Università degli Studi di Milano-Bicocca

Data Management

Final Project

Impatto del lockdown sulla qualità dell'aria in Lombardia

Autori:
Stefano Boldrini - 838059 - s.boldrini2[at]campus.unimib.it
Andrea Corvaglia - 802487 - a.corvaglia3[at]campus.unimib.it
Marco Lauria - 794839 - m.lauria2[at]campus.unimib.it
Giorgio Ottolina - 838017 - g.ottolina1[at]campus.unimib.it

10 settembre 2020

Indice

Sommario

In questo lavoro viene proposta un'analisi descrittiva dell'impatto del blocco dovuto al CoViD-19 sulla qualità dell'aria in Lombardia. In particolare viene descritta un'architettura per l'acquisizione, l'integrazione e la gestione di dati in *streaming* da sensori. Nell'analisi si tiene anche conto dei livelli di precipitazione rilevati da sensori posizionati indipendentemente. Proponiamo contestualmente una strategia di integrazione spaziale di questi. La rilevanza statistica dell'impatto è valutata confrontando i valori dell'anno corrente con una serie storica dei 5 anni precedenti. Gli strumenti utilizzati sono Python per l'elaborazione dei dati di riferimento e per l'acquisizione in *streaming* insieme ad Apache Kafka, mentre la successiva integrazione e selezione dei dati avviene sulla piattaforma Apache NiFi,appoggiandosi a HBase e MongoDB per lo *storage* e le successive aggregazioni tramite *query* per la produzione delle visualizzazioni con Tableau.

1 Introduzione

A partire dal 2005, anno in cui venne siglato il protocollo di Kyoto, il problema dell'inquinamento atmosferico e la minaccia che esso rappresenta all'ambiente e alla salute umana è diventato sempre più un argomento di dibattito politico e sociale. Un numero crescente di governi, aziende, associazioni non-profit, attivisti e persone comuni pongono al centro dell'attenzione le gravi conseguenze che l'inquinamento atmosferico genera sia sulla sostenibilità del mondo in cui viviamo sia sulle conseguenze che questo comporta sulla salute umana. I paesi maggiormente industrializzati e sviluppati al mondo, tra cui l'Italia, sono da ritenersi i principali responsabili e ciò si riconduce ad attività industriali, traffico veicolare e aereo, raffinazioni, impianti di riscaldamento, produzione di energia alimentati a gasolio, carbone e oli combustibili, che sono le principali fonti di inquinamento atmosferico. L'urgenza di adottare soluzioni e politiche volte alla salvaguardia del territorio, salute umana e sostenibilità ambientale sono care a numerose amministrazioni locali, tra cui la regione Lombardia, che, sia per l'intensivo sfruttamento del territorio per attività industriali e sia per la sua conformazione territoriale con un'altissima concentrazione di popolazione rappresenta da decenni una delle più inquinate zone d'Europa a livello atmosferico. Inoltre, i livelli di questo inquinamento sono influenzati dalle condizioni del meteo, gli andamenti climatici stagionali, piogge e venti. Ad esempio l'aria stagnante, l'inversione di temperatura o le basse velocità del vento sono fenomeni tipici della pianura Padana, i quali fanno sì che l'inquinamento atmosferico rimanga in un'area locale, causando alte concentrazioni di sostanze inquinanti. L'intensivo monitoraggio negli anni delle condizioni della qualità dell'aria e del meteo da parte di ARPA Lombardia, l'agenzia regionale per la protezione dell'ambiente, rendono possibile un'analisi che consente di valutare se il momento storico senza precedenti che l'Italia sta affrontando con l'imposizione del *lockdown* dettato dall'emergenza CoViD-19 sta effettivamente avendo un impatto sulla qualità dell'aria, come in molte sedi e occasioni sostenuto, oppure no. Il blocco delle attività produttive, industriali e non, e il blocco del traffico automobilistico e aereo hanno realmente impattato positivamente la qualità dell'aria? Oppure l'inquinamento creato da impianti domestici di riscaldamento, le attività produttive agro-alimentari sempre in funzione e le condizioni meteorologiche del periodo non hanno consentito un miglioramento rilevante? Inoltre, osservando le zone in cui questo impatto è stato maggiore o diversificato rispetto alle sostanze inquinanti, è possibile comprendere quali e quanti aspetti della qualità dell'aria sono legati al traffico cittadino?

