl]	3. pboost [bar]
4. qair [mm^3/cyc/cyl]	4. qair [mm^3/cyc/cyl]	4. qair [mm^3/cyc/cyl]
5. pf [bar]	5. pf [bar]	5. pf [bar]
6. Tgas(intake) [°C]	6. speed [RPM]	6. Tgas(intake) [°C]
7. speed [RPM]	7. Tgas(intake) [°C]	7. speed [RPM]

Tabella 9. Feature importance per il training set allargato.

Si può notare come gli stessi parametri del motore siano rimasti quelli fondamentali per il modello.

Durante il pre-processing prima del lancio delle simulazioni, come descritto anche nella teoria dell'algoritmo di machine learning, un passaggio importante è stato sempre quello di imporre un range di valori per gli iperparametri del Random Forest così che, grazie alla `GridSearchCV`, potesse trovarne la combinazione migliore. A causa della variabilità nelle dimensioni dei dataset creati, tendenzialmente si è potuto osservare che una più alta quantità di dati necessitava di un più alto numero di alberi di regressione (`n_estimator`), a parità degli alti iperparametri. Allora è stato condotto uno studio per cercare una legge

generale che legasse gli andamenti di questi parametri caratteristici alla variazione della *size* del training dataset, e conseguentemente del test dataset, impostando l'iperparametro `random_state` ad un valore fissato; ciò si traduce nel fornire all'algoritmo un training dataset composto sempre dagli stessi dati estrapolati (e non li sceglie, quindi, casualmente come farebbe in genere non considerando questo parametro). Sono state fatte diverse simulazioni per ogni dimensione del dataset di allenamento, ovvero 80%, 60%, 40% e 20%, come riportato in *Tabella 10* e *Tabella 11*

Risultati Random Forest								
*Complete Dataset - **FIXED_RANDOM_STATE***								
train_test_split	test_size	20%				40%			
Best Hyperparameters	N_estimators	100	500	500	500	40	50	200	200
	Max_features	7	7	7	8	7	7	8	8
	Max_depth	9	11	13	12	11	12	9	10
	min_sam_leaf	1	1	1	1	2	1	1	1
	min_sam_split	4	4	4	4	4	4	4	4
Variables Importance (90%)	Admissible variables	7	7	7	7	7	7	7	7
Score	BestP training score	0,977	0,98	0,982	0,982	0,969	0,972	0,974	0,973
	BestP training RMSE	7,7E-05	6,5E-05	6,1E-05	6E-05	0,0001	9E-05	8,4E-05	9E-05
	Test final score	0,898	0,899	0,902	0,903	0,867	0,849	0,877	0,874
	Test final RMSE	0,0005	0,0005	0,0005	0,0005	0,0005	0,0007	0,0005	0,0005

Tabella 10. Risultati Random Forest per `random_state` fissato. Training dataset 80% e 60%.

Risultati Random Forest										
*Complete Dataset - **FIXED_RANDOM_STATE***										
train_test_split	test_size	60%				80%				
Best Hyperparameters	N_estimators	10	200	100	500	40	100	500	1000	1500
	Max_features	9	6	6	9	9	9	11	10	10
	Max_depth	9	11	13	13	9	11	13	11	15
	min_sam_leaf	1	1	1	1	1	1	1	1	1
	min_sam_split	4	4	4	4	4	4	4	4	4
Variables Importance (90%)	Admissible variables	6	7	8	6	6	7	6	7	6
Score	BestP training score	0,945	0,968	0,967	0,969	0,965	0,962	0,965	0,963	0,962
	BestP training RMSE	0,0002	0,00011	0,00011	0,0001	0,00017	0,0002	0,00017	0,0002	0,00018
	Test final score	0,847	0,852	0,851	0,852	0,833	0,837	0,84	0,833	0,843
	Test final RMSE	0,00058	0,00056	0,00057	0,0006	0,00056	0,0006	0,0005	0,0006	0,00053

Tabella 11. Risultati Random Forest per `random_state` fissato. Training dataset 40% e 20%.

Sono stati mantenuti fissi, inoltre, i parametri relativi al minimo numero di dati per i nodi foglia e al minimo numero di dati per lo split dei nodi interni (`min_sam_leaf, min_sam_split`), dato che è stato visto come in tutte le simulazioni il range di variazione di questi due era molto limitato, conducendo, quindi, uno studio sull'influenza della variabilità degli altri tre sulle prestazioni. Purtroppo, come ben osservabile, non si può estrapolare una legge a causa dell'andamento sempre diverso degli iperparametri in ogni *size* del dataset, con valori di accuratezza che restano circa gli stessi.

4.1.2 Risultati – Dataset PMA

Lo studio condotto sul dataset dei punti di funzionamento con strategia di iniezione *Pilot Main After* è analogo al caso precedente, con la differenza di una quantità di dati inferiore. L'analisi seguente non viene fatta per lo studio delle massime prestazioni del modello, in quanto solitamente richiederebbe molti punti di funzionamento del motore per l'allenamento dell'algoritmo, ma solo una verifica della robustezza, nel caso in cui si hanno a disposizione pochi dati. Tale ultimo caso può accadere, per esempio, in una acquisizione di punti motore, nella quale per qualche motivo conduce ad eliminarne tanti per errori di varia natura. Come mostreranno i risultati nelle tabelle successive, le prestazioni saranno decisamente più basse a causa di un minor potenziale costituito da un numero minore di campioni a disposizione per l'allenamento dell'algoritmo, ma comunque ritenuti accettabili. Una analisi di questo tipo, dipendentemente dalla decadenza dell'accuratezza al diminuire dei dati di training, può dare un'indicazione di quanto deve essere la *size* del dataset fornito. Infatti, a titolo di esempio, se al diminuire dei dati di training l'accuratezza non varia, significa che potrebbe risultare necessaria una quantità minore di dati per l'allenamento.

