



UNIVERSITÀ DEGLI STUDI DI FIRENZE
SCUOLA DI INGEGNERIA - DIPARTIMENTO DI INGEGNERIA
DELL'INFORMAZIONE

Tesi di Laurea Magistrale in Ingegneria Informatica

**PROGETTAZIONE E SVILUPPO DI COMPONENTI
PER LA PIATTAFORMA AIRQINO DEDICATA AL
MONITORAGGIO DELLA QUALITÀ DELL'ARIA**

Candidato
Edoardo D'Angelis

Relatori
Prof. Andrew D. Bagdanov
Prof. Pietro Pala

Correlatori
Dott. Walter Nunziati
Dott. Alice Cavaliere

Anno Accademico 2020/2021

Abstract

Air pollution is currently one of the main issues affecting urbanized areas worldwide. Local administrations monitor these harmful gases by means of reference monitoring stations provided by regional/national environmental protection agencies. These stations, however, have limitations due to coarse spatial coverage of the whole municipality, low time-frequency, and high costs. In this framework, the National Research Council of Italy (CNR-IBE) and the Tuscany Region Environmental Protection Agency (ARPAT) agreed to an initiative to create a low-cost network aimed at monitoring air quality over an Italian urban area. The rural town of Capannori, located in the Tuscany region (Italy), was chosen as a testing area since it lies within a critical area both affected by a variety of emission sources and weather conditions unfavourable to pollutant dispersion. The air quality analysis was carried out by means of several innovative low-cost stations named AIRQino, equipped with sensors for collecting air pollution (PM_{2.5}, PM₁₀, NO₂, O₃, CO, CO₂) and meteorological parameters (air temperature and relative humidity). Concentrations of PM were mainly considered in this work for providing indicative air quality measurements to supplement fixed measurements collected by the official urban monitoring network. This work, still ongoing, has two main objectives: (i) to show the robustness of AIRQino at measuring PM concentrations; (ii) to investigate the PM concentrations dynamics at

higher spatial and time scale distribution compared to the reference station.

Sommario

...

Indice

Abstract	i
Sommario	iii
1 Introduzione	1
1.1 Contesto	1
1.1.1 Descrizione del problema	1
1.1.2 Motivazioni	2
1.2 La piattaforma AirQino	2
1.2.1 Hardware dei sensori	2
1.2.2 Architettura e tecnologie	3
1.2.3 Progetti correlati	3
1.2.4 Progetti simili	4
2 Sviluppi tecnologici	6
2.1 Replica del database di produzione	6
2.1.1 Motivazioni	7
2.1.2 Streaming Replication	8
2.1.2.1 Preparazione del database primario	10
2.1.2.2 Configurazione della replica	12
2.1.2.3 Automazione con Docker	13

2.2	Ottimizzazione di query temporali	15
2.2.1	Motivazioni	15
2.2.2	Continuous aggregates	18
2.2.3	Risultati ottenuti	19
3	Calibrazione	22
3.1	Dati a disposizione	23
3.1.1	Dataset NO ₂	25
3.1.2	Dataset PM _{2.5} e PM ₁₀	27
3.1.3	Preprocessamento	29
3.1.3.1	Dataset ARPAT NO ₂	29
3.1.3.2	Dataset ARPAT PM _{2.5} e PM ₁₀	31
3.1.3.3	Dataset SMART16	33
3.1.3.4	Unione dei dataset	35
3.2	Regressione	36
3.2.1	Introduzione	37
3.2.2	Correlazione e coefficiente di determinazione	38
3.2.3	Analisi dei residui	42
3.2.3.1	Distribuzione degli errori	43
3.2.3.2	Indipendenza degli errori	44
3.2.3.3	Omogeneità della varianza dei residui	45
3.2.3.4	Influenza di outliers	46
3.2.4	Modelli di regressione	47
3.2.4.1	Regressione lineare	47
3.2.4.2	Regressione lineare robusta (Huber)	49
3.2.4.3	Regressione lineare avanzata	50
3.2.4.4	Regressione Ridge	52
3.2.4.5	Regressione Lasso	54

3.2.4.6	Regressione polinomiale	55
3.2.4.7	Regressione con Random Forest	56
3.2.4.8	Regressione con Gradient Boosting	57
3.2.4.9	Regressione con SVR	58
3.2.4.10	Regressione con KernelRidge	59
3.3	Esperimenti e risultati ottenuti	60
3.3.1	Risultati NO ₂	61
3.3.2	Risultati PM _{2.5}	66
3.3.3	Risultati PM ₁₀	71
3.4	Validazione	76
3.5	Discussione	79
4	Interfaccia di calibrazione	81
4.1	Motivazioni	81
4.2	Tecnologie	81
4.2.1	Backend	81
4.2.2	Frontend	81
4.3	Funzionamento	81
4.4	Autenticazione	82
4.5	CI e deploy automatico	82
Conclusioni e sviluppi futuri		84
Bibliografia		85

Capitolo 1

Introduzione

tesi realizzata in collaborazione con magenta e ibe cnr + foto ecc

1.1 Contesto

Il monitoraggio della qualità dell'aria è una delle attività più importanti per la tutela della salute pubblica. La qualità dell'aria può essere influenzata da molte sorgenti di emissione, tra cui le automobili, le centrali elettriche, gli impianti di riscaldamento e le fabbriche. I principali inquinanti atmosferici sono il biossido di zolfo, gli idrocarburi policiclici aromatici, il monossido di carbonio e gli ozono. Gli effetti dell'inquinamento atmosferico sulla salute sono molteplici e possono essere a breve o a lungo termine. I principali rischi sono l'asma, le malattie cardiovascolari, il cancro e le malattie respiratorie. Il monitoraggio della qualità dell'aria permette di individuare le sorgenti di emissione e di intervenire per ridurre l'inquinamento atmosferico.

1.1.1 Descrizione del problema

...

1.1.2 Motivazioni

...

1.2 La piattaforma AirQino

AirQino è una piattaforma di monitoraggio ambientale ad alta precisione, realizzata dal Consiglio Nazionale delle Ricerche (CNR) in collaborazione con TEA Group e Quanta Srl. [1] Il progetto nasce dall'esigenza di realizzare una rete di stazioni mobile per un monitoraggio più completo della qualità dell'aria in ambito urbano, in linea con la Direttiva 2008/50/EC, che riconosce e regolamenta l'importanza di misure aggiuntive rispetto a quelle delle stazioni fisse.

Nonostante infatti l'attività svolta da ARPA, a causa del numero limitato di stazioni e/o di sorgenti monitorate, ad oggi, la conoscenza sullo stato dell'inquinamento dell'aria da parte degli Enti Locali rimane molto limitata.

1.2.1 Hardware dei sensori

Per quanto riguarda la caratteristiche dei sensori, i sensori di tipo MOS sono costituiti da un film (credo allumina? Per fabbricare gli strati sensibili del film, si prepara una pasta viscosa: al materiale funzionale, sotto forma di polvere, viene aggiunta una miscela di agenti reologici in solventi volatili) depositato su una piastra di elementi riscaldanti la cui temperatura operativa è generalmente compresa tra 300 e 500°C. Di solito il materiale funzionale del film più adatto per la rilevazione di biossido di azoto è l'ossido di ferro e lantanio (LaFeO_3) che oltre ad avere una buona sensibilità agli ossidi di azoto ha una bassa sensibilità al monossido di carbonio. Per la rilevazione dell'ozono viene invece utilizzato triossido di tungsteno (WO_3). Questo ti-

po di materiale funzionale risulta molto sensibile ai gas ossidanti come O₃ e NO₂. Qualsiasi sia il materiale funzionale, il principio di funzionamento per tutti i MOS nella rilevazione di gas è quello di interagire con il gas presente all'interno dell'atmosfera tramite reazioni di ossidoriduzione, portando a un cambiamento di conduttività, che viene rilevato da un circuito apposito. Le variazioni della conduttività dei sensori è fortemente influenzata dalle variazioni di umidità e temperatura, come rilevato dalla letteratura sull'argomento [ref]. Nel caso dei sensori Mics che noi utilizziamo, il produttore non rilascia informazioni sull'influenza nella lettura dovuta alla temperatura/umidità ma che queste influiscono può essere ipotizzato come può essere ipotizzato che ci sia una influenza introdotta dalla temperatura nel circuito ADC del microcontrollore.

Questo segnale viene passato al convertitore analogico digitale del controllore che lo trasforma in counts (10 bit da 0 a 2 alla 10).

ossidoriduzione → piastra che si scalda a seconda dell'inquinante genera corrente

il segnale viene passato attraverso un convertitore analogico digitale e l'uscita è a 10 bit (questa unità la chiamo counts)

1.2.2 Architettura e tecnologie

...

1.2.3 Progetti correlati

...

1.2.4 Progetti simili

- **Airly** (<https://airly.org/>) è una piattaforma che consente di dividere informazioni ambientali in tempo reale, grazie alla quale è possibile monitorare la qualità dell'aria e i livelli di inquinamento;
- **Aqicn** (<https://aqicn.org>) è un progetto open source lanciato nel 2010 che consente di monitorare l'inquinamento atmosferico in tempo reale;
- **IQAir** (<https://aqicn.org>) è una società svizzera che produce e vende purificatori d'aria per uso residenziale e commerciale. La loro applicazione fornisce un rapporto in tempo reale sulla qualità dell'aria e previsione dell'inquinamento atmosferico;
- **Decentlab** (<https://decentlab.com>) è un'azienda svizzera che fornisce dispositivi e servizi di sensori wireless per soluzioni di monitoraggio distribuite ed economiche;
- **SMART Treedom** (<https://smart.treedom.net>) è il frutto dalla collaborazione tra Treedom e l'Istituto di Biometeorologia del Consiglio Nazionale delle Ricerche. La finalità del progetto è stata quella di prototipare un sistema integrato che possa essere modulato con diversi sensori in base al tipo di grandezza fisica che si vuole misurare e una tecnologia laser per la misura delle polveri sottili;
- **PlanetWatch** (<https://planetwatch.io>) è una piattaforma decentralizzata che consente di monitorare e proteggere il pianeta attraverso la condivisione di informazioni. Gli utenti possono condividere informazioni sull'ambiente, la sostenibilità e la responsabilità sociale;

- **HackAIR** (<https://hackair.eu>) è una piattaforma open source che consente ai cittadini di monitorare la qualità dell'aria nei propri quartieri. Gli utenti possono interagire con la piattaforma per segnalare la qualità dell'aria nel proprio quartiere, visualizzare i dati relativi alla qualità dell'aria e condividere informazioni e dati con altri utenti.

Capitolo 2

Sviluppi tecnologici

Questo capitolo riguarda i miglioramenti realizzati dal punto di vista tecnologico che sono andati direttamente ad impattare la piattaforma AirQino, migliorandone in un caso l'affidabilità dei dati e nell'altro i tempi di risposta del database per query particolarmente onerose.

2.1 Replica del database di produzione

Spesso fare analisi mediamente complesse sui dati contenuti in un database può comportare rallentamenti nei tempi di risposta. Se questi carichi risultano frequenti, il sistema può arrivare a bloccarsi e interrompere il servizio.

Una soluzione per risolvere questo problema è la creazione di una (o più) repliche del database primario. Nella replica, i dati e gli oggetti del database vengono copiati e distribuiti su un altro spazio fisico. Le operazioni onerose a questo punto possono essere fatte direttamente sulla replica che agisce come nodo secondario: in questo modo, il carico viene distribuito e non si intaccano le performance del database principale.

Il concetto di *replica* è diverso dal *mirroring*, in cui vengono create una o più copie di un database su diverse istanze del server, e funzionano come copie di riserva (e si attivano soltanto nel caso di guasto del nodo principale).

Un sistema di replica correttamente implementato può offrire diversi vantaggi, tra cui riduzione del carico (perchè i dati replicati possono essere distribuiti su più server), efficienza (i server offrono prestazioni migliori perchè meno gravati da query pesanti) e ridondanza (i dati sono raggiungibili da più indirizzi).

Di contro, questa tecnica comporta la necessità di mantenimento dei nodi secondari, spesso collocati su server diversi (con i costi a questi associati). Inoltre, replicate errate o non implementate in maniera corretta possono causare la mancata sincronizzazione tra i nodi, portando ad una perdita o incoerenza dei dati.

2.1.1 Motivazioni

La replica offre vantaggi principalmente legati alle prestazioni, disponibilità e sicurezza dei dati:

1. **Maggiore affidabilità:** tramite la replica del database viene garantita la disponibilità dei dati anche nel caso in cui una delle macchine presenti un guasto hardware. In questo caso, il sistema di gestione del database distribuito deve essere in grado di indirizzare gli utenti interessati ad uno degli altri nodi disponibili;
2. **Miglioramento delle prestazioni:** essendo i dati distribuiti su diverse istanze, accessi multipli non saturano i server. Questo aspetto risulta particolarmente importante per applicazioni che possono avere una grande quantità di richieste simultanee;

3. Maggiore sicurezza dei dati: Mentre in un sistema tradizionale i backup di un database (se effettuati) sono archiviati sullo stesso disco, con la replica del database vengono scritti su più server, aumentandone di fatto l'affidabilità e la ridondanza.

Esistono diverse tecniche di replicazione del database, che dipendono sia dalla tecnologia utilizzata (MySQL, Postgres) che dalla natura del database stesso (relazionale o non relazionale). Il database di AirQino fa uso di Timescale¹, basato su Postgres; una caratteristica di Postgres è la possibilità di replicazione con la tecnologia di **Streaming Replication**, descritta di seguito.

2.1.2 Streaming Replication

La streaming replication di PostgreSQL è una funzionalità che consente di replicare i dati in tempo reale da una istanza di PostgreSQL a un'altra. Questo significa che, se si modificano i dati in una delle istanze, questi saranno immediatamente replicati anche nell'altra istanza. Questa istanza database di replica di lettura ("standby" in termini PostgreSQL) è una replica fisica creata in modo asincrono dell'istanza del database primario.

PostgreSQL fa utilizzare un ruolo di "replica" (*replication role*) per eseguire la replica in streaming (il ruolo presenta dei privilegi ma non può essere utilizzato per modificare i dati, la replica infatti è di sola lettura).

La replica si basa sulle transazioni WAL (Write Ahead Log) e utilizza il protocollo TCP per garantire una connessione sicura tra i server e inviare in modo asincrono le modifiche al database via via che vengono effettuate.

¹Timescale: Time-series data simplified - <https://www.timescale.com>

È possibile promuovere una replica di lettura PostgreSQL a una nuova istanza database di origine. Una volta promossa, la replica smette di ricevere comunicazioni WAL e non è più un'istanza di sola lettura.

PostgreSQL salva le informazioni aggiornate del server primario in registro delle transazioni, noto come registro *write-ahead*, utile in preparazione per il ripristino da crash o il rollback. La replica in streaming funziona proprio trasferendo e applicando il WAL al server di replica in tempo reale.

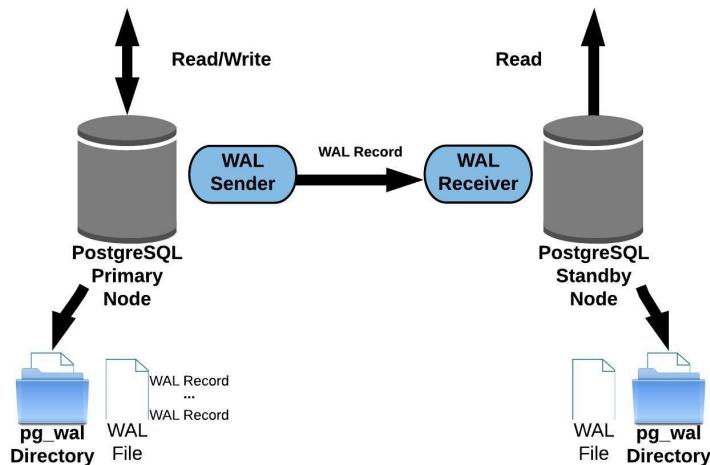


Figura 2.1: Streaming Replication di PostgreSQL

Fonte: <https://severalnines.com>

La replica in streaming può essere costruita in una configurazione 1:N, in cui è configurabile un solo server primario, ma è possibile configurare più server di replica (configurazione *multi-standby*). È anche possibile creare una configurazione a cascata in cui un server di replica si connette ad un altro server di replica.

Per la replica in streaming, è possibile scegliere se effettuare una replica sincrona o asincrona. La differenza tra replica sincrona e replica asincrona

sta nell'attesa o meno della risposta dal server di replica prima di completare l'elaborazione sul server primario:

- **Replica sincrona:** il server primario attende una risposta dal server di replica prima di completare un processo. In questo caso il tempo di risposta complessivo include anche il tempo di spedizione del registro;
- **Replica asincrona** (impostazione predefinita): il server primario completa un processo senza attendere una risposta dal server di standby. Pertanto, il tempo di risposta complessivo è più o meno lo stesso di quando non viene utilizzata la replica in streaming; in questa modalità il risultato aggiornato sul server primario potrebbe non essere immediatamente disponibile sul server di replica. [2]

Esistono di contro diverse limitazioni per le repliche con PostgreSQL:

- Ogni replica di lettura PostgreSQL è di sola lettura. Non è possibile creare una replica di lettura che sia scrivibile;
- Non è possibile creare una replica di lettura da un'altra replica di lettura a cascata;
- Gli utenti e i ruoli di accesso vengono rispecchiati dall'istanza primaria, il che significa che non è possibile utilizzare credenziali diverse o aggiungere utenti aggiuntivi soltanto alla replica;
- Non è possibile creare tabelle e viste nell'istanza di replica; in pratica dalla replica si possono solo eseguire query di tipo *SELECT*.

2.1.2.1 Preparazione del database primario

1. Per avviare la procedura si crea sul database un utente PostgreSQL con un ruolo adatto ad avviare la streaming replication:

```
1 SET password_encryption = 'scram-sha-256';
2 CREATE ROLE repuser WITH REPLICATION PASSWORD '  
SOME_SECURE_PASSWORD' LOGIN;
```

2. Aggiungere i seguenti parametri al file `/var/lib/postgresql/data/postgresql.conf`:

```
1 listen_addresses='*'
2 wal_level = replica
3 max_wal_senders = 2
4 max_replication_slots = 2
5 synchronous_commit = off
```

3. Aggiungere i seguenti parametri al file di configurazione `/var/lib/postgresql/data/pg_hba.conf` per configurare l'autenticazione basata su host in modo da accettare connessioni dalla replica²:

```
1 host    replication    repuser  <REPLICA_IP>/32      scram-sha-256
```

dove *repuser* è l'utente creato al passo 1 e *REPLICA_IP* è l'IP della macchina in cui si trova la replica;

4. Riavviare il database per applicare i cambiamenti;

5. Infine creare uno slot di replicazione sul database:

```
1 SELECT * FROM pg_create_physical_replication_slot('replica_1_slot');
```

²Dalla documentazione Streaming Replication: <https://www.postgresql.org/docs/current/warm-standby.html#STREAMING-REPLICATION>

2.1.2.2 Configurazione della replica

Di seguito sono elencati i passi da seguire per configurare la replica:

1. Stoppare l'istanza Postgres:

```
1 pg_ctl -D $PGDATA -m fast -w stop
```

2. Cancellare i contenuti della cartella *PGDATA*:

```
1 rm -rf $PGDATA/*
```

3. Avviare il backup del database primario:

```
1 pg_basebackup -h <PRIMARY_HOST> -p <PRIMARY_PORT> -D  
$PGDATA -U repuser -vP -R -W
```

Dove *PRIMARY_HOST* è l'IP della macchina in cui si trova il database primario, *PRIMARY_PORT* è la porta del database primario e *repuser* è l'utente di replicazione creato al passo 1.