Queste sono le domande di ricerca che il progetto vuole provare a risolvere.

Per poter rispondere a queste domande è preliminare chiedersi quali siano, tra le misure disponibili,le sostanze che è necessario monitorare, quali di queste hanno un maggiore impatto negativo sulla qualità dell'aria e quali di queste sono generate dalle attività rimaste attive oppure sospese.
`https://www.dati.lombardia.it/]` [6]. ARPA Lombardia attraverso un sistema di sensori dislocato sul territorio regionale registra periodicamente i valori di 15 sostanze inquinanti. Tra questi, ammoniaca, benzene, benzo(a)pirene, blackcarbon, monossido di azoto, ossidi di azoto e ozono non sono stati presi in considerazione per varie motivazioni, tra cui la non particolare rilevanza come sostanza inquinante o la mancanza di una sufficiente diffusione dei sensori corrispondenti. Questo al fine di con-

sentire un'analisi dettagliata e quanto più omogenea possibile. I principali inquinanti che condizionano la qualità dell'aria sono:

- PM2.5: Il particolato atmosferico PM 2,5 ha un rilevante impatto ambientale sul clima, sulla visibilità, sulla contaminazione di acqua e suolo, sugli edifici e sulla salute di tutti gli esseri viventi. Soprattutto gli effetti che può avere sull'uomo destano maggiore preoccupazione e interesse. In particolare, le particelle più piccole riescono a penetrare più a fondo nell'apparato respiratorio.

- PM 10: rappresenta uno degli inquinanti a maggiore criticità, specialmente nel contesto urbano. La differenza sostanziale tra PM2.5 e PM10 sta nella diversa dimensione del loro diametro, dove per le PM10 è inferiore a 10 μm (micron) e possono essere inalate e penetrare nel tratto superiore dell'apparato respiratorio, dal naso alla laringe. PM2.5 invece ha un diametro inferiore a 2,5 μm e possono essere respirate e spingersi nella parte più profonda dell'apparato, fino a raggiungere i bronchi. PM10 e PM2.5 sono prodotti da una vasta gamma di processi industriali attraverso la movimentazione di materiali sfusi, la combustione e la lavorazione dei minerali. Le industrie che utilizzano questi processi includono laterizi, raffinerie, cementifici, produzione di ferro e acciaio, cave e centrali elettriche a combustibile fossile

- Particolato totale sospeso: Il particolato sospeso (Polveri Totali Sospese, P.T.S.) è costituito dall'insieme di tutto il materiale non gassoso in sospensione nell'aria. La natura delle particelle è molto varia: ne fanno parte le polveri sospese, il materiale organico disperso dai vegetali (pollini e frammenti di piante), il materiale inorganico prodotto da agenti naturali (vento e pioggia), dall'erosione del suolo o da manufatti (frazioni più grossolane). Nelle aree urbane il materiale particolato può avere origine da lavorazioni industriali (cantieri edili, fonderie, cementifici), dall'usura dell'asfalto, degli pneumatici, dei freni e delle frizioni e dalle emissioni di scarico degli autoveicoli, in particolare quelli con motore diesel.

- Metalli pesanti: nickel , cadmio, arsenico e piombo sono inquinanti spesso presenti nell'aria a seguito di emissioni provenienti da diversi tipi di attività industriali.

- Monossido di carbonio: il monossido di carbonio (CO) è un gas inodore, incolore, infiammabile e molto tossico. È prodotto da reazioni di combustione in difetto di aria. È un inquinante prevalentemente primario, emesso direttamente da tutti i processi di combustione incompleta dei composti carboniosi. Le sorgenti possono essere di tipo naturale (incendi, vulcani, emissioni da oceani, etc.) o di tipo antropico (traffico veicolare, riscaldamento, attività industriali come la produzione di ghisa e acciaio, raffinazione del petrolio, lavorazione del legno e della carta, etc.).