Risultati Random Forest					
Injection Strategy PMA					
train_test_split	test_size	20%	40%	60%	80%
Best Hyperparameters	N_estimators	50	100	30	10
	Max_features	6	5	6	3
	Max_depth	8	9	9	7
	min_sam_leaf	2	1	2	1
	min_sam_split	4	4	3	4
Variables Importance(90%)	Admissible variables	6	7	7	8
Score	BestP training score	0,97	0,973	0,964	0,935
	BestP training RMSE	0,00023	0,00016	0,00027	0,00026
	Test final score	0,883	0,848	0,819	0,638
	Test final RMSE	0,00076	0,00094	0,00094	0,0024
t_comput [s]	GrisSearchCV	7000	7000	7000	7000
	Re-fitting for feature_importances	1	1	1	1
	FinalScore_test	1	1	1	1

Tabella 12. Risultati Random Forest sul dataset PMA.

Per quanta riguarda la *Feature importance*, otteniamo i risultati mostrati in *Tabella 13.*

INJECTION STRATEGY_DATA SET			
Test_size 20%		Test_size 40%	
1. lambda	[-]	1. lambda	[-]
2. pboost	[bar]	2. pboost	[bar]
3. qtot	[mm^3/cyc/cyl]	3. qtot	[mm^3/cyc/cyl]
4. pf	[bar]	4. qair	[mg/cyc/cyl]
5. Tgas(intake)	[°C]	5. pf	[bar]
6. qair	[mg/cyc/cyl]	6. Tgas(intake)	[°C]
		7. speed	[RPM]
Test_size 60%		Test_size 80%	
1. lambda	[-]	1. lambda	[-]
2. pboost	[bar]	2. Tgas(intake)	[°C]
3. qtot	[mm^3/cyc/cyl]	3. qtot	[mm^3/cyc/cyl]
4. Tgas(intake)	[°C]	4. pboost	[bar]
5. qair	[mg/cyc/cyl]	5. pf	[bar]
6. pf	[bar]	6. EGR valve pos	[%]
7. speed	[RPM]	7. Swirl act pos	[%]
		8. SOImain	° before PMS]

Tabella 13. Feature importance per la strategia di iniezione PMA

I parametri motoristici importanti rimangono circa gli stessi fintanto che il dataset di training ha un certo numero di dati. Infatti, nell'ultimo caso, corrispondente ad un esiguo numero di campioni di allenamento pari al 20% del totale del dataset PMA, viene data una certa importanza a delle nuove variabili, le quali non saranno comunque considerate fondamentali a causa della condizione di allenamento povera di dati. Se considerarli dal punto di vista fenomenologico sarà determinato con studi che esulano dal presente lavoro.

4.1.3 Risultati – Dataset PPM

Analogamente a quanto fatto nei paragrafi precedenti, riportiamo i risultati per quanto riguarda la strategia di iniezione *Pilot Pilot Main* nella seguente *Tabella 14*:

Risultati Random Forest					
Injection Strategy PPM					
train_test_split	test_size	20%	40%	60%	80%
Best Hyperparameters	N_estimators	40	50	40	10
	Max_features	6	6	7	5
	Max_depth	10	8	9	8
	min_sam_leaf	1	1	2	1
	min_sam_split	2	2	4	5
Variables Importance(90%)	Admissible variables	7	8	7	7
Score	BestP training score	0,98	0,977	0,953	0,929
	BestP training RMSE	0,00001	0,00001	0,00003	0,00003
	Test final score	0,889	0,861	0,865	0,751
	Test final RMSE	0,00005	0,00009	0,000063	0,00015
t_comput [s]	GrisSearchCV	7000	7000	7000	7000
	Re-fitting for feature_importances	1	1	1	1
	FinalScore_test	1	1	1	1

Tabella 14. Risultati Random Forest per dataset PPM

e la *Feature importance* per il medesimo caso in *Tabella 15.*

INJECTION STRATEGY_DATA SET			
Test_size 20%		Test_size 40%	
1. lambda	[-]	1. lambda	[-]
2. qtot	[mm^3/cyc/cyl]	2. qtot	[mm^3/cyc/cyl]
3. qair	[mg/cyc/cyl]	3. qair	[mg/cyc/cyl]
4. pf	[bar]	4. pf	[bar]
5. pboost	[bar]	5. pboost	[bar]
6. qpiltot	[mm^3/cyc/cyl]	6. EGR valve pos	[%]
7. Tgas(intake)	[°C]	7. pil1	[mm^3/cyc/cyl]
		8. Tgas(intake)	[°C]
Test_size 60%		Test_size 80%	
1. lambda	[-]	1. lambda	[-]
2. qtot	[mm^3/cyc/cyl]	2. qair	[mg/cyc/cyl]
3. pboost	[bar]	3. qtot	[mm^3/cyc/cyl]
4. qair	[mg/cyc/cyl]	4. Tgas(intake)	[°C]
5. pf	[bar]	5. pboost	[bar]
6. qpiltot	[mm^3/cyc/cyl]	6. pf	[bar]
7. EGR valve pos	[%]	7. EGR valve pos	[%]

Tabella 15. Feature importance per dataset PPM

Anche in questo caso si hanno i medesimi parametri motoristici importanti, con l'ingresso di qualcuno nuovo ma comunque verso valori di importanza relativi modesti.

È risultato interessante anche porre l'attenzione su come si sarebbe comportato l'algoritmo se si fosse allenato su tutto il dataset PMA e una parte variabile del dataset PPM per poi essere testato sulla rimanente parte di quest'ultimo. Nel paragrafo successivo vengono mostrati i risultati relativi a questo caso.

4.1.4 Risultati – Mixed Dataset PMA e PPM

Il modello, in questa analisi, viene applicato al 100% del dataset PMA e ad una percentuale di quello PPM variabile, al diminuire dei dati di training, ovvero 80%, 60%, 40% e 20%. I risultati riportati di seguito.