Da notare che verrà chiesta in maniera interattiva la password di replicazione impostata in precedenza (*SOME_SECURE_PASSWORD*);

4. Infine avviare l'istanza Postgres:

```
1 pg_ctl -D $PGDATA -w start
```

A questo punto la replica dovrebbe essere funzionante e sincronizzata 1:1 in tempo reale con il database primario.

2.1.2.3 Automazione con Docker

L'intero setup (sia database primario che la replica) può essere automatizzato con Docker e docker-compose, in modo da far partire la sincronizzazione all'avvio del container. Poichè i passi per creare la replica richiedono di stoppare l'istanza Postgres, non si può eseguire all'interno del container Docker già avviato perchè si stopparebbe tutto il container. Per questo è necessario dare i comandi per la replica da uno script di *entrypoint* che viene eseguito prima di avviare il container:

1. Creare un Dockerfile da immagine Timescale (`timescaledb:latest`), con l'aggiunta di uno script *entrypoint*, così strutturato:

```
1 FROM timescale/timescaledb:latest-pg13
2 ADD replica.sh /docker-entrypoint-initdb.d/
```

2. Creare il file `replica.sh` che verrà eseguito tutte le volte che si avvia il database:

```
1 echo "Stopping Postgres instance..."
2 pg_ctl -D ${PGDATA} -m fast -w stop
3
4 echo "Clearing PGDATA folder..."
5 rm -rf ${PGDATA}
6
7 echo "Creating base backup..."
8 PGPASSWORD=${REPLICATION_PASSWORD} pg_basebackup -h ${REPLICATION_HOST} -p ${REPLICATION_PORT} -D ${PGDATA} -U ${REPLICATION_USER} -vP -R -w
9
10 echo "Restarting Postgres instance..."
11 pg_ctl -D ${PGDATA} -w start
```

Da notare che la flag `-W` di `pg_basebackup` chiede la password in maniera interattiva, che non funziona per script automatizzati. Come alternativa si può usare la flag `-w` (minuscola) e passare la password come variabile di ambiente chiamata `PGPASSWORD`;

3. Creare il file `docker-compose.yml`:

```
1 services:
2   replica:
3     build:
4       context: .
5       dockerfile: Dockerfile
6     environment:
7       # Cartella PDATA custom
8       PGDATA: /var/lib/postgresql/data/pgdata
9
10    # Parametri di replicazione
11    REPLICA_USER: repuser # Utente di replicazione impostato al punto 1
12    REPLICATION_HOST: x.x.x.x # IP del db primario
13    REPLICATION_PORT: x # Porta del db primario
14    REPLICATION_PASSWORD: SOME_SECURE_PASSWORD #
15      Password di replicazione impostata al punto 1
16
17 ports:
18   - 45432:5432
19
20 volumes:
21   - /var/replica-pg13-timescale:/var/lib/postgresql/data
```

4. Avviare il container con `docker-compose up`.

In questo modo la replica verrà sincronizzata con il database primario automaticamente all'avvio del container Docker. Allo stesso modo è possibile automatizzare con Docker anche il setup su database primario.

2.2 Ottimizzazione di query temporali

Ci sono molti vantaggi nell'aggregazione di dati:

- **Flessibilità:** aggregare dati in tempo reale in base a qualsiasi criterio desiderato;
- **Risparmio di tempo:** non è necessario eseguire query aggiuntive per ottenere informazioni aggregate in tempo reale;
- **Risparmio di spazio** i dati aggregati in tempo reale occupano meno spazio rispetto ai dati non aggregati.

I dati delle serie temporali però tendono a crescere molto rapidamente, e grandi volumi di dati possono rallentare quando si eseguono query con lo scopo di aggregare i dati (ad esempio per generare report o riepiloghi sull'andamento).

Per garantire efficienza e scalabilità con questa tipologia di dati, si rende quindi necessario garantire che le query su dati temporali abbiano un tempo di risposta costante, indipendentemente dalla quantità di dati e senza gravare sul database.

2.2.1 Motivazioni

La piattaforma AirQino (vedi 1.2) raccoglie dati emessi ogni minuto da decine di centraline in diverse località. Per memorizzare questi dati viene utilizzato un database Postgres con estensione Timescale, che offre delle funzionalità specifiche proprio per dati di tipo temporale.

Una delle funzionalità di AirQino consente di mostrare un grafico dell'andamento (su base media oraria) dell'ultima settimana per diverse grandezze

(temperatura, umidità, CO₂, NO₂, PM_{2.5}, PM₁₀ ecc..), come mostrato in figura 2.2.

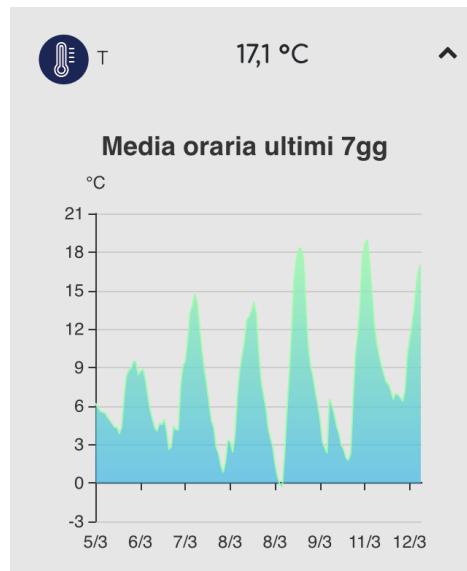


Figura 2.2: Grafico dell’andamento della temperatura nell’ultima settimana per una centralina AirQino.

Fonte: <https://airqino.magenta-lab.it>

Per calcolare la media oraria nell’ultima settimana, ad ogni richiesta sul database viene eseguito una query SQL di questo tipo (ridotta per semplicità):

```

1 SELECT time_bucket('1_hour', sd.data_acquired) AS bucket, avg(sd.float_value)
2 FROM station_data sd
3 WHERE sd.data_acquired > NOW() - INTERVAL '7_days'
4 AND sd.sensor_id = 29510691 /* id centralina */
5 ORDER BY bucket DESC;
```

La query fa uso della funzione *time_bucket* di Timescale, che consente di partizionare la tabella con i dati dei sensori minuto per minuto direttamente

in fasce orarie. L'utilizzo combinato con la funzione `avg()` va poi a prendere la media del valore su tutta l'ora.

Questo però significa che se si desidera eseguire questa query più di una volta, il database deve eseguire la scansione dell'intera tabella e ricalcolare la media ogni volta. Nella maggior parte dei casi, tuttavia, i dati nella tabella non sono cambiati in modo significativo, quindi non è necessario eseguire la scansione dell'intero set di dati.

La figura 2.3 mostra i tempi di risposta a questa query per tutte le centraline AirQino prima di aver applicato l'ottimizzazione.

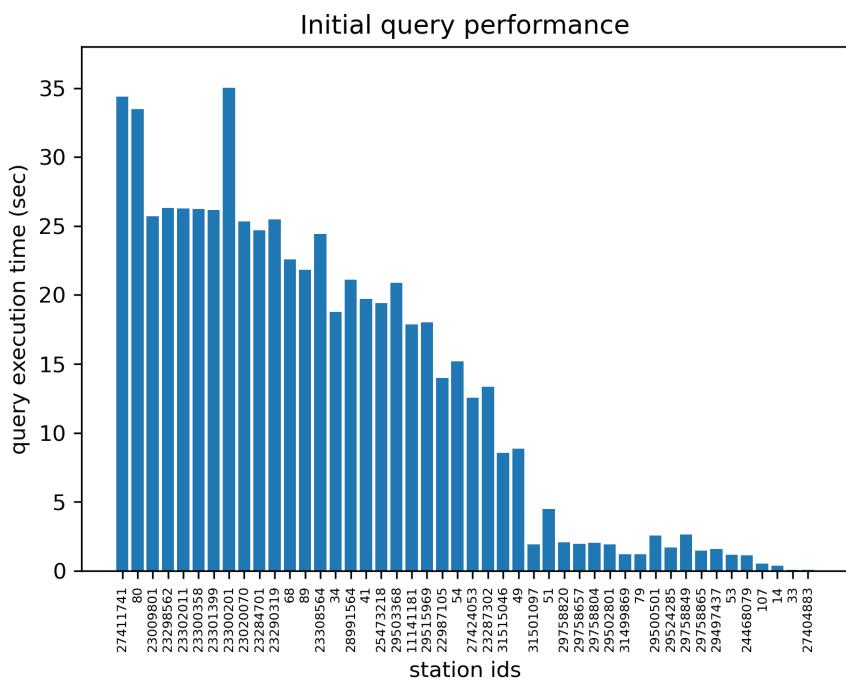


Figura 2.3: Tempi di risposta per query temporali sulle centraline AirQino prima dell'ottimizzazione (media su 10 iterazioni)

Come si può vedere, per le centraline più attive la query per recuperare la media oraria impiega fino a 35 secondi, un tempo molto elevato.

Per rendere più veloce l'aggregazione dei dati, Timescale mette a disposizione una funzionalità chiamata **continuous aggregates** (*aggregati continui*), spiegata di seguito.

2.2.2 Continuous aggregates

I continuous aggregate sono una funzionalità integrata in TimescaleDB che consente di aggregare i dati in tempo reale, senza la necessità di eseguire query aggiuntive. Funzionano in maniera simile alle viste materializzate di PostgreSQL, con la differenza che non necessitano di essere aggiornati manualmente ogni volta che un nuovo dato viene inserito nella tabella. Infatti, la riaggregazione viene eseguita automaticamente in background tramite una **refresh policy** definita dall'utente in fase di creazione della vista.

L'utilizzo dei countinuous aggregates offre diversi vantaggi:

- Miglioramento delle **performance**, infatti non è più necessario scansioneare tutte le volte la tabella con i dati raw ma è sufficiente leggere questi risultati precalcolati;
- Funzionalità avanzate, come la possibilità di salvare i dati raw solo per un periodo di tempo limitato, continuando però a mantenere i dati aggregati. Così facendo si hanno dei dati riassuntivi per eventi che sono molto indietro nel tempo, mentre per gli eventi più recenti si continua ad avere tutti i dati raccolti, portando and un grosso **risparmio di spazio occupato**.

Di contro, lo svantaggio di questa soluzione è che i dati non sono aggiornati ad ogni INSERT effettuata, ma solo ad un intervallo di tempo specificato dall'utente. Questo significa che in fase di lettura i dati più recenti non sono presenti nelle viste materializzate. Per limitare questo problema si potrebbe

pensare di diminuire l'intervallo di tempo necessario per il refresh dei dati, ma questo porterebbe ad un aumento del carico computazionale. C'è quindi un compromesso tra la velocità di lettura e l'aggiornamento dei dati. [3]

Timescale inoltre introduce il concetto di *hypertable*, normali tabelle SQL ma con partizionamento automatico su base temporale.

2.2.3 Risultati ottenuti

Di seguito sono riportati i passi eseguiti sul database di produzione AirQino per l'attivazione del meccanismo di continuous aggregates sulla tabella con i dati dei sensori (*station_data*):

1. Creare la *hypertable* a partire dalla tabella originale, migrando tutti i dati:

```
1 SELECT create_hypertable('station_data', 'data_acquired', migrate_data =>
    true);
```

Dove *data_acquired* è la colonna temporale della tabella, in cui vengono salvati data e ora della misurazione. Questa operazione potrebbe impiegare molto tempo, in base alla quantità di dati nella tabella. Sul database di produzione AirQino ha richiesto circa 36 minuti.

2. Creare la vista materializzata per il calcolo delle medie orarie:

```
1 CREATE MATERIALIZED VIEW station_data_hourly_avg
2 WITH (timescaledb.continuous) AS
3 SELECT time_bucket('1_hour', sd.data_acquired) AS bucket,
4     station_id,
5     sensor_id,
6     avg(sd.float_value)
7 FROM station_data sd
```

```
8 GROUP by bucket, station_id, sensor_id;
```

Questa operazione sul database di produzione AirQino ha richiesto circa 5 minuti.

3. Creare una refresh policy adeguata:

```
1 SELECT add_continuous_aggregate_policy('station_data_hourly_avg',
2 start_offset => INTERVAL '7_days',
3 end_offset => INTERVAL '1_hour',
4 schedule_interval => INTERVAL '1_hour');
```

Dove *start_offset* indica quanto indietro nel tempo vado a calcolare l'aggregato, *end_offset* indica fino a quando arrivo e *schedule* indica ogni quanto voglio refreshare la vista. In questo caso voglio refreshare ogni ora i dati da una settimana fa fino a un'ora fa. Per ragioni di performance conviene sempre escludere l'ultimo bucket (in questo caso l'ultima ora di dati arrivati).

Una volta creata la vista, è possibile ricreare la SELECT originale per la media degli ultimi 7 giorni prendendo i dati da questa nuova tabella:

```
1 SELECT sd.avg
2 FROM station_data_hourly_avg sd
3 WHERE bucket > NOW() - INTERVAL '7_days'
4 AND sd.sensor_id = 29510691 /* id centralina */
5 ORDER BY bucket DESC;
```

Questa nuova query è risultata molto più performante rispetto alla precedente, perchè le medie orarie sono di fatto già precalcolate. I risultati sulle centraline AirQino sono riportati di seguito in figura 2.4.

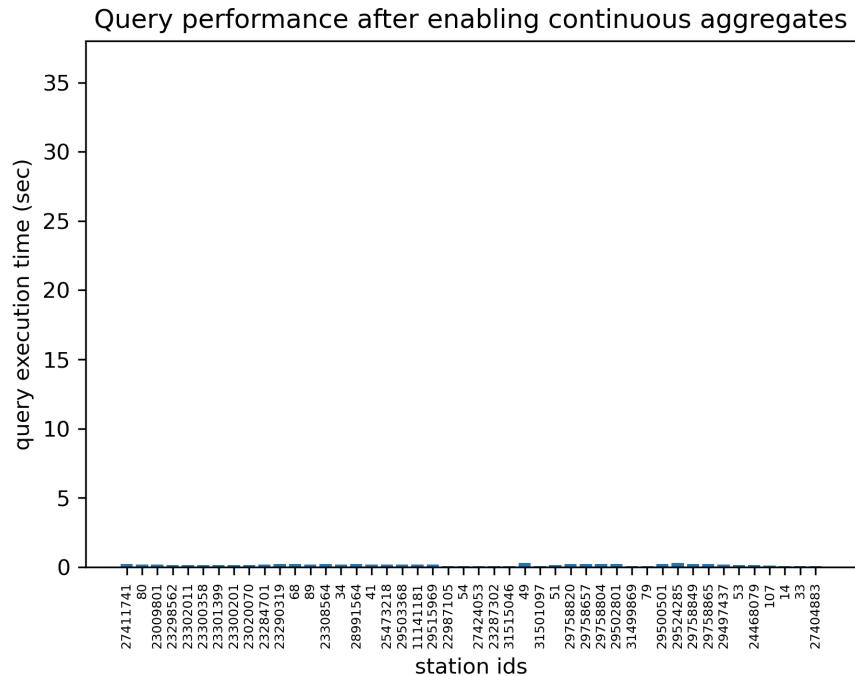


Figura 2.4: Tempi di risposta per query temporali sulle centraline AirQino dopo l'ottimizzazione (media su 10 iterazioni)

Da cui si può notare che con la nuova query, ottimizzata con continuous aggregates, i tempi di risposta risultano costanti per tutte le centraline indipendentemente dalla quantità di dati, garantendo scalabilità ed efficienza.

Capitolo 3

Calibrazione

Questo capitolo riguarda la parte di tesi incentrata sulla calibrazione delle centraline AirQino (1.2). Nella sezione 3.1 vengono presentati i dataset a disposizione, la loro struttura e il lavoro di preprocessamento fatto.

La sezione 3.2 racchiude una breve panoramica teorica sui concetti di regressione, correlazione, metriche (coefficiente di correlazione, coefficiente di determinazione, scarto quadratico medio), analisi dei residui e modelli di regressione (sia lineare che non lineari).

La sezione 3.3 elenca gli esperimenti svolti per ciascun inquinante (NO_2 , $\text{PM}_{2.5}$ e PM_{10}) e presenta i risultati ottenuti (con analisi dei residui e confrontando i vari modelli, sia su tutto l'anno che mese per mese), mentre in 3.4 viene presentato il lavoro svolto per la validazione del modello di regressione migliore su un set di dati non visto in precedenza.

Infine, la sezione 3.5 riassume tutto il lavoro svolto e fornisce un'analisi qualitativa dei risultati ottenuti.

3.1 Dati a disposizione

I dataset messi a disposizione sono due:

- Dataset delle misurazioni di concentrazione di NO_2 nell'aria relative alla centralina SMART16 AirQino, da confrontare con i dati NO_2 ARPAT¹ della stazione Capannori (Lucca);
- Dataset delle misurazioni di concentrazione di $\text{PM}_{2.5}$ e PM_{10} nell'aria relative alla centralina SMART16 AirQino, da confrontare con i dati $\text{PM}_{2.5}$ e PM_{10} ARPAT della stazione Capannori (Lucca).

In entrambi i casi la centralina SMART16 è stata in co-locazione con la stazione ARPAT di Capannori per tutto periodo di interesse.



Figura 3.1: Una centralina AirQino

Fonte: <https://airqino.magentaLab.it>

¹Agenzia regionale per la protezione ambientale della Toscana



Figura 3.2: Centralina ARPAT in sede Capannori (provincia di Lucca)

Fonte: <http://arpat.toscana.it>

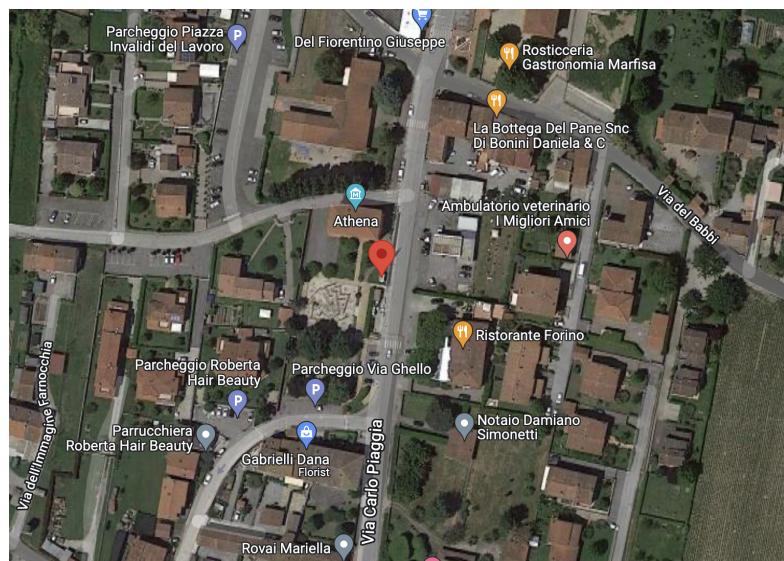


Figura 3.3: Posizione della centralina SMART16 (AirQino)
e ARPAT (Capannori) a Lucca

3.1.1 Dataset NO₂

Il dataset di misurazioni NO₂ comprende sia i dati della centralina AirQino SMART16 che i dati di ARPAT (Capannori). Ci sono però delle differenze sostanziali tra i due set di dati:

	Periodo	Unità	Frequenza dati
SMART16	01/01/2020 - 31/12/2020	<i>counts</i>	ogni 1/2 minuti
ARPAT	01/01/2020 - 31/12/2020	µg/m ³	medie giornaliere

Tabella 3.1: Differenze tra i dati di SMART16 e ARPAT (per NO₂)

Da notare che le centraline AirQino misurano la concentrazione di NO₂ con il sensore **MiCS-2714** (come già accennato in 1.2.1) che fornisce output in *counts* (unità di misura del segnale convertito da analogico a digitale, con uscita a 10 bit). Questo significa che sarà compito della fase di calibrazione convertire l'output direttamente in unità ingegneristica (in questo caso µg/m³).

Nella figura 3.4 sono riportate la frequenza di misurazione (intesa come il numero di misurazioni effettuate ogni ora) e l'andamento della concentrazione di NO₂ nell'aria (in *counts*) per come sono state misurate dalla centralina SMART16 nel periodo di interesse (01/01/2020 - 31/12/2020).

La figura 3.5 invece riporta l'andamento della concentrazione di NO₂ nell'aria (in µg/m³) misurato dalla stazione ARPAT di Capannori nello stesso periodo.

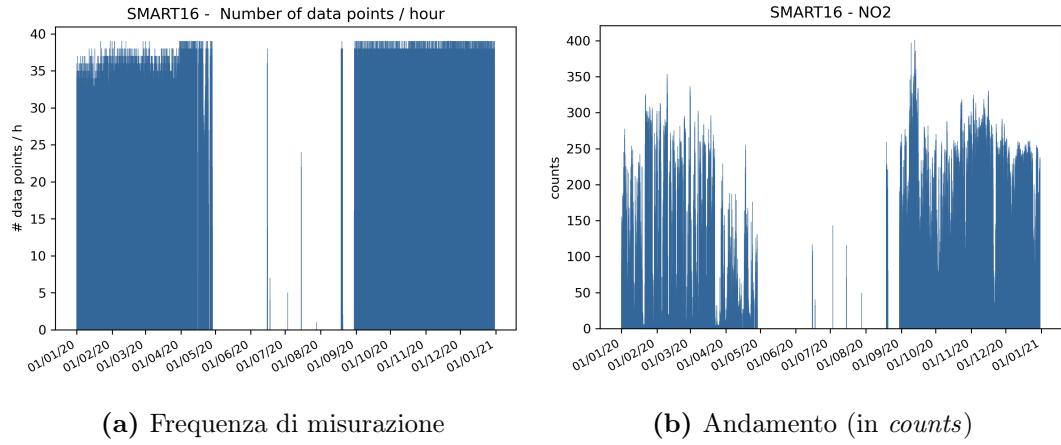


Figura 3.4: Frequenza di misurazione e andamento NO₂ (SMART16)
nel periodo 01/01/2020 - 31/12/2020

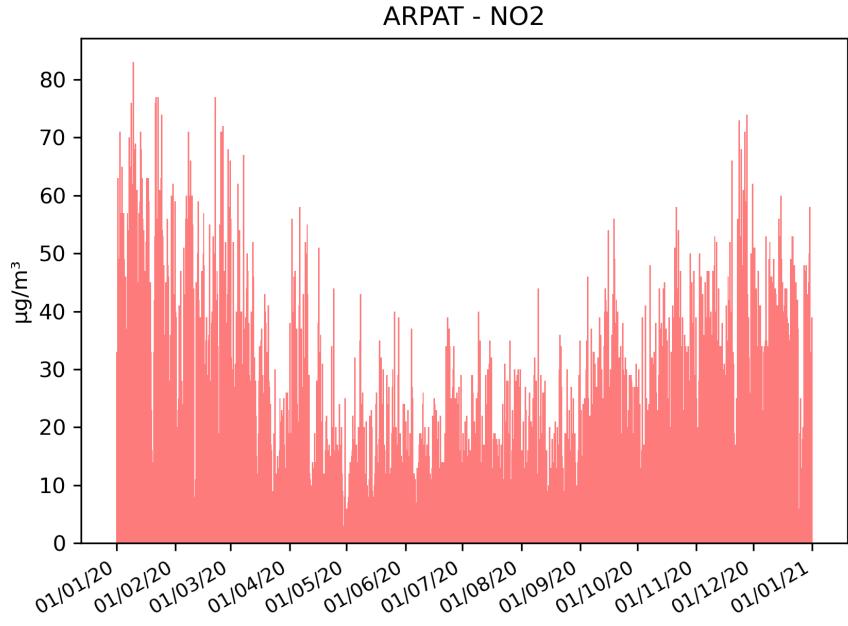


Figura 3.5: Andamento NO₂ (in $\mu\text{g}/\text{m}^3$) misurato dalla stazione ARPAT
di Capannori nel periodo 01/01/2020 - 31/12/2020

3.1.2 Dataset PM_{2.5} e PM₁₀

Il dataset di misurazioni PM_{2.5} e PM₁₀, che mette a confronto i dati della centralina AirQino SMART16 e i dati di ARPAT (Capannori), risulta invece così strutturato:

	Periodo	Unità	Frequenza dati
SMART16	01/09/2020 - 31/08/2021	µg/m ³	ogni 1/2 minuti
ARPAT	01/09/2020 - 31/08/2021	µg/m ³	medie ogni 8h

Tabella 3.2: Differenze tra i dati di SMART16 e ARPAT (per PM_{2.5} e PM₁₀)

Il sensore utilizzato dalle centraline SMART è il **SDS011** (vedi 1.2.1) che ha la caratteristica di fornire l'uscita direttamente in unità ingegneristica (µg/m³), quindi in questo caso non c'è bisogno di convertire l'unità di misura nella fase di calibrazione (a differenza di quanto accade con NO₂).

Nella figura 3.6 è riportata la frequenza di misurazione (numero di misurazioni effettuate ogni ora) di PM_{2.5} e PM₁₀ nell'aria (in µg/m³) per la centralina SMART16 nel periodo di interesse (01/09/2020 - 31/08/2021).

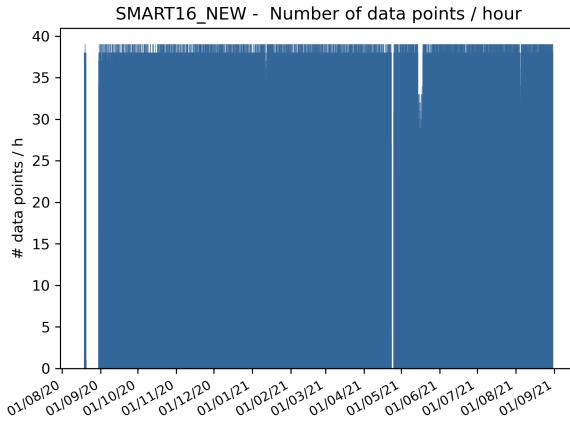


Figura 3.6: Frequenza di misurazione PM_{2.5} e PM₁₀ (SMART16)

Nella figura 3.7 sono riportati gli andamenti della concentrazione di PM_{2.5} e PM₁₀ nell'aria (in $\mu\text{g}/\text{m}^3$) per come sono state misurate dalla centralina SMART16 nel periodo di interesse (01/09/2020 - 31/08/2021).

La figura 3.15 invece riporta gli andamenti della concentrazione di PM_{2.5} e PM₁₀ nell'aria (in $\mu\text{g}/\text{m}^3$) misurato dalla stazione ARPAT di Capannori nello stesso periodo.

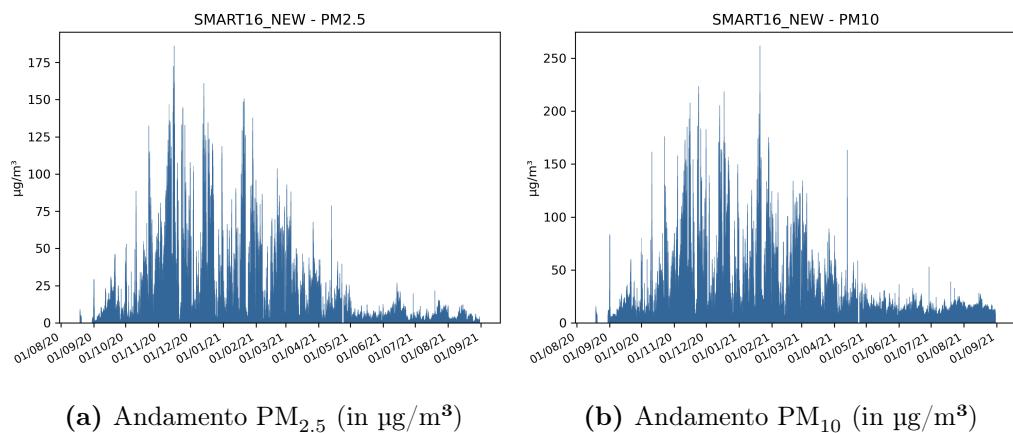


Figura 3.7: Andamento PM_{2.5} e PM₁₀ (SMART16)

nel periodo 01/09/2020 - 31/08/2021

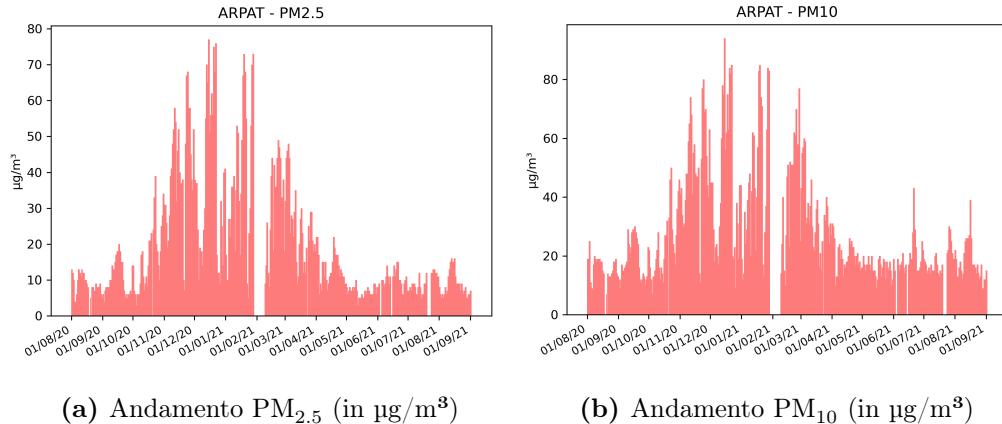


Figura 3.8: Andamento PM_{2.5} e PM₁₀ (ARPAT) nello stesso periodo

3.1.3 Preprocessamento

Per facilitare il caricamento e l'elaborazione, si è resa necessaria una fase iniziale di preprocessamento dei dati, descritta di seguito.

3.1.3.1 Dataset ARPAT NO₂

Il dataset originale NO₂, fornito da ARPAT, consiste in un file csv da 8785 righe, con encoding ISO-8859-1, valori separati da punto e virgola (;), e strutturato come in figura 3.9. In particolare:

- La data è in formato AAAAMMGG (colonna 'DATA');
- L'ora risulta in formato intero (colonna 'ORA FINE MISURA', con valori da 1 a 24, dove 24 indica le 00:00);
- Data e ora sono da considerarsi con fuso orario locale;
- La colonna VALIDITÀ indica se la misurazione è valida oppure no;
- La media oraria è riportata in $\mu\text{g}/\text{m}^3$.

LU-CAPANNORI_NO2_2020						
STAZIONE	PARAMETRO	DATA (formato AAAAMMGG)	ORA FINE MISURA (solare)	MEDIA ORARIA IN $\mu\text{g}/\text{m}^3$ A 20 °C	VALIDITA'	
LU-CAPANNORI	NO2	20200101	1	33	1	
LU-CAPANNORI	NO2	20200101	2	NaN		0
LU-CAPANNORI	NO2	20200101	3	28	1	
LU-CAPANNORI	NO2	20200101	4	25	1	
LU-CAPANNORI	NO2	20200101	5	24	1	
LU-CAPANNORI	NO2	20200101	6	22	1	
LU-CAPANNORI	NO2	20200101	7	21	1	
LU-CAPANNORI	NO2	20200101	8	20	1	
LU-CAPANNORI	NO2	20200101	9	19	1	
LU-CAPANNORI	NO2	20200101	10	27	1	
LU-CAPANNORI	NO2	20200101	11	31	1	
LU-CAPANNORI	NO2	20200101	12	32	1	
LU-CAPANNORI	NO2	20200101	13	31	1	
LU-CAPANNORI	NO2	20200101	14	26	1	
LU-CAPANNORI	NO2	20200101	15	25	1	
LU-CAPANNORI	NO2	20200101	16	25	1	
LU-CAPANNORI	NO2	20200101	17	49	1	
LU-CAPANNORI	NO2	20200101	18	63	1	
LU-CAPANNORI	NO2	20200101	19	52	1	

Figura 3.9: Struttura del dataset originale NO₂ fornito da ARPAT

In questa fase sono state effettuate le seguenti modifiche:

- Data e ora sono state unite in una singola colonna;
- Data e ora sono state convertite in formato standard UTC² per confor-marsi al dataset di AirQino;
- I dati non validi sono stati scartati;
- Le colonne sono state rinominate per semplicità ('data' per la data e 'avg' per il valore di NO₂).

Il risultato è un file csv di dimensioni ridotte che si presenta come riportato in figura 3.10:

²Il tempo coordinato universale o tempo civile, abbreviato con la sigla UTC, è il fuso orario scelto come riferimento globale, a partire dal quale sono calcolati tutti i fusi orari del mondo.

data	avg
2020-01-01 00:00:00+00:00	33.0
2020-01-01 02:00:00+00:00	28.0
2020-01-01 03:00:00+00:00	25.0
2020-01-01 04:00:00+00:00	24.0
2020-01-01 05:00:00+00:00	22.0
2020-01-01 06:00:00+00:00	21.0

Figura 3.10: Struttura del dataset ARPAT NO₂ processato

3.1.3.2 Dataset ARPAT PM_{2.5} e PM₁₀

Il dataset originale PM_{2.5} e PM₁₀, fornito da ARPAT, consiste in un singolo file csv da 33.674 righe con encoding UTF-8, valori separati da virgola (,), e strutturato come in figura 3.11. In particolare:

- Ci sono dati dal 18/01/2018 al 20/11/2021, ma per questo lavoro è stato considerato solo il periodo 01/09/2020 - 31/08/2021;
- La data è in formato gg/mm/aaaa (colonna 'DATA');
- L'ora è in formato intero (colonna 'ORA' con valori da 1 a 24, dove 24 indica le 00:00);
- Data e ora sono da considerarsi con fuso orario locale;
- I valori di PM_{2.5} e PM₁₀ sono riportati rispettivamente nelle colonne 'PM2.5_LU-CAPANNORI' e 'PM10_LU-CAPANNORI';

- I dati sono riportati come medie orarie, ma di fatto rappresentano medie ogni 8 ore.

LU-CAPANNORI_PM_Dati_Orari			
DATA	ORA	PM10_LU-CAPANNORI	PM2.5_LU-CAPANNORI
18/1/2018	1	34	24
18/1/2018	2	34	24
18/1/2018	3	34	24
18/1/2018	4	34	24
18/1/2018	5	34	24
18/1/2018	6	34	24
18/1/2018	7	34	24
18/1/2018	8	34	24
18/1/2018	9	34	24
18/1/2018	10	34	24

Figura 3.11: Struttura del dataset originale ARPAT PM_{2.5} e PM₁₀

Per questo dataset sono state effettuate le seguenti modifiche:

- Data e ora sono state unite in una singola colonna;
- Data e ora sono state convertite in formato standard UTC per conformati al dataset di AirQino;
- I dati non validi sono stati scartati;
- Le colonne sono state rinominate per semplicità ('data' per la data, 'pm2.5' per i valori di PM_{2.5} e 'pm10' per i valori di PM₁₀);
- I dati sono stati ricampionati e salvati come medie ogni otto ore.

Il risultato è un file csv di 4211 righe e che si presenta come riportato in figura 3.12:

data	pm10	pm2.5
2018-01-17 21:00:00+00:00	34.0	24.0
2018-01-18 05:00:00+00:00	34.0	24.0
2018-01-18 13:00:00+00:00	34.0	24.0
2018-01-18 21:00:00+00:00	35.25	26.5
2018-01-19 05:00:00+00:00	36.0	28.0
2018-01-19 13:00:00+00:00	36.0	28.0
2018-01-19 21:00:00+00:00	37.875	29.25

Figura 3.12: Struttura del dataset ARPAT PM_{2.5} e PM₁₀ processato con ricampionamento a 8 ore

3.1.3.3 Dataset SMART16

Il dataset originale per la centralina SMART16 di AirQino consiste in due file csv (uno di 201.279 righe per NO₂, e l’altro di 324.431 righe per PM_{2.5} e PM₁₀), strutturati rispettivamente come in figura 3.13 e 3.14. In particolare:

- Nel primo ci sono dati dal 01/01/2020 al 31/12/2020, nel secondo invece dal 18/08/2020 al 30/08/2021.
- Data e ora sono già in formato standard UTC;
- I valori di NO₂ sono riportati nella colonna ’no2’;
- I valori di PM_{2.5} e PM₁₀ sono riportati rispettivamente nelle colonne ’pm2_5’ e ’pm10’;
- In entrambi i file sono riportate anche le coordinate inviate dalla centralina al momento della misurazione (colonne ’long’ e ’lat’);

- In entrambi i file i dati sono riportati con frequenza di 1/2 minuti.