Risultati Random Forest					
Mixed Injection Strategies					
			100% Training PMA + % Training PPM		
			80%	60%	40%	20%
Best Hyperparameters	N_estimators		30	50	30	30
	Max_features		5	4	4	3
	Max_depth		10	11	11	11
	min_sam_leaf		1	2	1	1
	min_sam_split		4	4	2	3
Variables Importance(90%)	Admissible variables		7	8	7	8
Score	BestP training score		0,977	0,974	0,981	0,978
	BestP training RMSE		0,00009	0,00011	0,000092	0,00011
	Test final score		0,832	0,819	0,682	0,645
	Test final RMSE		0,000057	0,000078	0,00018	0,00022
t_comput [s]	GrisSearchCV		4000	4000	4000	4000
	Re-fitting for feature_importances		1	1	1	1
	FinalScore_test		1	1	1	1

Tabella 16. Risultati della Random Forest sul mixing di dataset PMA e PPM

Si riportano anche i risultati in termini di importanza dei parametri.

MIXED INJECTION STRATEGY				
100% Training PMA + % Training PPM				
	80%	60%	40%	20%
1.	lambda [-]	lambda [-]	lambda [-]	lambda [-]
2.	pboost [bar]	qtot [mm^3/cyc/cyl]	qtot [mm^3/cyc/cyl]	qtot [mm^3/cyc/cyl]
3.	qtot [mm^3/cyc/cyl]	pboost [bar]	pboost [bar]	pf [bar]
4.	qair [mg/cyc/cyl]	qair [mg/cyc/cyl]	pf [bar]	pboost [bar]
5.	pf [bar]	Tgas (intake) [°C]	Tgas (intake) [°C]	Tgas (intake) [°C]
6.	Tgas (intake) [°C]	SOImain [deg bef PMS (+360)]	speed [RPM]	Swirl actuator pos [%]
7.	speed [RPM]	pf [bar]	qair [mg/cyc/cyl]	speed [RPM]
8.		speed [RPM]		SOImain [deg bef PMS (+360)]

Tabella 17. Feature importance per il mixing dei dataset PMA e PPM

Risulta ormai chiaro come l'importanza fenomenologica di alcuni parametri motoristici che prendono parte al processo di combustione, risultano determinanti nella formazione di particolato anche secondo il modello predittivo finora analizzato. Sono presenti, come nei due casi precedenti, alcuni parametri di minor importanza, ma che comunque devono essere considerati per la creazione dell'inquinante in quanto altrimenti porterebbe ad errori significativi.

4.2 Applicazione del modello – motore Diesel C11.0 L

Nel presente caso si applica il modello di apprendimento automatico precedentemente descritto ad un motore Diesel HD (Heavy Duty) Cursor 11.0 L a 6 cilindri, conforme agli standard europei Euro VI[12]. Non si dispone di tutte le specifiche tecniche del motore.

Le misurazioni, secondo le quali avviene la presa dei punti di funzionamento e dei parametri motoristici, sono state effettuate in condizioni *steady state* (condizioni stazionarie), con un numero totale di punti motore pari a 4879, quantità nettamente superiore al caso precedente. Il primo dataset completo, fornito di tutte le misurazioni e composto da 4879 punti per 206 variabili rilevate, subisce una prima scrematura a causa di errori di misura e variabili non significativi diventando così un nuovo dataset da 4711 punti per 27 parametri. Quest'ultimo è suddiviso in sei parti, dipendentemente dalla configurazione delle misurazioni:

- Prove su engine map senza EGR: totale 152 punti;
- Prove con diversi valori di SOI e pressione nel rail senza EGR: totale 949 punti;
- Prove su engine map con EGR: totale 149 punti;
- Prove con diversi valori di SOI e pressione nel rail con EGR: totale 959 punti;
- Trade off EGR/VGT[13]: totale 421 punti;
- DoE ASCMO: totale 2081 punti.

L'ultima configurazione non viene considerato per quasi tutte le simulazioni del presente caso. Facendo ciò, allora, risulterà un dataset definitivo di 2630 punti per 27 variabili, in cui

[12] Le classi di emissioni dei veicoli Heavy Duty sono rappresentate in numeri romani.
[13] Variazione della geometria della turbina.

alcune sono considerate due volte perché in un caso misurate a banco e nell'altro stimate in centralina ECU. Allora vengono creati due dataset, in base a quest'ultima tipologia di ottenimento dati:

- Bench Dataset: 2630 x 17 variabili (16 input e 1 output);
- ECU Dataset: 2630 x 17 variabili (16 input e 1 output).

Come si può notare, rispetto al motore precedente, si ha una dimensione dei dataset maggiore (più del doppio) sia in termini di features (variabili) che in quantità di dati per ognuna; allora, i range di scelta degli iperparametri saranno più piccoli per compensare l'allungamento dei tempi di computazione del modello.

I parametri motoristici misurati, costituenti i *features* dell'algoritmo, sono suddivisi in *input* e *output*, come mostra la *Tabella 18.*

Input	**Unità di misura**	
Speed	rpm	Velocità di rotazione del motore
q_tot	mm^3/cyl/cyc	Quantità totale di combustibile iniettata
p_rail	Bar	Pressione del combustibile nel rail
q_pil	mm^3/cyl/cyc	Quantità di combustibile dell'iniezione pilot
q_main	mm^3/cyl/cyc	Quantità di combustibile dell'iniezione main
SOI_pil	deg	SOI dell'iniezione pilot
SOI_main	deg	SOI dell'iniezione main
IMAP	bar	Pressione nel collettore di aspirazione
IMAT	K	Temperatura nel collettore di aspirazione
O2	%	Concentrazione di ossigeno
EMAP	bar	Pressione nel condotto di scarico

q_air	kg/cyc/cyl	Quantità di aria aspirata in camera
Inj.RAF	-	Rapporto tra quantità di aria e combustibile
EGR.HPEvalve	-	EGR tramite condotto
EGR.flap	-	EGR tramite flap
EGR.Xr	-	Grado di EGR: rapporto tra le moli di EGR e le moli totali tranne l'acqua nel collettore di aspirazione

Tabella 18. Parametri motoristici del motore Cursor11 utilizzati come *input* del modello.