SMART16														
	long	lat	data		tair	rad	co2	pm2_5	pm10	o3	no2	co	voc	ds18
0	10.577585	43.80190666666667	2020-01-01 00:00:02	2.8	97.6	458	92	71.0	352	212	164	444	15.3	
1	10.577585	43.80190666666667	2020-01-01 00:01:36	2.8	97.9	458	101	78.0	354	212	169	449	15.29	
2	10.577585	43.80190666666667	2020-01-01 00:03:10	2.9	98.1	457	111	82.0	356	208	165	443	15.28	
3	10.577585	43.80190666666667	2020-01-01 00:04:44	2.9	98.2	456	111	81.0	352	199	163	438	15.25	
4	10.577585	43.80190666666667	2020-01-01 00:06:18	3.0	98.2	456	102	80.0	349	191	162	436	15.26	
5	10.577585	43.80190666666667	2020-01-01 00:07:52	3.0	98.0	456	113	83.0	349	195	164	438	15.25	
6	10.577585	43.80190666666667	2020-01-01 00:09:26	3.0	97.6	456	105	79.0	349	199	164	439	15.26	
7	10.577585	43.80190666666667	2020-01-01 00:11:00	3.0	97.3	456	96	74.0	346	191	162	434	15.27	
8	10.577585	43.80190666666667	2020-01-01 00:12:34	3.0	97.0	452	89	66.0	340	174	161	430	15.26	
9	10.577585	43.80190666666667	2020-01-01 00:14:08	3.0	96.7	452	95	69.0	337	166	160	428	15.25	
10	10.577585	43.80190666666667	2020-01-01 00:15:42	3.0	96.4	452	93	69.0	336	170	160	430	15.25	

Figura 3.13: Struttura del dataset originale SMART16 per NO₂

SMART16_new													
data	long	lat	tair	rad	co2	pm10	pm2_5	o3	no2	ds18			
2020-08-18 21:28:20	10.5728716666667	43.83988	21.2	98.4	501	17	8.0	352	310	27.67			
2020-08-18 21:29:54	10.5728716666667	43.83988	21.3	98.4	476	14	9.0	328	223	29.57			
2020-08-18 21:31:26	10.5728716666667	43.83988	21.3	98.4	471	14	9.0	315	208	30.52			
2020-08-18 21:33:00	10.5728716666667	43.83988	21.3	98.4	470	14	9.0	308	207	31.05			
2020-08-18 21:34:34	10.5728716666667	43.83988	21.3	98.4	468	15	9.0	302	205	31.4			
2020-08-18 21:36:08	10.5728716666667	43.83988	21.4	98.4	468	19	10.0	301	211	31.7			
2020-08-18 21:37:42	10.5728716666667	43.83988	21.4	98.4	468	17	10.0	300	213	31.74			
2020-08-18 21:39:16	10.5728716666667	43.83988	21.4	98.4	472	19	11.0	300	219	31.81			
2020-08-18 21:40:50	10.5728716666667	43.83988	21.5	98.4	472	18	10.0	299	217	31.91			
2020-08-18 21:42:24	10.5728716666667	43.83988	21.5	98.4	476	18	12.0	299	221	32.04			

Figura 3.14: Struttura del dataset originale SMART16 per PM_{2.5} e PM₁₀

Per questi due dataset sono state effettuate le seguenti modifiche:

- I dati del dataset NO₂ sono stati ricampionati a medie orarie;
- I dati del dataset PM_{2.5} e PM₁₀ sono stati ricampionati a otto ore;
- I dati non validi sono stati scartati.

Di seguito sono riportati i risultati del preprocessamento dei due dataset:

data	no2	data	pm2_5	pm10
2020-01-01 00:00:00+00:00	155.556	2020-08-30 05:00:00+00:00	1.54	7.92
2020-01-01 01:00:00+00:00	92.967	2020-08-30 13:00:00+00:00	2.03	9.434
2020-01-01 02:00:00+00:00	77.057	2020-08-30 21:00:00+00:00	3.269	11.192
2020-01-01 03:00:00+00:00	52.618	2020-08-31 05:00:00+00:00	2.551	8.919
2020-01-01 04:00:00+00:00	68.706	2020-08-31 13:00:00+00:00	2.607	6.227
2020-01-01 05:00:00+00:00	68.576	2020-08-31 21:00:00+00:00	20.698	48.196
2020-01-01 06:00:00+00:00	90.818	2020-09-01 05:00:00+00:00	9.533	17.854
		2020-09-01 13:00:00+00:00	1.076	5.686
		2020-09-01 21:00:00+00:00	905	4.102

(a) Dataset NO₂ processato(b) Dataset PM_{2.5} e PM₁₀ processato**Figura 3.15:** Struttura dei dataset SMART16 processati

con ricampionamento a una e otto ore

3.1.3.4 Unione dei dataset

In seguito, per facilitare l'elaborazione dei dati e l'applicazione delle tecniche di regressione (3.2), i dataset SMART e ARPAT (sia NO₂ che PM_{2.5} e PM₁₀) sono stati uniti in un unico dataset basandosi sul valore comune (colonna 'data'). I risultati di questa unione sono riportati di seguito (figure 3.16 e 3.17) e rappresentano i dataset finali utilizzati nella fase di sperimentazione (3.3).

data	airqino_no2	arpat_no2
2020-01-01 00:00:00+00:00	155.556	33.0
2020-01-01 02:00:00+00:00	77.057	28.0
2020-01-01 03:00:00+00:00	52.618	25.0
2020-01-01 04:00:00+00:00	68.706	24.0
2020-01-01 05:00:00+00:00	68.576	22.0
2020-01-01 06:00:00+00:00	90.818	21.0
2020-01-01 07:00:00+00:00	105.182	20.0
2020-01-01 08:00:00+00:00	148.182	19.0
2020-01-01 09:00:00+00:00	113.419	27.0

Figura 3.16: Struttura del dataset finale per NO₂ (SMART16 vs ARPAT)

data	airqino_pm2.5	airqino_pm10	arpat_pm2.5	arpat_pm10
2020-08-19 21:00:00+00:00	3.595	6.619	5.0	7.0
2020-08-30 05:00:00+00:00	1.54	7.92	6.0	14.0
2020-08-30 13:00:00+00:00	2.03	9.434	6.0	14.0
2020-08-30 21:00:00+00:00	3.269	11.192	5.25	13.25
2020-08-31 05:00:00+00:00	2.551	8.919	5.0	13.0
2020-08-31 13:00:00+00:00	2.607	6.227	5.0	13.0
2020-08-31 21:00:00+00:00	20.698	48.196	5.75	12.25
2020-09-01 05:00:00+00:00	9.533	17.854	6.0	12.0
2020-09-01 13:00:00+00:00	1.076	5.686	6.0	12.0
2020-09-01 21:00:00+00:00	905	4.102	4.5	9.75
2020-09-02 05:00:00+00:00	728	4.781	4.0	9.0
2020-09-02 13:00:00+00:00	713	4.976	4.0	9.0
2020-09-02 21:00:00+00:00	1.692	5.268	6.25	11.25
2020-09-03 05:00:00+00:00	1.051	5.068	7.0	12.0

Figura 3.17: Struttura del dataset finale per PM (SMART16 vs ARPAT)

3.2 Regressione

Nella statistica applicata si osserva (o si ipotizza) l'esistenza di relazioni fra due o più grandezze. Sorge allora il problema di determinare una funzione che, in base ai dati ricavati mediante esperimenti o rilevazioni statistiche, rappresenti questi relazioni permettendo di analizzare meglio i fenomeni osservati.

Con il termine regressione si intende proprio una tecnica statistica che serve a stimare la relazione esistente tra due o più variabili.

3.2.1 Introduzione

Limitando lo studio a problemi che stabiliscono relazioni fra due sole variabili, si tratta, partendo dalle coppie (x_i, y_i) di dati rilevati, di determinare una funzione $y = f(x)$ che rappresenti la relazione.

Per fare questo si può procedere in due modi:

- determinare una funzione che assuma esattamente i valori (x_i, y_i) rilevati;
- determinare una funzione che si accosti il più possibile ai punti (x_i, y_i) .

La prima opzione, ovvero la ricerca di una funzione (generalmente espressa da un polinomio) che passi esattamente per i punti (x_i, y_i) è piuttosto laboriosa. Nelle applicazioni statistiche si preferisce invece determinare la funzione il cui grafico si avvicini il più possibile ai punti rilevati.

Osservando l'andamento del fenomeno si sceglie il tipo di funzione interpolatrice: lineare, quadratica, esponenziale, ecc. e quindi si procede alla determinazione dei parametri, ossia delle costanti che compaiono nella funzione scelta in modo che sia soddisfatta una condizione di accostamento prefissata.

Per conseguire questo scopo il metodo più utilizzato è il metodo dei **mimi quadrati**, che costituisce un'applicazione della ricerca del minimo di una funzione di più variabili mediante gli strumenti dell'analisi infinitesimale.

Considerate due variabili X e Y sulle quali vengono effettuate n rilevazioni:

$$(x_1, y_1), (x_2, y_2), \dots, (x_i, y_i), \dots, (x_n, y_n)$$

Sia $y = f(x; a, b, c, \dots, k)$ la funzione interpolatrice scelta. Siano inoltre \hat{y}_i valori predetti sulla curva corrispondenti ai valori x_i rilevati.

La condizione di accostamento data dal metodo dei minimi quadrati è quella di determinare i valori dei parametri in modo che sia minima la somma dei quadrati delle differenze fra i valori osservati y_i e i valori predetti \hat{y}_i (figura 3.18), ovvero:

$$\varphi(a, b, c, \dots, k) = \sum_{i=1}^n [y_i - f(x_i; a, b, c, \dots, k)]^2$$

dove i valori x_i e y_i sono noti, mentre sono incogniti i parametri a, b, c, \dots, k della funzione. [4]

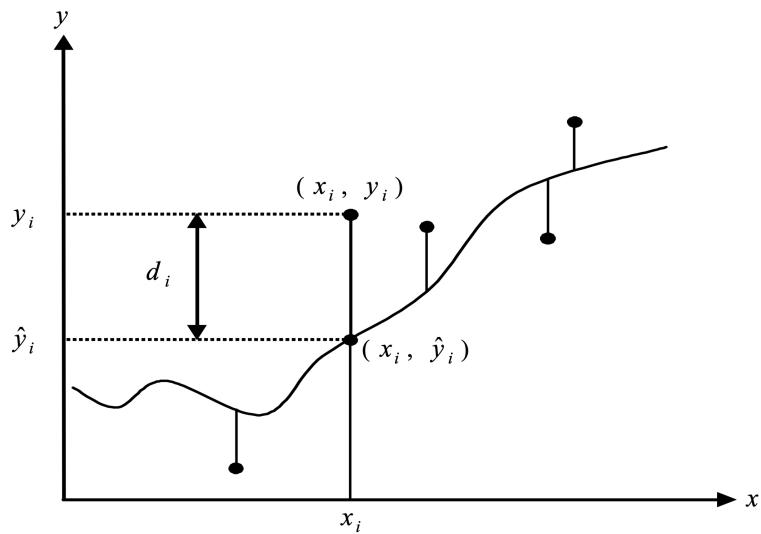


Figura 3.18: Condizione dei *minimi quadrati* [4]

3.2.2 Correlazione e coefficiente di determinazione

Quando la dipendenza tra le due variabili è lineare, si parla di correlazione lineare, e può essere valutata mediante il coefficiente di correlazione lineare

(r):

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

dove il termine al numeratore rappresenta la *covarianza* di X e Y , cioè la variabilità congiunta delle coppie (x_i, y_i) di valori corrispondenti rispetto al proprio valor medio; il denominatore invece rappresenta il prodotto delle deviazioni standard di X ed Y .

Il coefficiente di correlazione lineare gode di importanti proprietà:

- $-1 \leq r \leq 1$;
- si ha $r = 1$ quando tutti i dati sono allineati lungo una retta crescente (figura 3.19);

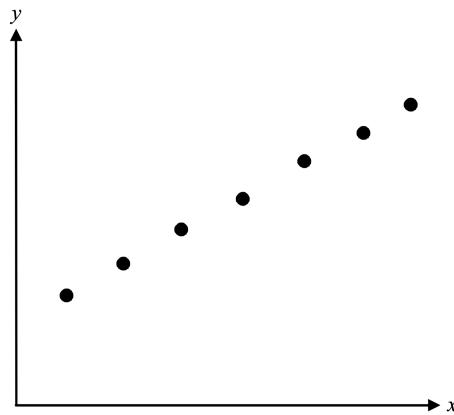
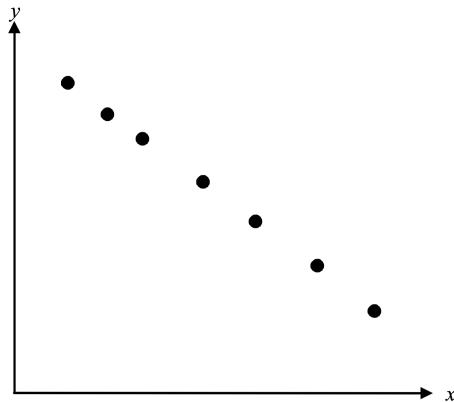
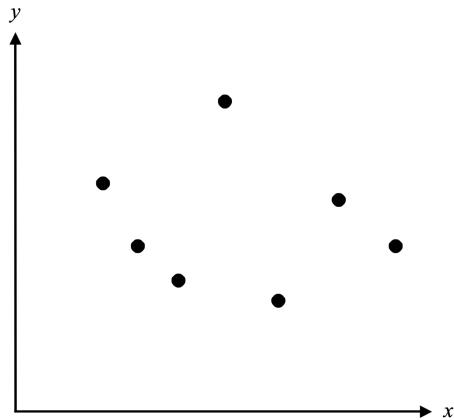


Figura 3.19: Correlazione lineare positiva

- si ha $r = -1$ quando tutti i dati sono allineati lungo una retta decrescente (figura 3.20);

**Figura 3.20:** Correlazione lineare negativa

- si ha $r = 0$ quando non esiste una relazione lineare tra i dati (figura 3.21).

**Figura 3.21:** Nessuna correlazione

È noto che la varianza della variabile Y (σ_y^2) si può scomporre in una parte ($\sigma_{\hat{y}}^2$), detta *varianza spiegata*, in cui la variabilità della Y è dovuta alla dipendenza di Y dalla variabile X , e in una parte (σ_e^2), detta *varianza non spiegata* in cui la variabilità della Y non dipende dalla variabile X , ma da altri fattori. Si può quindi introdurre un secondo indicatore, dato dal

rapporto tra la varianza spiegata e la varianza totale, chiamato **coefficiente di determinazione**:

$$r^2 = \frac{\sigma_{\hat{y}}^2}{\sigma_y^2}$$

che indica quale frazione della variazione della variabile Y può essere ricondotta e spiegata dalle variazioni della variabile X .

Sapendo che:

$$\sigma_y^2 = \sigma_{\hat{y}}^2 + \sigma_e^2$$

allora:

$$r^2 = \frac{\sigma_{\hat{y}}^2}{\sigma_{\hat{y}}^2 + \sigma_e^2}$$

è evidente, quindi, che se la variabilità non spiegata è trascurabile, σ_e^2 tende ad annullarsi ed r^2 avrà un valore prossimo ad 1, mentre diventerà via via minore di 1 al diminuire dell'accordo tra la funzione calcolata e le osservazioni sperimentali.

Minore è la somma residua rispetto alla somma totale dei quadrati, maggiore sarà il valore del coefficiente di determinazione, r^2 , il quale è un indicatore del livello di precisione con cui l'equazione ottenuta dall'analisi di regressione spiega la relazione tra le variabili. [5]

Un'altra metrica utile in ambito delle regressioni è l'errore quadratico medio (in inglese *Mean Squared Error*, MSE) che indica la discrepanza quadratica media fra i valori dei dati osservati ed i valori dei dati stimati:

$$MSE = \frac{\sum_{i=1}^n (x_i - \hat{x}_i)^2}{n}$$

La sua radice quadrata fornisce un ulteriore indice statistico, la cosiddetta radice dell'errore quadratico medio (in inglese *root-mean-square error*, RMSE). L'RMSE può essere anche calcolato come deviazione standard degli scarti. Da notare che l'MSE ed RMSE non sono quantità a-dimensional, ma assumono l'unità di misura della grandezza considerata (RMSE) ed il suo quadrato (MSE).

3.2.3 Analisi dei residui

Esistono metodi utili per diagnosticare le violazioni delle ipotesi di regressione di base: questi si basano principalmente sullo studio dei residui del modello. Spesso infatti la retta di regressione è una semplificazione della realtà e non coglie tutta la variabilità presente in un insieme di dati. [6]

Si definiscono i residui come:

$$e_i = y_i - \hat{y}_i, \quad i = 1, 2, \dots, n$$

dove y_i è il valore osservato e \hat{y}_i è il valore predetto.

Poiché un residuo può essere visto come la deviazione tra i dati e l'adattamento, è anche una misura della variabilità nella variabile di risposta, non spiegata dal modello di regressione. [7]

Eventuali scostamenti dalle ipotesi sugli errori dovrebbero quindi manifestarsi nei residui. L'analisi grafica dei residui è una tecnica efficace per

verificare la linearità della relazione tra le variabili e scoprire diversi tipi di inadeguatezze del modello, tra cui:

- se i residui hanno distribuzione normale (3.2.3.1);
- se gli errori non sono indipendenti rispetto ai valori di X (3.2.3.2);
- se la varianza dei residui è omogenea (3.2.3.3);
- se ci sono degli outliers che influenzano la pendenza della retta (3.2.3.4).

3.2.3.1 Distribuzione degli errori

La distribuzione normale degli errori può essere verificata attraverso un grafico dei quantili, detto anche q-q plot. In questa tipologia di grafico, i quantili teorici di una distribuzione Normale sono riportati sull'asse orizzontale. I quantili dei residui standardizzati sono invece riportati sull'asse verticale. L'idea è che se i residui hanno una distribuzione normale, i loro quantili dovrebbero coincidere con quelli della distribuzione normale. A livello visivo, questo significa che i punti dovrebbero disporsi lungo la *bisettrice*, indicata dalla retta presente nel grafico (figura 3.22).

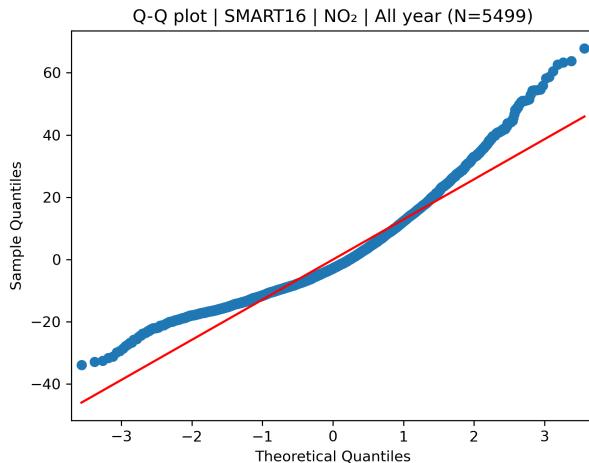


Figura 3.22: Esempio di grafico distribuzione degli errori (q-q plot)

Nella pratica, non capita quasi mai che i punti si dispongano esattamente lungo la bisettrice. Per poter dire che gli errori hanno una distribuzione normale ci si accontenta quindi che i punti siano vicino alla retta.

3.2.3.2 Indipendenza degli errori

Se una variabile indipendente (X) risulta correlata con il termine d'errore, è possibile utilizzare questa variabile per predire quale sarà l'errore del modello di regressione. Questo in generale non è un buon segno, perché la componente di errore di un modello di previsione deve essere sempre imprevedibile.

Per verificare l'indipendenza tra la variabile indipendente e i residui è utile osservare un grafico come quello riportato in figura 3.23, dove sull'asse orizzontale si riportano i valori della x nell'ordine in cui sono stati raccolti, mentre sull'asse verticale i valori dei residui.

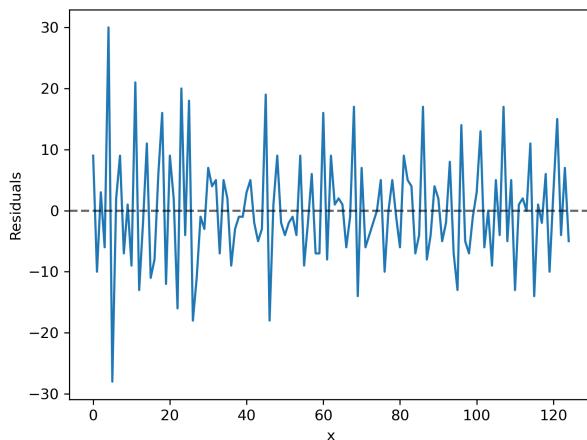


Figura 3.23: Esempio di plot dei residui

L’ipotesi è confermata se non è individuabile nessuna relazione tra le due variabili (ovvero se non compaiono trend o strutture cicliche nel tempo).

3.2.3.3 Omogeneità della varianza dei residui

Per verificare l’ipotesi di omogeneità delle varianze dei residui, è necessario creare un grafico a dispersione (*scatterplot*). I valori stimati della y si riportano sull’asse orizzontale delle x . Sull’asse verticale delle y invece si indicano i valori dei residui (figura 3.24).

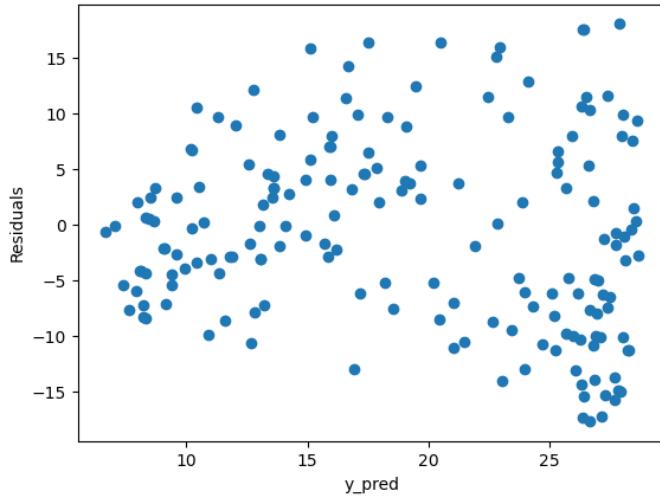


Figura 3.24: Esempio di grafico distribuzione dei residui

Se c’è omogeneità della varianza dei residui, i punti saranno dispersi in modo simile sia nella parte sinistra che in quella destra del grafico. Questa proprietà se verificata prende il nome di **omoschedasticità**.

3.2.3.4 Influenza di outliers

Il grafico a dispersione tra valori predetti e residui permette di individuare anche i possibili *outliers*, ovvero i punti isolati nel grafico (quelli con residui maggiori). Tuttavia, per verificare se ci sono outliers in un modello di regressione, spesso si utilizzano altre tecniche (ad esempio eliminando i punti problematici tramite la distanza di Cook, descritta in 3.2.4.3, oppure applicando stime robuste meno sensibili alle le osservazioni problematiche, ad esempio con la funzione peso di Huber descritta in 3.2.4.2). Nel primo caso è utile anche provare a rifare le analisi di regressione escludendo le osservazioni potenzialmente problematiche e vedere se ci sono differenze nei coefficienti del modello.

Nei modelli di regressione infatti anche un singolo outlier può influenzare in maniera sostanziale la capacità di adattamento del modello ai dati, soprattutto se il campione non risulta molto numeroso.

3.2.4 Modelli di regressione

I modelli di regressione sono ampiamente utilizzati sia per la previsione o la descrizione dei dati che per la stima e il controllo dei parametri.

3.2.4.1 Regressione lineare

Considerata una funzione lineare a due variabili:

$$y = a + b * x$$

In questo caso si deve rendere minima la funzione:

$$\varphi(a, b) = \sum_{i=1}^n [y_i - (a + bx_i)]^2$$

Annulloando le derivate parziali prime rispetto ad a e b si ha il sistema:

$$\begin{cases} \sum_{i=1}^n 2[y_i - (a + bx_i)](-1) = 0 \\ \sum_{i=1}^n 2[y_i - (a + bx_i)](-x_i) = 0 \end{cases}$$

che risolto, fornisce i valori dei parametri:

$$\begin{cases} \hat{a} = \bar{y} - b\bar{x} \\ \hat{b} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \end{cases}$$

dove \bar{x} e \bar{y} indicano le *medie aritmetiche*, rispettivamente di x_i e y_i .

La stima del parametro b , *coefficiente angolare* della funzione lineare, può essere rappresentato nella forma:

$$\hat{b} = \frac{\sum_{i=1}^n \frac{(x_i - \bar{x})(y_i - \bar{y})}{n}}{\sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n}}$$

dove il denominatore è la *varianza* di X (σ_X^2), mentre il numeratore è detto *covarianza* di X e Y (σ_{XY}) e misura la variabilità congiunta delle coppie (x_i, y_i) di valori corrispondenti rispetto al proprio valor medio; quindi, il coefficiente b della retta interpolante esprime la variabilità congiunta di X e Y rapportata alla variabilità della sola X . [8]

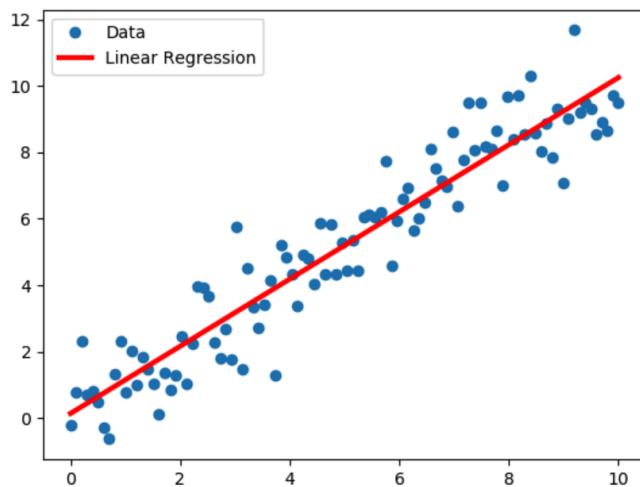


Figura 3.25: Esempio di regressione lineare

La precisione della retta calcolata dalla regressione lineare dipende dal grado di dispersione nei dati. Più i dati sono lineari, più il modello risulterà accurato.

3.2.4.2 Regressione lineare robusta (Huber)

La regressione Huber (in inglese Huber regression, anche detta regressione robusta) è una metodologia statistica per la stima dei parametri di un modello lineare in presenza di *outliers*.

Ci sono situazioni in cui si verifica presenza di valori anomali che influiscono sul modello di regressione, nel senso che possono avere una forte influenza sul metodo dei minimi quadrati, di fatto *deviando* troppo l'equazione di regressione nella loro direzione. Il metodo dei minimi quadrati, infatti, in questi casi ha lo svantaggio di avere la tendenza ad essere dominato da questi valori — infatti sommando il quadrato dei residui ($\sum_{i=1}^n a_i^2$ dove a_i è il residuo i-esimo), la media risulta troppo influenzata da pochi valori a_i particolarmente grandi.

Ci sono due modi per affrontare questa situazione:

- Scartare le osservazioni *scomode* (vedi regressione lineare avanzata 3.2.4.3);
- Applicare procedure di stime robuste in modo che siano meno sensibili alle osservazioni troppo influenti (figura 3.26).

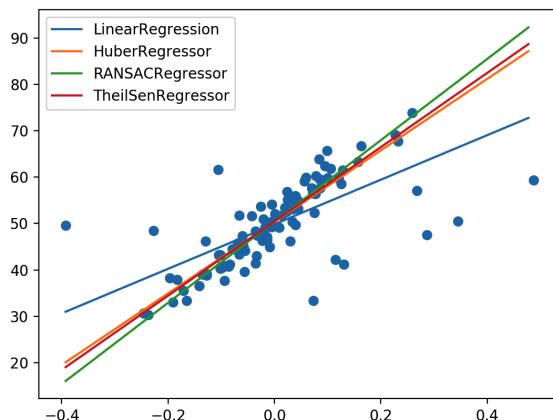


Figura 3.26: Comportamento di modelli di regressione robusta in presenza di outliers

Una delle funzioni di stima robusta, comunemente usata in diversi metodi di regressione per ridurre la sensibilità dei parametri alla presenza di outliers, è la **funzione di Huber**, appartenente ad una classe di estimatori denominata **M-estimator**, e che risulta quadratica per piccoli valori di x , e lineare per valori più grandi. È definita come:

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{per } |a| \leq \delta \\ \delta(|a| - \frac{1}{2}\delta), & \text{altrimenti} \end{cases}$$

Dove la variabile a fa riferimento al residuo, cioè la differenza tra valore osservato e valore predetto ($a = y - f(x)$).

3.2.4.3 Regressione lineare avanzata

Come accennato in 3.2.4.2, un'altra tecnica per la gestione di outlier è quella di applicare il modello sul dataset dopo aver rimosso i valori anomali. Esistono molte metriche su cui basarsi per rimuovere gli outlier da un set di dati: un metodo che viene spesso utilizzato nella regressione è la **distanza di Cook**.

La distanza di Cook è una stima dell'*influenza* di una osservazione in un dataset, in termini di residuo (outlier) o di elevato *leverage*: è un riepilogo di quanto cambierebbe un modello di regressione nel caso in cui venga rimossa l'i-esima osservazione.

In presenza di outliers la distanza di Cook aumenta, e quindi questi dati ad alta influenza hanno un maggiore impatto sulle stime dei parametri della regressione.

La distanza di Cook [9] dell'osservazione i ($\forall i = 1, \dots, n$) è definita come:

$$D_i = \frac{\sum_{j=1}^n (\hat{y}_j - \hat{y}_{j(i)})^2}{ps^2}$$

dove:

- n è il numero di osservazioni;
- \hat{y}_j è il valore predetto;
- $\hat{y}_{j(i)}$ è la risposta ottenuta escludendo l'i-esima osservazione.

Oppure, in modo equivalente:

$$D_i = \frac{e_i^2}{ps^2} \left[\frac{h_i}{(1 - h_i)^2} \right]$$

dove:

- $e_i = y_i - \hat{y}_i$ è l'i-esimo residuo;
- p è il numero di coefficienti della regressione;
- s^2 è l'errore quadratico medio (MSE);
- h_i è il peso che l'i-esimo osservazione ha sul valore della regressione (*leverage*).

Un esempio di rilevazione grafica di outlier tramite distanza di Cook è riportato in figura 3.27.

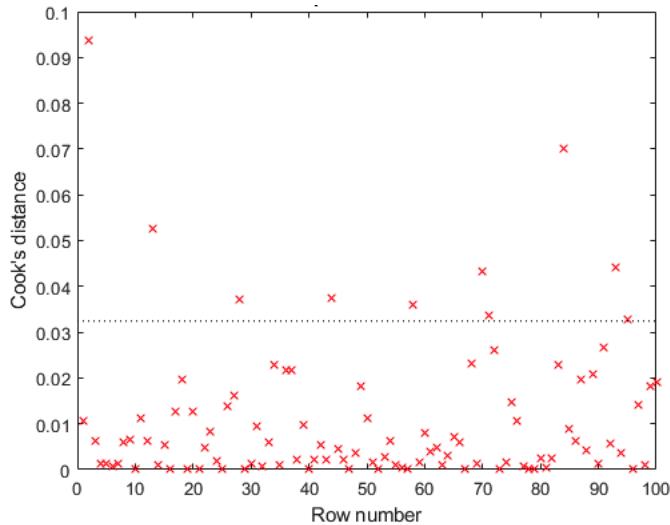


Figura 3.27: Riconoscimento di outlier tramite distanza di Cook

Vi sono diverse opinioni riguardo al valore di soglia di *cut-off*, oltre la quale un dato può essere considerato un outlier. In [10] viene proposta:

$$D_i > \frac{4}{n}$$

dove n è il numero di osservazioni. La distanza di Cook può anche essere utilizzata per individuare regioni dello spazio nelle quali sarebbe necessario effettuare una validazione, ad esempio acquisendo più dati.

3.2.4.4 Regressione Ridge

Nella statistica e nel Machine Learning, la regressione Ridge è un metodo di analisi di regressione che applica una fase di **regolarizzazione** al fine di migliorare l'accuratezza della previsione, prevenire l'*overfitting* e penalizzare la complessità del modello. Insieme al LASSO (vedi 3.2.4.5) è un modello

di regressione che viene ripreso anche da tecniche di Boosting di Machine Learning. In generale esistono due tipi di penalizzazione:

- **L1**: penalizza il valore assoluto dei coefficienti del modello (es. Lasso);
- **L2**: penalizza il quadrato del valore dei coefficienti del modello (es. Ridge).

La regressione Ridge usa la penalità L2: in pratica questo produce coefficienti piccoli, ma nessuno di loro è mai annullato (*feature shrinkage*).

Richiamando il metodo dei minimi quadrati (3.2.1) si deve minimizzare la somma dei quadrati dei residui (RSS):

$$\text{RSS} = \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2$$

Nella regressione Ridge si aggiunge anche un termine di penalità, ottenendo quindi:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^p \beta_j^2$$

Dove λ è un parametro di *tuning* che serve proprio a controllare l'effetto della penalità: un valore $\lambda = 0$ infatti non avrà effetto sul risultato finale (l'equazione viene ricondotta a quella dei minimi quadrati), al contrario per $\lambda \rightarrow \infty$ invece i coefficienti di regressione stimati tenderanno a zero poiché si darà molto peso alla penalità del modello. [11]

Il modello di regressione di Ridge presenta dei vantaggi rispetto a quello dei minimi quadrati, soprattutto per quanto riguarda il *bias-variance trade-off*: in generale, quando c'è una relazione lineare tra i predittori e la variabile

risposta, il modello dei minimi quadrati comporta poco bias ma alta varianza. Questo si traduce nel fatto che una piccola variazione nell'osservazione può generare un cambiamento notevole nei coefficienti stimati. [12]

3.2.4.5 Regressione Lasso

Lo svantaggio della regressione Ridge è il fatto di considerare tutte le variabili per la predizione nel modello finale. Il termine di regolarizzazione $\lambda \sum_{j=1}^p \beta_j^2$ tende ad assegnare ai coefficienti valori vicini allo zero, ma non perfettamente zero, a meno che $\lambda = 0$. Questo crea problemi non tanto per l'accuratezza della predizione, ma per l'interpretazione delle varabili, soprattutto quando il numero di queste diventa alto.

La regressione Lasso (acronimo di *least absolute shrinkage and selection operator*, ovvero operatore di restringimento e selezione minimo assoluto) è un'alternativa alla regressione Ridge utilizzata proprio per superare questo problema. L'unica differenza sta nel termine di regolarizzazione, ovvero:

$$\lambda \sum_{j=1}^p |\beta_j|$$

Per cui l'equazione del modello diventa:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^p |\beta_j| = RSS + \lambda \sum_{j=1}^p |\beta_j|$$

Anche nel caso di regressione Lasso il parametro di regolarizzazione tende a stimare i valori dei coefficienti verso lo zero ma, a differenza della regressione Ridge, la penalità $\lambda \sum_{j=1}^p |\beta_j|$ costringe uno o più coefficienti ad essere esattamente zero per certi valori di λ . [12]

3.2.4.6 Regressione polinomiale

La regressione polinomiale è una generalizzazione della regressione lineare, infatti utilizza lo stesso metodo matematico della variante lineare, ma assume che la relazione di funzione che caratterizza i dati sia meglio descritta, anzichè da una retta, da un polinomio. In questo caso il metodo dei minimi quadrati può essere utilizzato anche per adattare una funzione polinomiale a un insieme di dati. Considerato un polinomio di grado k :

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots + a_kx^k$$

In questo caso il sistema di equazioni da risolvere è:

$$\begin{cases} na_0 + a_1 \sum_{i=1}^n x_i + a_2 \sum_{i=1}^n x_i^2 + \dots + a_k \sum_{i=1}^n x_i^k = \sum_{i=1}^n y_i \\ a_1 \sum_{i=1}^n x_i + a_2 \sum_{i=1}^n x_i^2 + \dots + a_k \sum_{i=1}^n x_i^k = \sum_{i=1}^n x_i y_i \\ \dots \\ a_1 \sum_{i=1}^n x_i^k + a_2 \sum_{i=1}^n x_i^{k+1} + \dots + a_k \sum_{i=1}^n x_i^{2k} = \sum_{i=1}^n x_i^k y_i \end{cases}$$

che, risolto, permette di ricavare i parametri $a_0, a_1, a_2, \dots, a_k$.

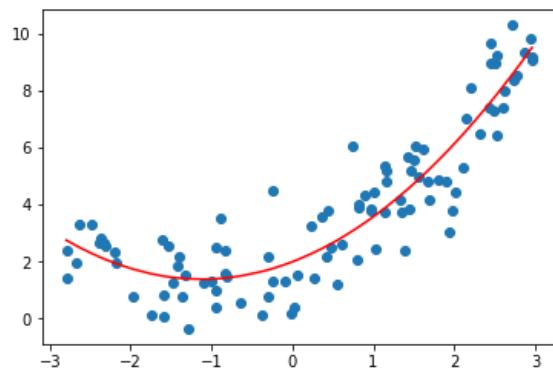


Figura 3.28: Esempio di regressione polinomiale

Ci sono diverse considerazioni importanti che emergono quando si adatta un polinomio in una variabile: una di queste riguarda la scelta dell'ordine del modello. Come regola generale, l'uso di polinomi di ordine elevato ($n > 3$) dovrebbe essere evitato: un modello di ordine basso è quasi sempre preferibile a un modello di ordine elevato per ragioni di minore complessità, di coerenza con i dati e per evitare *overfitting*.

Come caso estremo, è sempre possibile trovare un polinomio di grado $n - 1$ ad n punti che risulti in un buon adattamento dei dati. Nella maggior parte dei casi, però, questo non farebbe nulla per migliorare la comprensione della funzione sconosciuta, né sarebbe probabilmente un buon predittore.

3.2.4.7 Regressione con Random Forest

La regressione Random Forest è un algoritmo di apprendimento supervisionato che utilizza il metodo di apprendimento *ensemble* per la regressione tramite alberi di decisione. Il metodo di apprendimento ensemble è una tecnica che combina le previsioni di più algoritmi di apprendimento automatico per effettuare una previsione più accurata rispetto a un singolo modello. [13]

In particolare, una foresta casuale (random forest) opera adattando una serie di alberi decisionali su vari sottocampioni del set di dati e utilizza la media dei risultati per migliorare l'accuratezza predittiva e controllare l'*overfitting*. In breve, l'algoritmo funziona eseguendo i seguenti passi:

1. Sceglie a caso k osservazioni dati dal training set;
2. Costruisce un albero decisionale associato a queste k osservazioni;
3. Sceglie il numero N di alberi da costruire e ripete i passaggi 1 e 2 per ciascuno;

4. Per una nuova osservazione, fa in modo che ciascuno degli N alberi preveda il valore di y , e assegna il nuovo punto alla media su tutti i valori y previsti.

Uno dei principali svantaggi degli alberi decisionali è che risultano molto inclini a provocare *overfitting*: funzionano bene sui dati di training, ma non sono così flessibili per fare previsioni su campioni non visti in precedenza. Sebbene esistano soluzioni alternative per migliorare questo aspetto, come ad esempio ridurre il numero di alberi, questo riduce il loro potere predittivo. Generalmente sono modelli con *bias* medio e varianza alta, ma sono semplici e di facile interpretazione.