Output	**Unità di misura**	
Emi.Soot	mg/cyc/cyl	Emissioni di *soot*

Tabella 19. Parametri motoristici del motore Cursor11 utilizzati come *output* del modello.

Una precisazione può essere fatta relativamente ai valori misurati di *soot* presenti nei dataset: i valori rilevati tramite prove a banco o stimate in centralina hanno come originaria unità di misura g/h, perciò si sono trasformati, come si può osservare dalla *Tabella 19,* per essere comparabili rispetto ai features con la seguente relazione:

$$^{mg}/_{cyc * \ cyl} = \frac{\frac{g}{h} * \ 1000}{rpm * \ 60 * \ 3}$$

Riportiamo di seguito i risultati per i diversi dataset creati, descritti precedentemente.

4.2.1 Risultati – Dataset Banco

I primi risultati riguardano il dataset delle misure rilevate a banco e in cui le simulazioni sono state condotte la *size* del training variabile, cioè 80%, 60%, 40% e 20%, tramite il parametro della Random Forest `train_test_split`.

Risultati Random Forest								
Bench Strategy								
train_test_split	test_size	20%			40%			
Best Hyperparameters	N_estimators	1500	70	70	250	500	100	70
	Max_features	10	10	10	7	11	11	10
	Max_depth	13	12	12	10	15	14	12
	min_sam_leaf	1	1	1	1	1	1	1
	min_sam_split	2	3	2	3	2	2	3
Variables Importance(90%)	Admissible variables	11	10	11	11	10	10	11
Score	BestP training score	0,95	0,934	0,956	0,906	0,944	0,937	0,965
	BestP training RMSE	8E-06	1E-05	7E-06	2E-05	1E-05	1E-05	4E-06
	Test final score	0,86	0,845	0,874	0,809	0,81	0,853	0,589
	Test final RMSE	2E-05	2E-05	1E-05	3E-05	2E-05	2E-05	9E-05
t_comput [s]	GrisSearchCV	8000	8000	8000	4000	2000	3000	5000
	Re-fitting for feature_importances	1	1	1	1	1	1	1
	FinalScore_test	1	1	1	1	1	1	1

Tabella 20. Risultati Random Forest dataset delle misure a banco per dimensione del training dataset pari a 80% e 60% del totale.

Risultati Random Forest							
Bench Strategy							
train_test_split	test_size	60%			80%		
Best Hyperparameters	N_estimators	40	50	50	40	40	40
	Max_features	4	10	9	6	7	11
	Max_depth	10	12	12	10	10	11
	min_sam_leaf	1	2	1	1	1	1
	min_sam_split	3	4	3	3	4	2
Variables Importance(90%)	Admissible variables	10	11	10	10	11	11
Score	BestP training score	0,916	0,937	0,922	0,933	0,917	0,954
	BestP training RMSE	2E-05	7E-06	2E-05	2E-05	8E-06	5E-06
	Test final score	0,643	0,587	0,777	0,577	0,573	0,583
	Test final RMSE	4E-05	7E-05	2E-05	5E-05	7E-05	7E-05
t_comput [s]	GrisSearchCV	4000	4000	4000	3000	3000	3000
	Re-fitting for feature_importances	1	1	1	1	1	1
	FinalScore_test	1	1	1	1	1	1

Tabella 21. Risultati Random Forest dataset delle misure a banco per dimensione del training dataset pari a 40% e 20% del totale.

Come è possibile notare, al diminuire delle dimensioni dei dati di allenamento, i risultati in termini di accuratezza ed errore quadratico medio decadono fino ad arrivare a valori molto bassi per un training size di 40% e 20%. Si ritiene necessario, allora, avere una quantit' maggiore di dati per tale parte.

In termini di *feature* importance sono riportati i risultati di seguito.

Feature importance								
test_size=20%								
1.	RAF	[-]	1.	RAF	[-]	1.	RAF	[-]
2.	O2	[%]	2.	O2	[%]	2.	O2	[%]
3.	p_rail	[bar]	3.	p_rail	[bar]	3.	p_rail	[bar]
4.	EGR_Xr	[-]	4.	EGR_Xr	[-]	4.	EGR_Xr	[-]
5.	SOI_main	[deg]	5.	SOI_main	[deg]	5.	SOI_main	[deg]
6.	T_intake	[K]	6.	q_tot	[mm^3/cyl/cyc]	6.	T_intake	[K]
7.	SOI_pil1	[deg]	7.	T_intake	[K]	7.	p_intake	[bar]
8.	q_tot	[mm^3/cyl/cyc]	8.	SOI_pil1	[deg]	8.	q_main	[mm^3/cyl/cyc]
9.	speed	[RPM]	9.	q_air	[mg/cyl/cyc	9.	q_air	[mg/cyl/cyc
10.	q_air	[mg/cyl/cyc	10.	q_main	[mm^3/cyl/cyc]	10.	q_tot	[mm^3/cyl/cyc]
11.	q_main	[mm^3/cyl/cyc]				11.	SOI_pil1	[deg]
test_size=40%								
1.	RAF	[-]	1.	RAF	[-]	1.	O2	[%]
2.	O2	[%]	2.	O2	[%]	2.	p_rail	[bar]
3.	p_rail	[bar]	3.	p_rail	[bar]	3.	EGR_Xr	[-]
4.	EGR_Xr	[-]	4.	EGR_Xr	[-]	4.	RAF	[-]
5.	T_intake	[K]	5.	T_intake	[K]	5.	SOI_main	[deg]
6.	SOI_main	[deg]	6.	SOI_main	[deg]	6.	SOI_pil1	[deg]
7.	speed	[RPM]	7.	q_air	[mg/cyl/cyc]	7.	q_tot	[mm^3/cyl/cyc]
8.	q_main	[mm^3/cyl/cyc]	8.	SOI_pil1	[deg]	8.	T_intake	[K]
9.	SOI_pil1	[deg]	9.	q_tot	[mm^3/cyl/cyc]	9.	speed	[RPM]
10.	p_intake	[bar]	10.	q_main	[mm^3/cyl/cyc]	10.	q_main	[mm^3/cyl/cyc]
11.	q_air	[mg/cyl/cyc]				11.	q_air	[mg/cyl/cyc]
test_size=60%								
1.	RAF	[-]	1.	O2	[%]	1.	RAF	[-]
2.	O2	[%]	2.	p_rail	[bar]	2.	O2	[%]
3.	EGR_Xr	[-]	3.	RAF	[-]	3.	p_rail	[bar]
4.	p_rail	[bar]	4.	SOI_main	[deg]	4.	EGR_Xr	[-]
5.	T_intake	[K]	5.	EGR_Xr	[-]	5.	T_intake	[K]
6.	speed	[RPM]	6.	T_intake	[K]	6.	SOI_pil1	[deg]
7.	q_air	[mg/cyl/cyc]	7.	speed	[RPM]	7.	speed	[RPM]
8.	q_tot	[mm^3/cyl/cyc]	8.	q_air	[mg/cyl/cyc]	8.	SOI_main	[deg]
9.	SOI_pil1	[deg]	9.	SOI_pil1	[deg]	9.	q_main	[mm^3/cyl/cyc]
10.	q_main	[mm^3/cyl/cyc]	10.	q_main	[mm^3/cyl/cyc]	10.	q_tot	[mm^3/cyl/cyc]
			11.	q_tot	[mm^3/cyl/cyc]			

test_size=80%		
1. RAF [-]	1. p_rail [bar]	1. O2 [%]
2. O2 [%]	2. SOI_pil1 [deg]	2. p_rail [bar]
3. EGR_Xr [-]	3. O2 [%]	3. RAF [-]
4. p_rail [bar]	4. RAF [-]	4. q_air [mg/cyc/cyl]
5. SOI_pil1 [deg]	5. T_intake [K]	5. EGR_Xr [-]
6. speed [RPM]	6. SOI_main [deg]	6. T_intake [K]
7. T_intake [K]	7. EGR_Xr [-]	7. SOI_main [deg]
8. q_air [mg/cyc/cyl]	8. speed [RPM]	8. SOI_pil1 [deg]
9. p_intake [bar]	9. q_main [mm^3/cyl/cyc]	9. speed [RPM]
10. q_main [mm^3/cyl/cyc]	10. q_air [mg/cyc/cyl]	10. q_main [mm^3/cyl/cyc]
	11. p_intake [bar]	11. q_tot [mm^3/cyl/cyc]

Tabella 22. Feature importance dataset delle misure a banco.

Anche in questo caso si ha la predominanza di una serie di parametri motoristici, che risultano fondamentali per il processo di combustione e la conseguente emissione di particolato dal punto di vista fenomenologico. Come già visto, la dosatura con cui lavora il motore risulta la caratteristica più importante, pertanto compare tra i primi posti anche la quantità' di ossigeno, strettamente correlata.

Come fatto per l'altro motore, anche in questo caso si è condotto uno studio per la verifica della solidità' del codice, prendendo una *size* del training dataset molto grande, ovvero 99%, 97% e 95%.

Risultati Random Forest				
Bench Strategy				
train_test_split	test_size	1%	3%	5%
Best Hyperparameters	N_estimators	1500	1500	1000
	Max_features	15	14	14
	Max_depth	19	17	15
	min_sam_leaf	1	1	1
	min_sam_split	2	2	2
Variables Importance(90%)	Admissible variables	10	10	10
Score	BestP training score	0,961	0,956	0,96
	BestP training RMSE	6E-06	7E-06	6E-06
	Test final score	0,949	0,89	0,882
	Test final RMSE	5E-06	1E-05	1E-05
t_comput [s]	GrisSearchCV	20000	20000	20000
	Re-fitting for feature_importances	10	10	10
	FinalScore_test	10	10	10

Tabella 23. Risultati Random Forest dataset delle misure a banco per dimensione del training dataset pari a 99%, 97%, 95% del totale.

E in termini di feature importance si ottiene:

Feature importance									
test_size=1%			test_size=3%			test_size=5%		
1.	RAF	[-]	1.	RAF	[-]	1.	RAF	[-]
2.	O2	[%]	2.	O2	[%]	2.	O2	[%]
3.	p_rail	[bar]	3.	p_rail	[bar]	3.	p_rail	[bar]
4.	SOI_main	[deg]	4.	SOI_pil1	[deg]	4.	SOI_pil1	[deg]
5.	SOI_pil1	[deg]	5.	SOI_main	[deg]	5.	SOI_main	[deg]
6.	q_tot	[mm^3/cyl/cyc]	6.	EGR_Xr	[-]	6.	q_tot	[mm^3/cyl/cyc]
7.	EGR_Xr	[-]	7.	q_tot	[mm^3/cyl/cyc]	7.	EGR_Xr	[-]
8.	q_air	[mg/cyl/cyc]	8.	q_air	[mg/cyl/cyc]	8.	q_main	[mm^3/cyl/cyc]
9.	q_main	[mm^3/cyl/cyc]	9.	speed	[RPM]	9.	speed	[RPM]
10.	speed	[RPM]	10.	q_main	[mm^3/cyl/cyc]	10.	q_air	[mg/cyl/cyc]

Tabella 24. Feature importance dataset delle misure a banco.

Le accuratezze sono decisamente elevate proprio per l'alto numero di dati di allenamento disponibili. Si ritiene quindi necessario un alto size del dataset per far apprendere l'algoritmo in maniera adeguata.

4.2.2 Risultati – Dataset ECU

Analogamente a quanto fatto per il dataset precedente, con il presente si riportano tutti i risultati delle simulazioni, con le stesse dimensioni dei dataset di training e test.