3.2.4.8 Regressione con Gradient Boosting

Il Gradient Boosting è una tecnica di Machine Learning che ha alla base la stima iterativa di alberi sui residui ottenuti ad ogni passo e l'aggiornamento in maniera adattiva delle stime. Questa tecnica riprende il concetto matematico del *gradient descent*, per cui lo split scelto sarà quello che favorisce l'avvicinamento al punto di minimo della funzione obiettivo.

Il *gradient descent* è un algoritmo di ottimizzazione che consente di individuare il valore minimo di una funzione di costo per sviluppare un modello con una previsione accurata.

L'algoritmo Gradient Boosting applicato a problemi di regressione può essere descritto nei seguenti passi:

1. Si inizializza il modello con un valore noto;
2. Si considera sul training set una *loss function*, ovvero una funzione differenziabile che esprima una valutazione della predizione (es. $\frac{1}{2}(y_i - \hat{y}_i)$ dove y_i è l'osservazione e \hat{y}_i è la predizione);

3. Scelto un numero massimo, si itera modellando un albero di regressione seguendo una procedura di discesa del gradiente, in modo da minimizzare la *loss function*. L'albero ottenuto viene aggiunto alla sequenza di alberi già esistente, nel tentativo di correggere o migliorare l'output finale del modello.

3.2.4.9 Regressione con SVR

Un altro modello per la regressione è SVR (*support-vector regression*), basato sui modelli SVM (support-vector machines) di apprendimento supervisionato, spesso impiegati nel problema della classificazione. [14]

Rispetto alla regressione lineare e al metodo dei minimi quadrati, SVR presenta più flessibilità perchè consente di definire una soglia di accettazione dell'errore nel modello. Infatti l'algoritmo SVR si propone di minimizzare non l'errore quadratico, come nel metodo dei minimi quadrati, ma i coefficienti (nello specifico, la norma al quadrato del vettore dei coefficienti). La funzione obiettivo quindi diventa:

$$\min \frac{1}{2} \|\mathbf{w}\|^2$$

Il termine di errore invece è gestito nei vincoli, dove si imposta l'errore assoluto minore o uguale a un margine specificato, chiamato errore massimo (ε):

$$|y_i - w_i x_i| \leq \varepsilon$$

Il tuning del parametro ε consente di ottenere la precisione desiderata del modello di regressione. Solitamente alla funzione obiettivo si aggiunge anche

delle variabili di *slack* (ξ_i), che indicano la deviazione dal margine di ciascun valore che supera la soglia ε (lo scopo è di minimizzarle il più possibile). La funzione obiettivo in questo caso diventa:

$$\min \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n |\xi_i|$$

dove C è un altro iperparametro regolabile: all'aumentare di C , aumenta anche la tolleranza per i punti al di fuori di ε . Quando invece C si avvicina a 0, la tolleranza si avvicina a 0 e l'equazione ricade nel caso semplificato.

I modelli di regressione SVM possono anche eseguire una regressione non lineare, applicando il *kernel trick* per mappare i dati in uno spazio multidimensionale. Il tipo di kernel da usare nell'algoritmo è un altro parametro da definire nel modello: i kernel più comuni sono quello lineare, polinomiale (di grado n) o RBF (basato su *funzione di base radiale*).

3.2.4.10 Regressione con KernelRidge

La regressione Kernel Ridge (o KRR, *Kernel Ridge Regression*) è un altro modello che combina la regressione Ridge (descritta in 3.2.4.4) con il *kernel trick*, imparando quindi una funzione lineare nello spazio indotto dal rispettivo kernel e dai dati. Per i kernel non lineari, questo corrisponde a una funzione non lineare nello spazio originale. [15]

La forma del modello di regressione KRR è identica a quello basato su SVR (descritto in 3.2.4.9), ma vengono utilizzate diverse funzioni di *loss*: KRR minimizza l'errore quadratico mentre SVR minimizza i coefficienti in base alla soglia ε . In generale il modello KRR risulta più veloce per dataset di medie dimensioni.

3.3 Esperimenti e risultati ottenuti

Una volta collezionati i dati sia dalle centraline AirQino che ARPAT è stata avviata la fase di calibrazione, con le seguenti attività per ciascun inquinante (NO_2 , $\text{PM}_{2.5}$ e PM_{10}):

- Scatterplot preliminare delle misurazioni AirQino confrontate con le misurazioni di riferimento ARPAT, per individuare eventuali relazioni tra le variabili;
- Analisi dei residui della regressione lineare, per verificare le assunzioni del modello (vedi sezione 3.2.3);
- Applicati tredici modelli di regressione diversi su tutto il dataset:
 - Lineare semplice (3.2.4.1);
 - Lineare *robusto*, per ridurre l'influenza di outlier utilizzando la funzione peso di Huber (3.2.4.2);
 - Lineare *avanzato*, con rimozione di *outlier* tramite distanza di Cook basata sull'influenza della singola osservazione (3.2.4.3);
 - Ridge e Lasso (3.2.4.4 e 3.2.4.5);
 - Polinomiale (grado 2 e 3, 3.2.4.6);
 - SVR (con kernel lineare, polinomiale e RBF, come descritto in 3.2.4.9);
 - Random Forest, Gradient Boosting e KernelRidge (3.2.4.7, 3.2.4.8 e 3.2.4.10).
- Applicati gli stessi modelli ai dati mese per mese, per validare la consistenza dei risultati;

- Infine, sono stati raccolti e confrontati i coefficienti delle curve ottenute dai diversi modelli.

Per tutti i risultati ottenuti e presentati nella sezione seguente sono stati presi alcuni accorgimenti:

- Per misurare le performance dei vari modelli di regressione sono state utilizzate due metriche: il coefficiente di determinazione (R^2) e la radice dell'errore quadratico medio (RMSE), come visto in 3.2.2;
- Tutti i modelli sono stati precedentemente preparati tramite *tuning* di parametri ottimali (dove possibile) con tecniche di *cross-validation*;
- Ciascun risultato riportato di seguito (R^2 e RMSE) rappresenta la media su 1000 iterazioni, con *shuffle* dei dati ad ogni iterazione per ridurre al minimo la possibilità di *overfitting*;
- Per allenare i modelli il dataset è stato suddiviso in train set 70% e test set 30%;
- Tutti gli esperimenti sono stati eseguiti con Python 3.9 e con l'aiuto di librerie aggiuntive quali `scikit-learn`, `pandas`, `matplotlib`, `numpy` e `seaborn`.

3.3.1 Risultati NO₂

Il confronto tra le misurazioni AirQino e le misurazioni di riferimento ARPAT per NO₂ sono riportate nello *scatterplot* di figura 3.29.

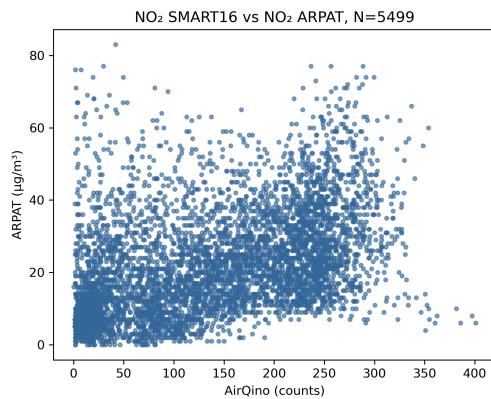


Figura 3.29: Scatterplot dataset NO₂

I risultati dell'analisi grafica dei residui sono riportati in figura 3.30.

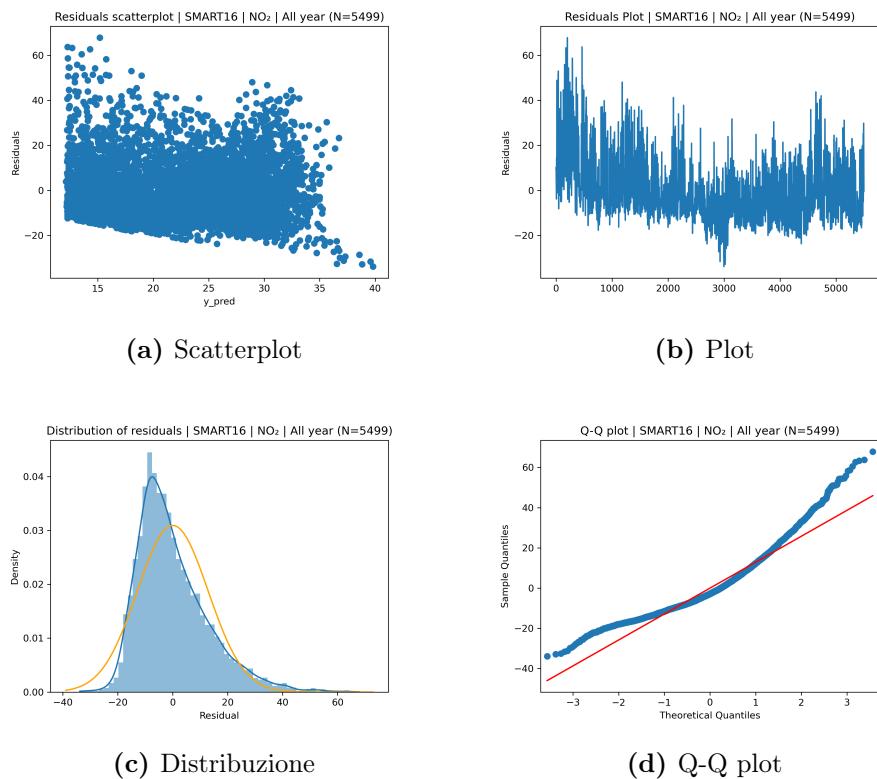


Figura 3.30: Analisi dei residui dataset NO₂

La tabella 3.3 mostra i risultati, in termini di R^2 e RMSE, della procedura di calibrazione applicata al sensore di NO_2 per ciascun modello di regressione, su tutto il dataset.

Modello di regressione	R^2	RMSE ($\mu\text{g}/\text{m}^3$)
Lineare	0.304	11.393
Lineare robusto (Huber)	0.312	11.424
Lineare avanzato (Cook)	0.373	9.447
Ridge	0.305	11.375
Lasso	0.303	11.403
Polinomiale (grado 2)	0.305	11.382
Polinomiale (grado 3)	0.302	11.381
Random Forest	-0.034	13.839
Gradient Boosting	0.103	12.887
SVR (Kernel lineare)	0.287	11.528
SVR (Kernel polinomiale)	0.255	11.816
SVR (Kernel RBF)	0.294	11.445
KernelRidge	0.301	11.344

Tabella 3.3: Risultati della calibrazione NO_2

E in figura 3.31 gli stessi risultati in forma di istogramma:

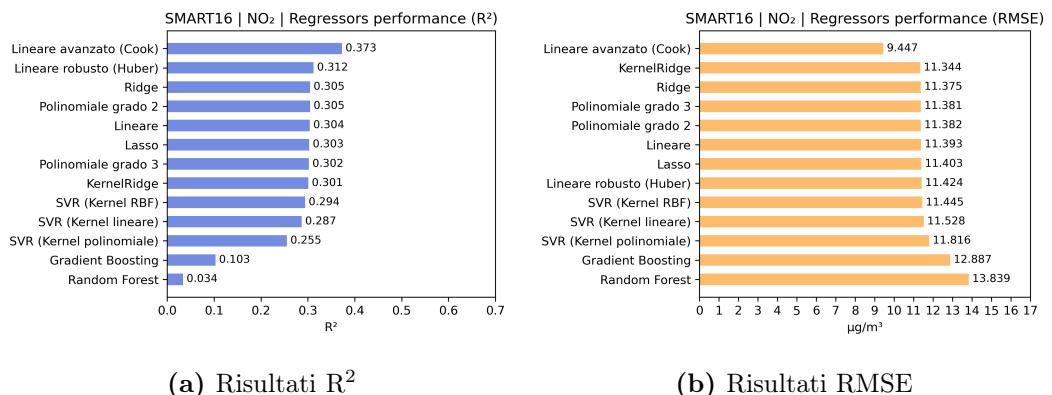


Figura 3.31: Iistogramma dei risultati della calibrazione NO_2

Per quanto riguarda l'applicazione del modello lineare *avanzato*, è stata riapplicata la regressione dopo aver tolto tutti i punti con distanza di Cook (3.2.4.3) sopra una certa soglia ($\frac{4}{\text{numero di osservazioni}} = \frac{4}{5110} \approx 0,0007$). In particolare sono stati individuati e rimossi 282 outlier su 5110 osservazioni totali, circa il 5.52% (figura 3.32).

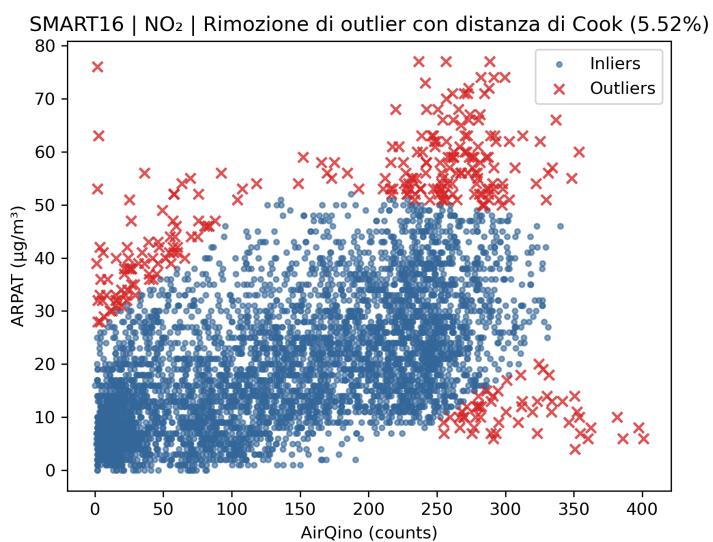


Figura 3.32: Rimozione di outlier con distanza di Cook su dataset NO₂

Nelle tabelle 3.4 e 3.5 sono invece riportati rispettivamente R^2 e RMSE della calibrazione su dataset NO₂ fatta mese per mese.

Anche in questo caso il modello di regressione migliore è risultato in media quello *avanzato*, con riconoscimento e rimozione di outlier, anche se per alcuni mesi il modello polinomiale di terzo grado e il modello KernelRidge hanno ottenuto risultati migliori.

Modello	Gen	Feb	Mar	Apr	Set	Ott	Nov	Dic
Lineare	0.44	0.45	0.35	0.21	0.16	0.36	0.16	0.42
Lineare robusto (Huber)	0.44	0.45	0.35	0.21	0.16	0.36	0.16	0.42
Lineare avanzato (Cook)	0.53	0.48	0.38	0.27	0.29	0.39	0.18	0.45
Ridge	0.45	0.45	0.35	0.20	0.17	0.37	0.16	0.43
Lasso	0.43	0.45	0.34	0.21	0.16	0.37	0.17	0.42
Polinomiale (grado 2)	0.42	0.46	0.37	0.24	0.27	0.39	0.17	0.42
Polinomiale (grado 3)	0.42	0.51	0.48	0.23	0.27	0.41	0.17	0.42
Random Forest	0.15	0.33	0.21	-0.14	-0.07	0.12	-0.16	0.12
Gradient Boosting	0.13	0.17	0.16	0.08	0.09	0.13	0.06	0.14
SVR (lineare)	0.42	0.43	0.32	0.18	0.14	0.37	0.14	0.42
SVR (polinomiale)	0.37	0.37	0.22	0.05	0.07	0.29	0.13	0.39
SVR (RBF)	0.38	0.50	0.46	0.19	0.25	0.39	0.14	0.43
KernelRidge	0.44	0.45	0.37	0.24	0.26	0.39	0.18	0.41

Tabella 3.4: Risultati della calibrazione NO₂ mese per mese (R^2)

Modello	Gen	Feb	Mar	Apr	Set	Ott	Nov	Dic
Lineare	13.31	11.68	9.90	9.30	8.73	8.72	12.30	9.86
Lineare robusto (Huber)	13.27	11.55	9.91	9.34	8.70	8.75	12.36	9.80
Lineare avanzato (Cook)	11.32	10.13	8.64	6.95	7.08	7.99	10.64	9.42
Ridge	13.23	11.52	9.91	9.18	8.73	8.67	12.33	9.72
Lasso	13.43	11.56	9.99	9.31	8.73	8.81	12.23	9.94
Polinomiale (grado 2)	13.54	11.46	9.66	9.11	8.19	8.57	12.36	9.72
Polinomiale (grado 3)	13.54	10.99	8.88	9.17	8.15	8.39	12.27	9.79
Random Forest	16.27	12.76	10.85	11.11	9.84	10.27	14.58	12.03
Gradient Boosting	16.64	14.30	11.30	9.90	9.09	10.34	13.04	11.98
SVR (lineare)	13.51	11.82	10.21	9.53	8.82	8.77	12.35	9.86
SVR (polinomiale)	14.10	12.43	10.74	10.07	9.29	9.23	12.51	10.05
SVR (RBF)	13.70	10.98	9.02	9.39	8.24	8.57	12.50	9.74
KernelRidge	13.35	11.51	9.80	9.17	8.21	8.62	12.06	9.78

Tabella 3.5: Risultati della calibrazione NO₂ mese per mese (RMSE)

Infine, la tabella 3.6 riporta i coefficienti delle curve di regressione ottenuti applicando i modelli al dataset NO₂. I coefficienti *a*, *b*, *c*, *d*, *e* si riferiscono ad una equazione polinomiale generica del tipo $y = a + bx + cx^2 + dx^3 + ex^4$.

Modello di regressione	a	b	c	d	e
Lineare	6.52	12.25	/	/	/
Lineare robusto (Huber)	6.62	11.98	/	/	/
Lineare avanzato (Cook)	6.84	9.56	/	/	/
Lasso	6.52	12.26	/	/	/
Ridge	6.52	12.25	/	/	/
Polinomiale (grado 2)	12.11	6.83	-0.10	/	/
Polinomiale (grado 3)	12.71	4.23	1.91	-0.40	/
Polinomiale (grado 4)	9.97	22.86	-22.40	10.21	-1.47

Tabella 3.6: Coefficienti della calibrazione NO₂

3.3.2 Risultati PM_{2.5}

Il confronto tra le misurazioni AirQino e le misurazioni di riferimento ARPAT per PM_{2.5} sono riportate nello *scatterplot* di figura 3.33.

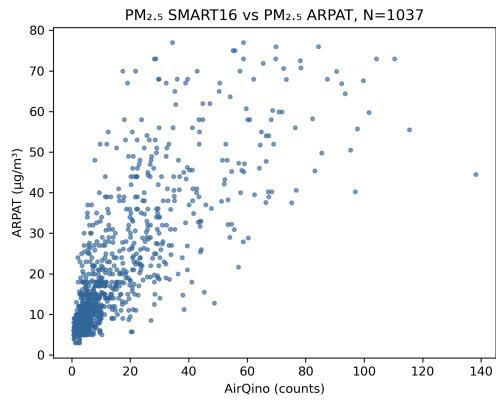


Figura 3.33: Scatterplot dataset PM_{2.5}

I risultati dell'analisi grafica dei residui sono riportati in figura 3.34.