Risultati Random Forest								
ECU Strategy								
train_test_split	test_size		20%			40%		
Best Hyperparameters	N_estimators		500	250	50	1000	1000	1000
	Max_features		7	9	7	7	13	15
	Max_depth		10	12	10	10	15	16
	min_sam_leaf		1	1	1	1	1	1
	min_sam_split		2	2	2	2	2	2
Variables Importance(90%)	Admissible variables		11	11	11	11	10	10
Score	BestP training score		0,921	0,934	0,89	0,911	0,954	0,943
	BestP training RMSE		1E-05	0,00001	0,000018	2E-05	8E-06	1E-05
	Test final score		0,8	0,875	0,86	0,806	0,803	0,868
	Test final RMSE		2E-05	0,00002	0,000015	2E-05	2E-05	2E-05
t_comput [s]	GrisSearchCV		9000	14200	8000	4000	10000	10000
	Re-fitting for feature_importances		1	1	1	1	10	10
	FinalScore_test		1	1	1	1	10	10

Tabella 25. Risultati Random Forest dataset delle misure stimate in centralina per dimensione del training dataset pari a 80% e 60% del totale.

Risultati Random Forest								
ECU Strategy								
train_test_split	test_size	60%			80%			
Best Hyperparameters	N_estimators	40	60	40	40	60	50	
	Max_features	6	7	11	6	6	10	
	Max_depth	10	12	12	10	11	11	
	min_sam_leaf	1	1	1	1	1	1	
	min_sam_split	3	3	3	2	2	2	
Variables Importance(90%)	Admissible variables	12	11	11	12	11	11	
Score	BestP training score	0,946	0,936	0,962	0,936	0,953	0,955	
	BestP training RMSE	6E-06	0,000013	0,000004	8E-06	6E-06	4E-06	
	Test final score	0,554	0,764	0,578	0,567	0,582	0,573	
	Test final RMSE	8E-05	0,000027	0,000071	7E-05	7E-05	7E-05	
t_comput [s]	GrisSearchCV	4000	4000	4000	3000	3000	3000	
	Re-fitting for feature_importances	1	1	1	1	1	1	
	FinalScore_test	1	1	1	1	1	1	

Tabella 26. Risultati Random Forest dataset delle misure stimate in centralina per dimensione del training dataset pari a 40% e 20% del totale.

Riportiamo i parametri più importanti secondo la feature importance.

Feature importance		
test_size=20%		
1. RAF [-]	1. RAF [-]	1. RAF [-]
2. O2 [%]	2. O2 [%]	2. p_rail [bar]
3. p_rail [bar]	3. p_rail [bar]	3. O2 [%]
4. EGR_Xr [-]	4. EGR_Xr [-]	4. EGR_Xr [-]
5. speed [RPM]	5. T_intake [K]	5. T_intake [K]
6. SOI_pil1 [deg]	6. SOI_main [deg]	6. q_main [mm^3/cyl/cyc]
7. q_main [mm^3/cyl/cyc]	7. q_main [mm^3/cyl/cyc]	7. speed [RPM]
8. SOI_main [deg]	8. speed [RPM]	8. SOI_main [deg]
9. q_tot [mm^3/cyl/cyc]	9. q_tot [mm^3/cyl/cyc]	9. q_tot [mm^3/cyl/cyc]
10 T_intake [K]	10. SOI_pil1 [deg]	10. SOI_pil1 [deg]
11 p_intake [bar]	11. p_intake [bar]	11. p_intake [bar]
test_size=40%		
1. RAF [-]	1. RAF [-]	1. RAF [-]
2. O2 [%]	2. O2 [%]	2. O2 [%]
3. p_rail [bar]	3. p_rail [bar]	3. p_rail [bar]
4. EGR_Xr [-]	4. SOI_main [deg]	4. EGR_Xr [-]
5. T_intake [K]	5. SOI_pil1 [deg]	5. SOI_main [deg]
6. SOI_main [deg]	6. T_intake [K]	6. q_main [mm^3/cyl/cyc]
7. speed [RPM]	7. EGR_Xr [-]	7. SOI_pil1 [deg]
8. SOI_pil1 [deg]	8. q_tot [mm^3/cyl/cyc]	8. q_tot [mm^3/cyl/cyc]
9. q_main [mm^3/cyl/cyc]	9. q_main [mm^3/cyl/cyc]	9. p_intake [bar]
10 p_intake [bar]	10. speed [RPM]	10. T_intake [K]
11 q_tot [mm^3/cyl/cyc]		
test_size=60%		
1. O2 [%]	1. RAF [-]	1. O2 [%]
2. EGR_Xr [-]	2. p_rail [bar]	2. p_rail [bar]
3. T_intake [K]	3. O2 [%]	3. EGR_Xr [-]
4. p_rail [bar]	4. SOI_pil1 [deg]	4. SOI_main [deg]
5. SOI_main [deg]	5. EGR_Xr [-]	5. RAF [-]
6. RAF [-]	6. T_intake [K]	6. q_main [mm^3/cyl/cyc]
7. SOI_pil1 [deg]	7. q_air [mg/cyl/cyc]	7. SOI_pil1 [deg]
8. q_main [mm^3/cyl/cyc]	8. speed [RPM]	8. speed [RPM]
9. EGR_valve [-]	9. SOI_main [deg]	9. q_tot [mm^3/cyl/cyc]
10 q_tot [mm^3/cyl/cyc]	10. p_intake [bar]	10. T_intake [K]
11 speed [RPM]	11. q_main [mm^3/cyl/cyc]	11. EGR_valve [-]
12 p_exhaust [bar]		

test_size=80%								
1.	O2	[%]	1.	O2	[%]	1.	p_rail	[bar]
2.	EGR_Xr	[-]	2.	RAF	[-]	2.	O2	[%]
3.	T_intake	[K]	3.	p_rail	[bar]	3.	SOI_main	[deg]
4.	RAF	[-]	4.	EGR_Xr	[-]	4.	EGR_Xr	[-]
5.	p_rail	[bar]	5.	SOI_main	[deg]	5.	T_intake	[K]
6.	SOI_pil1	[deg]	6.	T_intake	[K]	6.	RAF	[-]
7.	SOI_main	[deg]	7.	SOI_pil1	[deg]	7.	SOI_pil1	[deg]
8.	q_tot	[mm^3/cyl/cyc]	8.	speed	[RPM]	8.	p_intake	[bar]
9.	speed	[RPM]	9.	q_main	[mm^3/cyl/cyc]	9.	q_main	[mm^3/cyl/cyc]
10.	q_main	[mm^3/cyl/cyc]	10.	q_tot	[mm^3/cyl/cyc]	10.	q_tot	[mm^3/cyl/cyc]
11.	p_intake	[bar]	11.	q_air	[mg/cyc/cyl]	11.	speed	[RPM]
12.	q_air	[mg/cyc/cyl]						

Tabella 27. Feature importance dataset ECU

Notiamo anche in questo caso gli stessi parametri fondamentali con delle piccole diversità nell'ordine per i training set minori, ovvero 40% e 20%, ma ciò è strettamente legato alle basse accuratezze di questi casi.

Anche in questo caso si riportano i risultati relativi all'analisi condotta sull'estensione dei dataset di training.

Risultati Random Forest				
Data Collection Strategy				
train_test_split	test_size	1%	3%	5%
Best Hyperparameters	N_estimators	1000	1000	1000
	Max_features	15	14	15
	Max_depth	19	19	17
	min_sam_leaf	1	1	1
	min_sam_split	2	2	2
Variables Importance(90%)	Admissible variables	10	10	10
Score	BestP training score	0,961	0,959	0,96
	BestP training RMSE	5,8E-06	0,000006	0,000006
	Test final score	0,988	0,83	0,88
	Test final RMSE	1,7E-06	0,000028	0,00001
t_comput	GrisSearchCV	20000	15000	15000
	Re-fitting for feature_importances	10	10	10
	FinalScore_test	10	10	10

Tabella 28. Risultati Random Forest dataset delle misure stimate in centralina per dimensione del training dataset pari a 99%, 97% e 95% del totale.

Collection Data Strategy		
test_size=1%	test_size=3%	test_size=5%
1. RAF [-]	1. RAF [-]	1. RAF [-]
2. O2 [%]	2. O2 [%]	2. O2 [%]
3. p_rail [bar]	3. p_rail [bar]	3. p_rail [bar]
4. SOI_main [deg]	4. SOI_main [deg]	4. SOI_main [deg]
5. q_main [mm^3/cyl/cyc]	5. EGR_Xr [-]	5. SOI_pil1 [deg]
6. SOI_pil1 [deg]	6. q_main [mm^3/cyl/cyc]	6. q_main [mm^3/cyl/cyc]
7. EGR_Xr [-]	7. SOI_pil1 [deg]	7. EGR_Xr [-]
8. q_tot [mm^3/cyl/cyc]	8. q_tot [mm^3/cyl/cyc]	8. speed [RPM]
9. speed [RPM]	9. T_intake [K]	9. q_tot [mm^3/cyl/cyc]
10. T_intake [K]	10. speed [RPM]	10. T_intake [K]

Tabella 29. Feature importance dataset ECU.

In questo si vede bene l'importanza del rapporto tra aria e combustibile, della concentrazione di ossigeno e pressione nel rail, in quanto il modello si allena su una grande quantità' di dati.

Conclusioni

L'elaborato di tesi si è occupato della applicazione e validazione di un modello Random Forest, appartenente alla branca dell'intelligenza artificiale chiamata Machine Learning, per la stima della massa di particolato in condizioni stazionarie in motori Diesel, e, più precisamente, in due motori di taglia differente.

Nella prima parte del lavoro di entrambi i motori Diesel, si è applicato l'algoritmo agli interi dataset creati per studiarne e analizzarne le prestazioni al massimo del modello, potendo osservare, grazie alla *feature importance*, quali variabili risultassero determinanti per le previsioni e dando la possibilità in futuro di legarli all'aspetto fenomenologico. Infatti, da questo è ben visibile la potenzialità dell'algoritmo: dal punto di vista chimico-fisico, dosatura, pressione e inizio di iniezione del combustibile, quantità di combustibile, quantità di aria e concentrazione di ossigeno, percentuale di gas combusti ricircolati sono disposti in prima linea nella formazione e ossidazione di *soot*, esattamente come riesce a caratterizzarli il modello, definendoli i più importanti. Tutto ciò è stato considerato valido dai risultati ottenuti considerando un elevato numero di dati di training, in quanto per un *training size* di 99% e 97% si sono raggiunte accuratezze anche nettamente superiori al 90% (in un caso siamo arrivati fino al 98.8%).

La validazione alle massime potenzialità di cui sopra è da distinguere con il successivo passo del presente lavoro, costituito dall'analisi della robustezza del modello mediante variazione della dimensione del dataset di allenamento, considerando quelle casistiche secondo le quali, per motivi legati alla reperibilità ed errori di misura, non si disponesse di un sufficiente numero di dati previsti per il training del modello. Lo studio è stato condotto diminuendo di volta in volta la *size* del dataset di training dall'80% al 20% e analizzando i valori di accuratezza così ottenuti. In alcuni casi l'algoritmo ha riportato dei valori superiori

all'' 80% mentre in altri le accuratezze sono decadute fino a valori del 50-60% per una dimensione molto modesta della capacità di allenamento, indice, in questi ultimi, della necessità di avere più dati a disposizione.

Tuttavia, va osservato come l'attuale lavoro svolto costituisce solo uno dei primi passi per la realizzazione di un modello virtuale solido, strutturato, affidabile e pronto a sostituire le mappe che accompagnano i sensori reali a valle DPF. Infatti, uno dei principali step successivi è quello di estendere lo studio alla formazione degli ossidi di azoto NO_x, inquinante prodotto dai motori Diesel in quantità decisamente corpose, creando un codice più generalizzante che analizzi il campo delle emissioni inquinanti, prodotti da questi ICE, da una prospettiva più ampia ed ampliandolo tramite tecniche di apprendimento più spinto e profondo, conosciute come Neural Networks. Risulta essenziale notare, inoltre, come un aspetto fondamentale per l'apprendimento di una macchina, nei campi del Deep Learning, sia quello di considerare una normalizzazione dei dati, non considerato nel presente lavoro a causa della ridotta profondità dell'algoritmo Random Forest utilizzato.