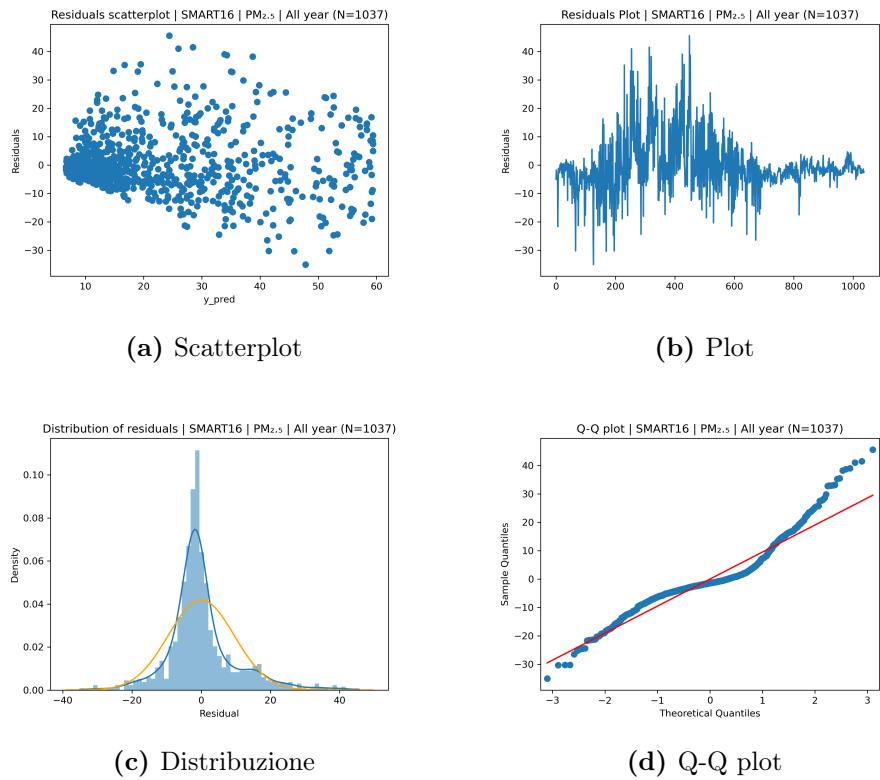


Figura 3.34: Analisi dei residui dataset PM_{2.5}

La tabella 3.7 mostra i risultati della procedura di calibrazione applicata al sensore di PM_{2.5} per ciascun modello di regressione, su tutto il dataset.

Modello di regressione	R ²	RMSE ($\mu\text{g}/\text{m}^3$)
Lineare	0.729	8.737
Lineare robusto (Huber)	0.730	8.706
Lineare avanzato (Cook)	0.801	5.894
Ridge	0.730	8.696
Lasso	0.728	8.735
Polinomiale (grado 2)	0.758	8.254
Polinomiale (grado 3)	0.756	8.227
Random Forest	0.613	10.411
Gradient Boosting	0.246	14.602
SVR (Kernel lineare)	0.719	8.877
SVR (Kernel polinomiale)	0.430	12.588
SVR (Kernel RBF)	0.739	8.581
KernelRidge	0.757	8.208

Tabella 3.7: Risultati della calibrazione PM_{2.5}

E in figura 3.35 gli stessi risultati in forma di istogramma:

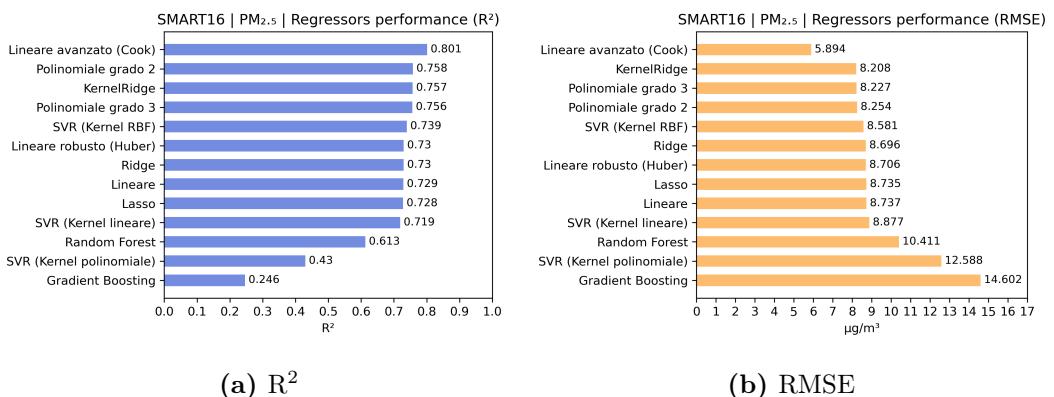


Figura 3.35: Istogramma dei risultati della calibrazione PM_{2.5}

Anche in questo caso nel modello lineare *avanzato*, è stata riapplicata la regressione dopo aver tolto tutti i punti con distanza di Cook (3.2.4.3) sopra la soglia pari a $\frac{4}{\text{numero di osservazioni}} = \frac{4}{1037} \approx 0,0038$. In particolare sono stati individuati e rimossi 80 outlier su 1037 osservazioni totali, circa il 7.71% (figura 3.36).

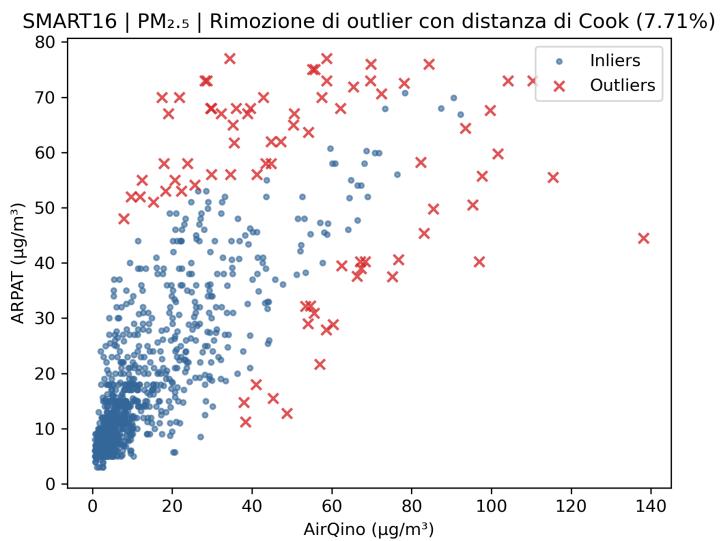


Figura 3.36: Rimozione di outlier con distanza di Cook su dataset PM_{2.5}

Nelle tabelle 3.8 e 3.9 sono invece riportati rispettivamente R^2 e RMSE della calibrazione su dataset PM_{2.5} fatta mese per mese.

Modello	Set	Ott	Nov	Dic	Gen	Feb	Mar	Apr	Mag	Giu	Lug	Ago
Lineare	0.29	0.49	0.18	0.72	0.61	0.60	0.56	0.19	0.02	0.26	0.55	0.61
Lineare robusto (Huber)	0.28	0.48	0.19	0.72	0.62	0.58	0.58	0.18	-0.00	0.27	0.55	0.61
Lineare avanzato (Cook)	0.58	0.62	0.15	0.73	0.66	0.68	0.62	0.29	0.02	0.28	0.69	0.70
Ridge	0.26	0.48	0.18	0.71	0.61	0.59	0.57	0.18	0.01	0.26	0.52	0.60
Lasso	0.28	0.47	0.18	0.72	0.60	0.59	0.56	0.18	0.01	0.26	0.54	0.62
Polinomiale (grado 2)	0.32	0.46	0.22	0.73	0.68	0.55	0.54	0.09	-0.03	0.28	0.59	0.59
Polinomiale (grado 3)	-0.70	0.31	0.19	0.66	0.65	0.54	0.53	-0.24	-0.10	0.26	0.33	0.60
Random Forest	0.23	0.37	-0.03	0.65	0.50	0.29	0.28	-0.17	-0.68	-0.24	0.46	0.49
Gradient Boosting	0.09	0.10	0.03	0.20	0.17	0.08	0.12	-0.02	-0.08	0.02	0.14	0.15
SVR (lineare)	0.14	0.45	0.14	0.69	0.60	0.53	0.53	0.15	0.01	0.24	0.52	0.61
SVR (polinomiale)	-0.55	-0.15	0.08	0.50	0.26	0.40	0.51	0.12	-0.02	0.17	-0.09	0.57
SVR (RBF)	0.16	0.44	0.22	0.65	0.58	0.39	0.37	-0.06	-0.36	-0.07	0.62	0.50
KernelRidge	0.36	0.45	0.25	0.73	0.67	0.57	0.54	0.06	-0.06	0.29	0.62	0.60

Tabella 3.8: Risultati della calibrazione PM_{2.5} mese per mese (R^2)

Modello	Set	Ott	Nov	Dic	Gen	Feb	Mar	Apr	Mag	Giu	Lug	Ago
Lineare	3.78	6.27	13.35	11.55	12.39	7.70	6.27	3.74	1.38	1.92	1.54	2.05
Lineare robusto (Huber)	3.77	6.25	13.19	11.58	12.17	7.87	6.27	3.77	1.38	1.93	1.53	2.04
Lineare avanzato (Cook)	2.92	4.99	12.78	10.62	11.55	7.08	5.53	3.38	1.25	1.80	1.26	1.72
Ridge	3.86	6.31	13.36	11.69	12.33	7.82	6.29	3.80	1.38	1.96	1.60	2.06
Lasso	3.82	6.32	13.32	11.56	12.48	7.85	6.29	3.77	1.36	1.95	1.57	2.04
Polinomiale (grado 2)	3.63	6.42	12.94	11.38	11.29	8.03	6.43	3.95	1.38	1.88	1.45	2.11
Polinomiale (grado 3)	5.10	6.89	13.13	12.57	11.46	8.13	6.56	4.43	1.42	1.94	1.73	2.03
Random Forest	3.85	6.86	14.86	12.62	13.54	10.14	7.93	4.44	1.76	2.48	1.68	2.27
Gradient Boosting	4.33	8.32	14.71	19.83	18.33	11.96	9.30	4.24	1.44	2.29	2.14	3.09
SVR (lineare)	3.98	6.46	13.55	12.10	12.46	8.30	6.50	3.79	1.39	1.98	1.57	2.05
SVR (polinomiale)	5.27	9.25	14.23	15.22	16.60	9.09	6.74	3.94	1.39	2.08	2.20	2.13
SVR (RBF)	3.99	6.51	12.98	12.88	12.60	9.41	7.54	4.25	1.57	2.33	1.41	2.23
KernelRidge	3.56	6.45	12.65	11.31	11.31	7.92	6.36	4.01	1.42	1.88	1.39	2.06

Tabella 3.9: Risultati della calibrazione PM_{2.5} mese per mese (RMSE)

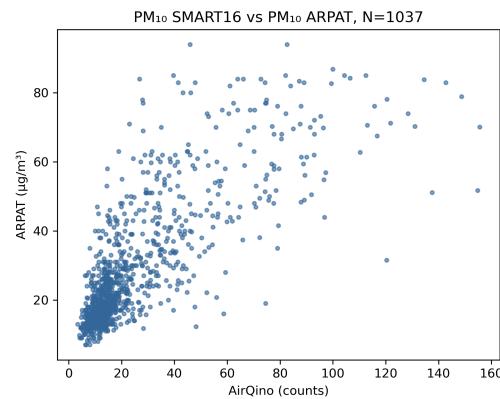
Infine, la tabella 3.10 riporta i coefficienti delle curve di regressione ottenuti applicando i modelli al dataset PM_{2.5}. Anche in questo caso i coefficienti a , b , c , d , e si riferiscono ad una equazione polinomiale generica del tipo $y = a + bx + cx^2 + dx^3 + ex^4$.

Modello di regressione	a	b	c	d	e
Lineare	13.15	9.50	/	/	/
Lineare robusto (Huber)	13.33	9.23	/	/	/
Lineare avanzato (Cook)	10.31	7.67	/	/	/
Lasso	13.02	9.60	/	/	/
Ridge	13.01	9.61	/	/	/
Polinomiale (grado 2)	5.93	22.37	-2.34	/	/
Polinomiale (grado 3)	5.19	25.32	-4.07	0.23	/
Polinomiale (grado 4)	4.17	30.76	-9.29	1.75	-0.13

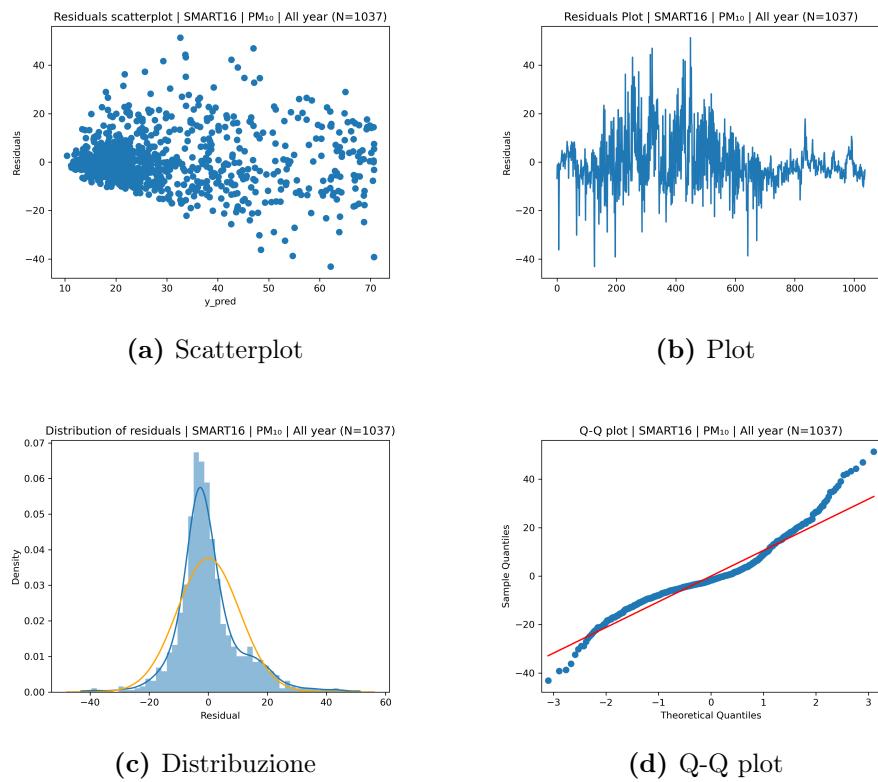
Tabella 3.10: Coefficienti della calibrazione PM_{2.5}

3.3.3 Risultati PM₁₀

Il confronto tra le misurazioni AirQino e le misurazioni di riferimento ARPAT per PM₁₀ sono riportate nello *scatterplot* di figura 3.37.

**Figura 3.37:** Scatterplot dataset PM₁₀

I risultati dell'analisi grafica dei residui sono riportati in figura 3.38.

**Figura 3.38:** Analisi dei residui dataset PM₁₀

La tabella 3.11 mostra i risultati della procedura di calibrazione applicata al sensore di PM₁₀ per ciascun modello di regressione, su tutto il dataset.

Modello di regressione	R ²	RMSE ($\mu\text{g}/\text{m}^3$)
Lineare	0.715	9.706
Lineare robusto (Huber)	0.716	9.760
Lineare avanzato (Cook)	0.785	7.103
Ridge	0.715	9.707
Lasso	0.716	9.754
Polinomiale (grado 2)	0.739	9.343
Polinomiale (grado 3)	0.738	9.339
Random Forest	0.590	11.686
Gradient Boosting	0.238	16.004
SVR (Kernel lineare)	0.707	9.860
SVR (Kernel polinomiale)	0.513	12.724
SVR (Kernel RBF)	0.719	9.672
KernelRidge	0.738	9.321

Tabella 3.11: Risultati della calibrazione PM₁₀

E in figura 3.39 gli stessi risultati in forma di istogramma:

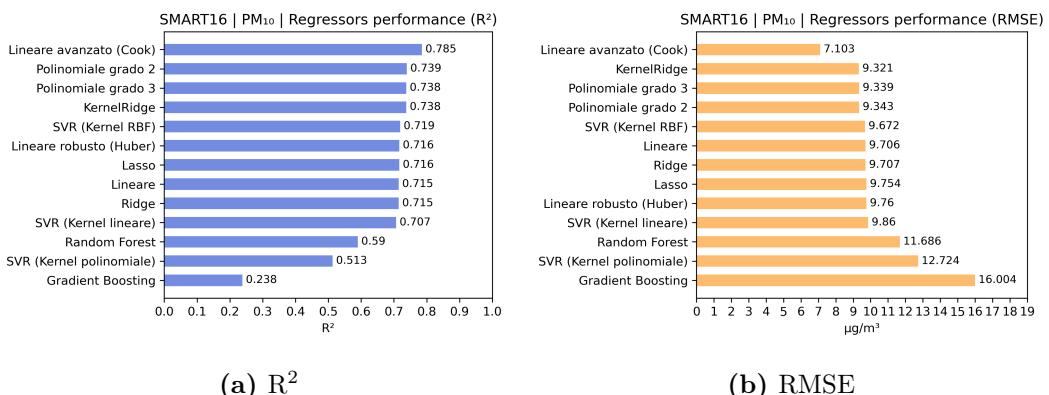


Figura 3.39: Istogramma dei risultati della calibrazione PM₁₀

Anche in questo caso nel modello lineare *avanzato*, è stata riapplicata la regressione dopo aver tolto tutti i punti con distanza di Cook (3.2.4.3) sopra la soglia pari a $\frac{4}{\text{numero di osservazioni}} = \frac{4}{1037} \approx 0,0038$. In particolare sono stati individuati e rimossi 70 outlier su 1037 osservazioni totali, circa il 6.71% (figura 3.40).

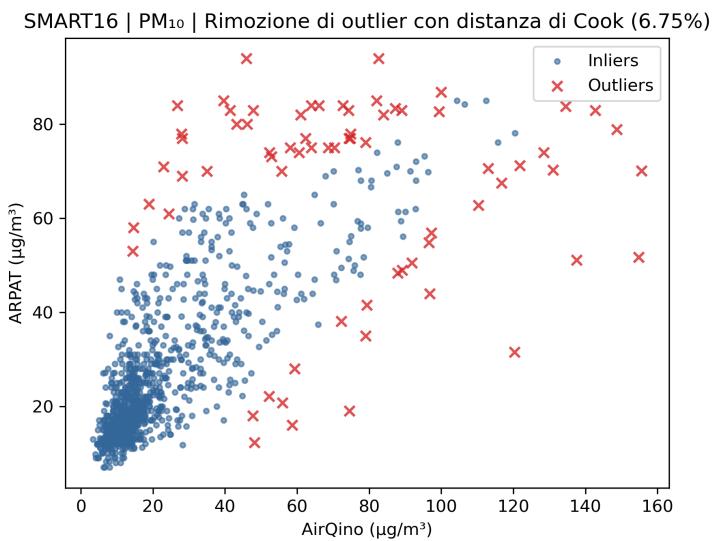


Figura 3.40: Rimozione di outlier con distanza di Cook su dataset PM₁₀

Nelle tabelle 3.12 e 3.13 sono invece riportati rispettivamente R^2 e RMSE della calibrazione su dataset PM₁₀ fatta mese per mese.