Indice delle figure

Indice delle Tabelle

Bibliografia

[1] F. Millo, *Propulsori termici, Laurea Magistrale Ingengeria meccanica, Propulsione dei veicoli terrestri,* (Appunti del corso, 2018).

[2] J. E. Dec, A Conceptual Model of DI Diesel Combustion Based on Laser-Sheet, SAE Technical Paper Series, 1997.

[3] E. Spessa, Controllo delle Emissioni Inquinanti, Laurea Magistrale ingegneria meccanica, Politecnico di Torino, (Appunti del Corso, 2018).

[4] J. Dec, «SAE 970873,» 1999.

[5] D.Slebers, «SAE 2001-01-0530, D.Slebers,» 2001.

[6] M. B. T. Kamimoto, SAE Transactions - Journal of Engines, (1988).

[7] A. Gèron, Hands-on Machine Learning with Scikit-Learn & TensorFlow, O'Reilly, 2017.

[8] D. Maltoni, «Fondamenti di Machine Learning,» sito web: http://bias.csr.unibo.it/maltoni/ml/#, Università di Bologna.

[9] Ghiles, «Wikipedia,» 11 Marzo 2016. [Online]. Available: https://commons.wikimedia.org/wiki/File:Overfitted_Data.png.

[10] D. M. E. S. Roberto Finesso, «Development and validation of a semi-empirical model for the estimation of particulate matter in diesel engines,» *Energy Conversion and Management,* 2014.

[11] G. S. &. J. Elder, Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions, (Morgan and Claypool), 2010.

[12] P. Harrington, Machine Learning In Action, Shelter Island, NY 11964: Manning Pubblications Co., 2012.

[13] T. T. R. F. J. Hastie, The Elements of Statistical Learning-Data Mining, Inference and Prediction., Springer, 2001.

[14] L. Breiman, Bagging Predictors, Statistics DepartmentUniversity of CaliforniaBerkeley, 1996.

[15] N. Donges, «Towards Data Science, 'The random Forest Algorithm',» 22 02 2018. [Online]. Available: https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd.

[16] M. B. Fraj, «Medium, All Things AI,» 21 12 2017. [Online]. Available: https://medium.com/all-things-ai/in-depth-parameter-tuning-for-random-forest-d67bb7e920d.

[17] C. Maino, «Tesi di Laurea Magistrale in ingengeria meccanica: "Validazione di un modello semi-empirico per la stima della massa di particolato in motori Diesel",» 2017.

Ringraziamenti

Ringrazio innanzitutto Daniela e Claudio per la loro massima e impeccabile disponibilità e per avermi dato la possibilità di intraprendere un percorso originale e affascinante che spero di cuore continui con la loro collaborazione.

Il mio lavoro di tesi lo dedico a mia mamma Manuela, senza la quale tutto questo non ci sarebbe stato, il mio mental coach, la mia guida, la persona che mi ha reso quello che sono e che mi ha fatto apprezzare le cose belle e brutte della vita, ma soprattutto colei a cui voglio un bene smisurato e che, forse, non ringrazierò mai abbastanza. Ringrazio profondamente la mia piccola ma grande Zuzzu, fonte di ispirazione per tante scelte fatte durante la mia vita e che ammiro profondamente. E poi c'è Fabiolino, entrato in famiglia e nella mia vita con grande impeto, persona dal gran cuore che stimo tanto e ringrazio altrettanto per tutto quello che ha fatto e sta facendo perché è anche grazie a lui che sono quello che sono e faccio quello che faccio; e poi diciamolo 'sopporta la Tura'! Ringrazio i miei dolci e adorabili nonnini per tutto quello che hanno fatto. Come ultimo, ma non per importanza, ringrazio dal profondo del cuore mio babbo Luca a cui voglio un bene dell'anima, anche se lo vedo poco frequentemente a causa della distanza, ma sono consapevole della sua continua presenza.

Ringrazio Matteo, Fana, Steppone, Franchina, Gianmarco, Simone e Bitto per i momenti torinesi belli trascorsi insieme e per quelli all'insegna della disperazione presami.
Un pensiero particolare va a "La gente ignora Giuly", ovverosia Tiziano, Adriano, Leonardo, e Gigi con cui ho trascorso la maggior parte della mia vita a Torino e con cui ho condiviso tanto. Per questo vi ringrazio particolarmente.
Un ringraziamento speciale ai miei 'Coinqui' Giulia e Luca che mi hanno accolto come una persona di casa fin da subito.
Dulcis in fundo a Torino, un enorme ringraziamento va a Nicco, conosciuto neanche un anno fa, con cui ho condiviso così tanti momenti intimi, pensieri ed emozioni da essermi affezionato particolarmente.

Un caloroso ringraziamento al mio 'valdarnotto' preferito, Il Barchie, e a Fede sempre presente, con cui ormai condivido una vita intera e a cui devo molto, che conosce ogni mio lato, che riesce a farmi sorridere anche nei momenti peggiori. Grazie davvero!
Vorrei ringraziare in maniera adeguata anche Matte, una sorta di mentore per la sua sfrenata e ineguagliabile saggezza.
Al Caso, detto anche Pallino, Gianfranco, Miranda e chi può ne ha più ne metta, la mia anima gemella, colui che mi fa pensare e riflettere, che riesce a tirar fuori ogni mio pensiero, ogni singola parola, ma soprattutto riesce a capire e vedere le mie sicurezze e debolezze per quelle che sono.
In ultimo, ma non certo per importanza, alla mia dolce e stupenda metà, Laura, che mi ha accompagnato in buona parte del percorso, che si è fatta carico di molti miei fardelli e ha sempre saputo alleggerirli, rendendo tutto molto semplice e magico, urlo mille e ancora mille grazie per il grande amore che ogni giorni mi dimostri. Ti amo!