Modello	Set	Ott	Nov	Dic	Gen	Feb	Mar	Apr	Mag	Giu	Lug	Ago
Lineare	-0.02	0.36	0.24	0.67	0.55	0.64	0.55	0.07	0.26	0.20	0.64	0.71
Lineare robusto (Huber)	-0.02	0.34	0.24	0.68	0.55	0.64	0.56	0.10	0.27	0.21	0.64	0.70
Lineare avanzato (Cook)	0.39	0.54	0.22	0.69	0.62	0.66	0.67	0.10	0.30	0.03	0.62	0.71
Ridge	-0.00	0.34	0.25	0.68	0.53	0.63	0.54	0.03	0.26	0.18	0.64	0.72
Lasso	0.01	0.35	0.23	0.69	0.53	0.64	0.53	0.05	0.26	0.20	0.62	0.71
Polinomiale (grado 2)	0.37	0.38	0.27	0.68	0.60	0.62	0.52	-0.03	0.25	0.11	0.65	0.67
Polinomiale (grado 3)	-1.00	0.26	0.23	0.68	0.58	0.58	0.49	-0.08	0.23	-1.17	0.59	0.60
Random Forest	0.16	0.03	0.11	0.55	0.41	0.46	0.45	-0.36	-0.10	-0.11	0.57	0.52
Gradient Boosting	0.09	0.07	0.07	0.18	0.13	0.12	0.15	-0.03	0.02	-0.03	0.15	0.14
SVR (lineare)	-0.36	0.33	0.19	0.66	0.54	0.59	0.49	0.09	0.25	0.27	0.62	0.71
SVR (polinomiale)	-0.98	-0.01	0.13	0.49	0.14	0.41	0.48	-0.00	0.20	0.17	0.65	0.68
SVR (RBF)	0.33	0.13	0.28	0.63	0.53	0.41	0.37	-0.21	0.07	-0.35	0.53	0.34
KernelRidge	0.38	0.40	0.30	0.68	0.61	0.65	0.52	-0.05	0.27	0.14	0.60	0.69

Tabella 3.12: Risultati della calibrazione PM₁₀ mese per mese (R^2)

Modello	Set	Ott	Nov	Dic	Gen	Feb	Mar	Apr	Mag	Giu	Lug	Ago
Lineare	6.40	8.47	14.19	13.61	15.06	10.52	7.24	4.55	2.38	4.55	2.44	3.20
Lineare robusto (Huber)	6.35	8.54	14.19	13.53	15.21	10.54	7.26	4.46	2.41	4.52	2.48	3.21
Lineare avanzato (Cook)	5.02	6.60	13.37	12.06	13.86	10.26	6.09	4.22	2.24	3.46	1.94	2.88
Ridge	6.42	8.60	14.04	13.51	15.48	10.53	7.30	4.56	2.39	4.63	2.45	3.15
Lasso	6.43	8.52	14.22	13.51	15.40	10.64	7.32	4.62	2.41	4.59	2.47	3.23
Polinomiale (grado 2)	5.04	8.33	13.79	13.37	14.06	10.75	7.39	4.70	2.42	4.86	2.37	3.41
Polinomiale (grado 3)	6.04	8.84	14.11	13.34	14.22	11.44	7.67	4.77	2.47	6.54	2.50	3.82
Random Forest	5.80	10.29	15.10	15.86	17.13	12.85	8.01	5.39	2.91	5.35	2.56	4.23
Gradient Boosting	6.16	10.34	16.02	22.09	21.02	16.97	10.27	4.77	2.80	5.51	3.87	5.83
SVR (lineare)	7.25	8.66	14.60	13.95	15.03	11.13	7.55	4.39	2.47	4.52	2.51	3.23
SVR (polinomiale)	8.21	10.45	15.33	17.05	20.15	13.55	7.70	4.68	2.50	4.57	2.40	3.37
SVR (RBF)	5.11	9.84	13.65	14.44	15.01	13.25	8.53	5.03	2.65	6.12	2.73	4.88
KernelRidge	5.04	8.23	13.60	13.52	14.06	10.59	7.31	4.79	2.37	4.65	2.52	3.30

Tabella 3.13: Risultati della calibrazione PM₁₀ mese per mese (RMSE)

Infine, la tabella 3.14 riporta i coefficienti delle curve di regressione ottenuti applicando i modelli al dataset PM₁₀. Anche in questo caso i coefficienti a , b , c , d , e si riferiscono ad una equazione polinomiale generica del tipo $y = a + bx + cx^2 + dx^3 + ex^4$.

Modello di regressione	a	b	c	d	e
Lineare	14.11	13.61	/	/	/
Lineare robusto (Huber)	14.37	13.25	/	/	/
Lineare avanzato (Cook)	10.98	11.32	/	/	/
Lasso	13.98	13.75	/	/	/
Ridge	13.93	13.80	/	/	/
Polinomiale (grado 2)	6.78	25.69	-2.58	/	/
Polinomiale (grado 3)	5.80	28.01	-3.72	0.14	/
Polinomiale (grado 4)	4.92	30.76	-5.97	0.76	-0.05

Tabella 3.14: Coefficienti della calibrazione PM₁₀

3.4 Validazione

Non sempre un modello che si adatta bene ai dati avrà successo anche nell'applicazione finale o nella previsione di nuove osservazioni: non vi è alcuna garanzia che l'equazione che fornisce il miglior adattamento ai dati esistenti sarà un predittore di successo. Fattori influenti che erano sconosciuti durante la fase di costruzione del modello possono influenzare in modo significativo le nuove osservazioni, rendendo le previsioni quasi inutili.

Per questo, la corretta convalida di un modello sviluppato per prevedere nuove osservazioni dovrebbe sempre implicare una fase di validazione fatta sul campo (*field validation*).

Una delle procedure per validare un modello di regressione consiste nel testarlo su dati nuovi, non visti in fase di training, e valutare se le prestazioni predittive del modello siano effettivamente in linea con le aspettative.

Per svolgere la fase di validazione è stato quindi considerato il periodo da 01/09/21 a 20/11/21, non utilizzato per allenare il modello, di cui sono disponibili sia le misurazioni della centralina SMART16 di AirQino che della stazione ARPAT di Capannori (LU).

Le tabelle 3.15 e 3.16 riassumono i risultati del confronto di misurazione della concentrazione di $\text{PM}_{2.5}$ e PM_{10} della centralina SMART16 AirQino nelle due configurazioni (con sensori calibrati e non calibrati).

	R^2	RMSE ($\mu\text{g}/\text{m}^3$)
Calibrato	0.633	4.522
Non calibrato	0.582	4.954

Tabella 3.15: Statistiche a confronto nella misurazione delle concentrazioni medie (8h) di $\text{PM}_{2.5}$ tra stazione AirQino calibrata e non calibrata rispetto alla stazione di riferimento ARPAT (dal 1 settembre 2021 al 20 novembre 2021).

	R^2	RMSE ($\mu\text{g}/\text{m}^3$)
Calibrato	0.494	5.711
Non calibrato	0.431	6.942

Tabella 3.16: confronto nella misurazione delle concentrazioni medie (8h) di PM_{10} tra stazione AirQino calibrata e non calibrata rispetto alla stazione di riferimento ARPAT (dal 1 settembre 2021 al 20 novembre 2021).

Le figure 3.41 e 3.42 mostrano l'andamento temporale delle misurazioni di $\text{PM}_{2.5}$ e PM_{10} , calibrate e non calibrate, della stazione AirQino rispetto

alle osservazioni della stazione ARPAT.

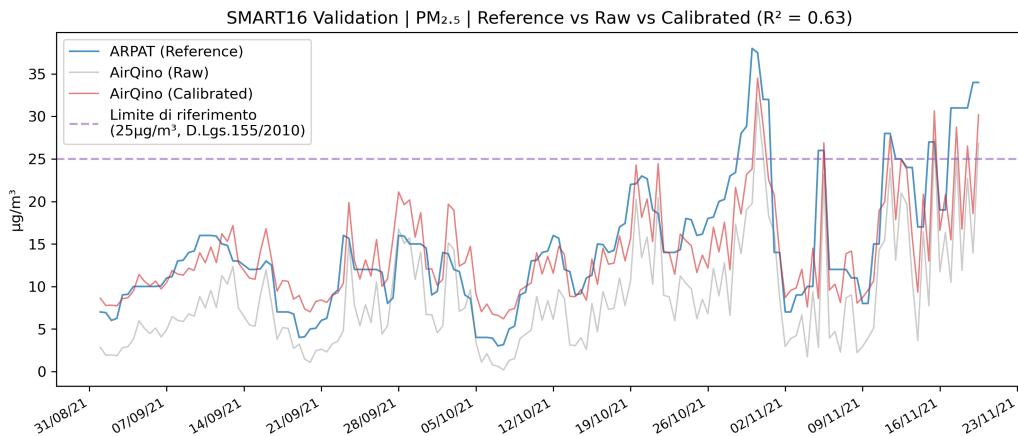


Figura 3.41: Confronto tra l'andamento temporale delle concentrazioni di PM_{2.5} misurate a Capannori (LU) dalla stazione AirQino calibrata e non calibrata con riferimento alla stazione ARPAT (medie a 8h, periodo dal 1 settembre 2021 al 20 novembre 2021)

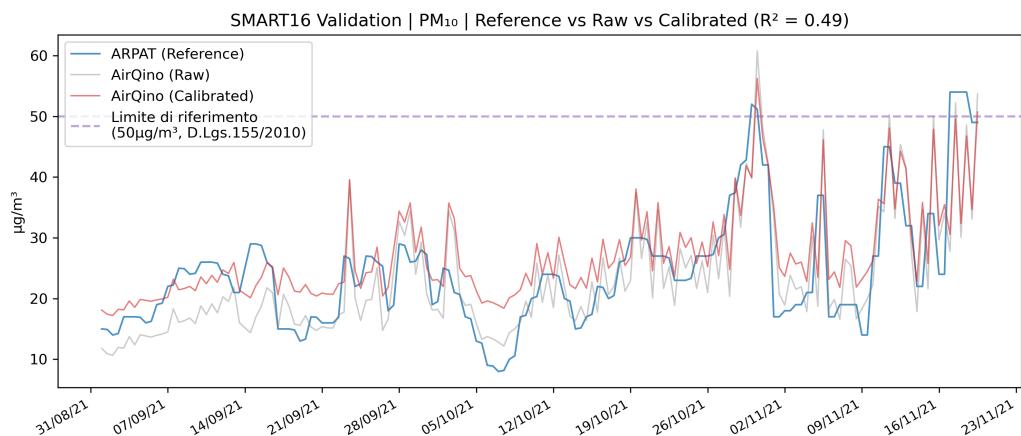


Figura 3.42: Confronto tra l'andamento temporale delle concentrazioni di PM₁₀ misurate a Capannori (LU) dalla stazione AirQino calibrata e non calibrata con riferimento alla stazione ARPAT (medie a 8h, periodo dal 1 settembre 2021 al 20 novembre 2021)

3.5 Discussione

Tutti risultati della fase di calibrazione hanno evidenziato un miglioramento generale nella misura delle concentrazioni di NO_2 e PM, sia nei valori di R^2 (da 0.582 a 0.633 per $\text{PM}_{2.5}$ e da 0.431 a 0.494 per PM_{10}) che per RMSE (da 4.954 a 4.522 $\mu\text{g}/\text{m}^3$ per $\text{PM}_{2.5}$ e da 6.942 a 5.711 $\mu\text{g}/\text{m}^3$ per PM_{10}). I modelli di regressione migliori sono riportati nella tabella 3.17, insieme ai coefficienti corrispondenti. In tutti i casi, il modello di regressione lineare *avanzata*, con rilevamento e rimozione di outlier tramite distanza di Cook (3.2.4.3), ha dimostrato essere il più efficiente.

Inquinante	Modello di regressione	a	b	R^2	RMSE ($\mu\text{g}/\text{m}^3$)
NO_2	Lineare avanzato (Cook)	6.84	9.56	0.373	9.447
$\text{PM}_{2.5}$	Lineare avanzato (Cook)	10.31	7.67	0.801	5.894
PM_{10}	Lineare avanzato (Cook)	10.98	11.32	0.785	7.103

Tabella 3.17: Riepilogo dei migliori modelli di regressione, coefficienti ($y = a + bx$) e risultati rilevati dalla calibrazione di sensori NO_2 , $\text{PM}_{2.5}$ e PM_{10}

Dai confronti sull'andamento temporale (figure 3.41 e 3.42) risulta evidente che gran parte delle osservazioni ARPAT $\text{PM}_{2.5}$ e PM_{10} vengono sotto- stimate dai sensori AirQino non calibrati. Questa discrepanza viene tuttavia ridotta significativamente dal processo di calibrazione: in molti casi il segnale calibrato infatti "imita" in maniera più fedele il riferimento.

Modelli di regressione polinomiale così come altri modelli non lineari (Random Forest, Gradient Boosting, SVR, KernelRidge) non hanno portato miglioramenti rispetto ai modelli lineari. Questo indica che la strada migliore sia l'utilizzo di tecniche matematiche e statistiche avanzate, in grado di rilevare possibili relazioni non lineari tra segnali del sensore e dati di

riferimento (come grafici di residui), che possono fornire informazioni utili su come migliorare il modello (ad esempio con la regressione robusta).

In generale, i risultati per i $\text{PM}_{2.5}$ e PM_{10} sono risultati di gran lunga migliori rispetto alla calibrazione dei sensori di NO_2 . Questo si può ricondurre in parte alla diversa frequenza dei dati (medie orarie per NO_2 , medie a 8h per i PM), e in parte anche alla natura e alle differenze dei sensori stessi (vedi 1.2.1), questi infatti hanno un principio fisico di funzionamento diverso (SDS011 per i PM è ad infrarossi, mentre il MiCS-2714 per NO_2 funziona per ossidazione).

Infine, i risultati per $\text{PM}_{2.5}$ si sono dimostrati essere sempre tendenzialmente migliori dei corrispettivi PM_{10} . Questo risulta in linea con quanto riportato in studi correlati ([16]) e sistemi low-cost simili ([17], [18], [19]).

Capitolo 4

Interfaccia di calibrazione

4.1 Motivazioni

...

4.2 Tecnologie

...

4.2.1 Backend

...

4.2.2 Frontend

...

4.3 Funzionamento

...

4.4 Autenticazione

Keycloak è un'identità federata open source, sviluppata da Red Hat. Può essere utilizzata per gestire l'autenticazione di utenti e servizi in ambienti cloud e on-premise. I principali vantaggi di Keycloak sono la scalabilità, l'affidabilità e la flessibilità.

Keycloak include un server e un agente. L'agente è installato sulle applicazioni che richiedono l'autenticazione, mentre il server gestisce tutte le richieste di autenticazione. Quando un utente tenta di accedere a una applicazione protetta da Keycloak, l'agente verifica se l'utente è autenticato e, in caso affermativo, fornisce le credenziali appropriate all'applicazione.

4.5 CI e deploy automatico

Continuous integration è una metodologia di sviluppo software che prevede il continuo e costante integrazione dei cambiamenti effettuati dai developer all'interno di un codice sorgente.

La continuous integration ha lo scopo di evitare problemi di sincronizzazione tra gli sviluppatori, riducendo il numero di bug rilevati in fase di testing e aumentando la qualità del codice prodotto.

I principali vantaggi della continuous integration sono:

- riduzione del numero di bug rilevati in fase di testing;
- aumento della qualità del codice prodotto;
- maggiore sincronizzazione tra gli sviluppatori;
- minor rischio di collisioni tra i cambiamenti effettuati dagli sviluppatori.

Jenkins è uno strumento open source di continuous integration. Jenkins permette di automatizzare il processo di integrazione dei cambiamenti effettuati dai developer.

tuati dai developer all'interno di un codice sorgente, eseguendo una serie di controlli per verificarne la correttezza.

Jenkins può essere utilizzato per gestire una varietà di progetti, tra cui sviluppo software, testing, build, deployment e automazione dei processi.

Conclusioni e sviluppi futuri

...

Bibliografia

- [1] G. Gualtieri, F. Camilli, A. Cavaliere, T. De Filippis, F. Di Gennaro, S. Di Leonardo, F. Dini, B. Gioli, A. Matese, W. Nunziati, L. Rocchi, P. Toscano, C. Vagnoli, and A. Zaldei, “An integrated low-cost road traffic and air pollution monitoring platform to assess vehicles’ air quality impact in urban areas,” *Transportation Research Procedia*, vol. 27, pp. 609–616, 2017. 20th EURO Working Group on Transportation Meeting, EWGT 2017, 4-6 September 2017, Budapest, Hungary.
- [2] S. Insausti, *PostgreSQL Streaming Replication - a Deep Dive*. 2018.
- [3] S. D’Amilo, *Progetto e implementazione di un’architettura innovativa per la gestione e visualizzazione dei dati IoT di una filiera produttiva*. 2021.
- [4] E. Belluco, *Excel per la statistica*. Franco Angeli, 2005.
- [5] M. S. Paoletta, *Linear Models and Time-Series Analysis*. Wiley, 2019.
- [6] P. Pozzolo, *Analisi dei residui del modello di regressione lineare*. 2020.
- [7] D. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*. Wiley, 2012.

- [8] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied Linear Statistical Models*. Chicago: Irwin, 1996.
- [9] R. D. Cook, *Detection of Influential Observation in Linear Regression*. Taylor and Francis, Ltd., 1977.
- [10] J. Fox, *Applied Regression Analysis and Generalized Linear Models*. Sage Publications, 2015.
- [11] G. Spanò, *Lasso vs Ridge Regression*. 2017.
- [12] A. Felice, *Applicazione di modelli di Machine Learning ad Albero Decisionale con R per il Credit Scoring nel settore elettronico*. 2021.
- [13] T. K. Ho, *Random decision forests*. 1995.
- [14] C. Cortes, *Support-Vector Networks*. 1995.
- [15] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- [16] A. Cavalieri, F. Carotenuto, F. Di Gennaro, B. Gioli, G. Gualtieri, F. Martelli, A. Matese, P. Toscano, C. Vagnoli, and A. Zaldei, “Development of low-cost air quality stations for next generation monitoring networks: Calibration and validation of pm2.5 and pm10 sensors,” *Sensors*, vol. 18, no. 9, 2018.
- [17] N. Zíková, P. Hopke, and A. Ferro, “Evaluation of new low-cost particle monitors for pm2.5 concentrations measurements,” *Journal of Aerosol Science*, vol. 105, 11 2016.

- [18] A. Mukherjee, L. G. Stanton, A. R. Graham, and P. T. Roberts, “Assessing the utility of low-cost particulate matter sensors over a 12-week period in the cuyama valley of california,” *Sensors*, vol. 17, no. 8, 2017.
- [19] N. Zikova, M. Masiol, D. C. Chalupa, D. Q. Rich, A. R. Ferro, and P. K. Hopke, “Estimating hourly concentrations of pm2.5 across a metropolitan area using low-cost particle monitors,” *Sensors*, vol. 17, no. 8, 2017.