- Biossido di zolfo: la presenza in atmosfera è dovuta soprattutto alla combustione di combustibili fossili (carbone e derivati del petrolio) in cui lo zolfo è presente come impurezza. In natura è prodotto prevalentemente dall'attività vulcanica mentre le principali sorgenti antropiche sono gli impianti per il riscaldamento e la produzione di energia alimentati a gasolio, carbone e oli combustibili

- Biossido di azoto: gli ossidi di azoto (NOX) vengono prodotti da tutti i processi di combustione ad alta temperatura (impianti di riscaldamento, motori dei veicoli, combustioni industriali, centrali di potenza, etc.), per ossidazione dell'azoto atmosferico e, in piccola parte, per ossidazione dei composti dell'azoto contenuti nei combustibili. Il biossido di azoto è un inquinante per lo più secondario, che si forma in atmosfera principalmente per ossidazione del monossido di azoto (NO).

Il virus ha colpito con maggiore violenza proprio la regione Lombardia, cioè quella più sviluppata in termini di PIL. Si può tracciare un parallelo con la Roma imperiale: lì era collocato il principale agglomerato di assembramenti urbani, resi insalubri non tanto dalle condizioni igienico-sanitarie di tale epoca storica, quanto piuttosto dall'elevato livello d'inquinamento; lì i crocevia di uomini e merci

erano il riflesso lampante di un notevole picco di globalizzazione. Recentemente, uno studio condotto dai ricercatori della TH Chan School of Public Health dell'Università di Harvard[7] ha decretato che livelli più elevati di PM2.5 contribuiscono a diffondere in maggior misura il virus e restano dunque associati a tassi di mortalità più alti. ARPA, l'Agenzia Regionale per la Protezione dell'Ambiente, ha diffuso i risultati di uno studio[8] condotto in collaborazione con la Regione Lombardia finalizzato ad osservare il mutamento dei fattori di pressione e l'andamento dati di qualità dell'aria causato dai provvedimenti di restrizione applicati per porre rimedio all'emergenza sanitaria in corso (stop completo del traffico veicolare, chiusura di moltissime aziende, persone relegate nelle loro abitazioni e attività agricole abbandonate a sé stesse). La qualità dell'aria misurata, e dunque la presenza di inquinanti nell'atmosfera, è strettamente correlata ad una molteplicità di fattori tali da influenzare, anche notevolmente, i dati misurati. E' importante ricordare poi che l'analisi di tali valori deve tener conto di fattori primari e secondari: le emissioni sono di certo una delle cause, ma essa è impattata anche dalle variazioni delle condizioni climatiche che possono determinare diffusione, diminuzione, dispersione, trasporto e accumulo degli inquinanti, così come fenomeni chimico-fisici che si verificano nell'atmosfera determinando meccanismi di formazione e trasformazione delle sostanze presenti in aria. I ritardi e le incomprensioni con cui l'occidente ha reagito alla diffusione del Covid-19 al proprio interno dimostrano senza ombra di dubbio che fino a pochissimo tempo fa le epidemie erano vissute come fenomeni del passato o, al massimo, confinate nel mondo meno sviluppato (se, per esempio, davvero gravissime come l'ebola). In poche parole, venivano considerate alla stregua di un concetto astratto e per lo più remoto, che non aveva a che fare con la realtà quotidiana.

2 Dataset

I *dataset* provengono dal portale open data Lombardia e sono forniti dall'Agenzia Regionale per la Protezione dell'Ambiente (ARPA Lombardia), che li ha raccolti negli anni attraverso sensori disposti in postazioni sparse per tutto il territorio della regione.
Si considerano due tipi di file contenenti le misure su base oraria per la qualità dell'aria e ogni $10'$ per il meteo. Le misure però non esplicitano nulla sulla natura del sensore, che invece è descritto in un ulteriore file che associa ad ogni id sensore tutte le informazioni utili come posizione, tipo di misura e postazione. Quindi, riassumendo, per ogni anno si sono utilizzati due file contenenti le misure e due file contenenti le informazioni dei sensori. Per l'analisi abbiamo considerato l'anno corrente e i quattro anni precedenti, quindi 10 file con le misure e 2 file con le informazioni di tutti i sensori della serie storica.

Le modalità di acquisizione sono state due a seconda del tipo di sorgente.

- Download del file
- Richiesta API Socrata Open Data (SODA) [1]

Fonti:

Qualità Aria:

- link ai dati per il 2020 :
  `https://www.dati.lombardia.it/Ambiente/Dati-sensori-aria/nicp-bhqiÃă`
- link ai dati sulla posizione dei sensori:
  `https://www.dati.lombardia.it/Ambiente/Stazioni-qualit-dell-aria/ib47-atvt`

Meteo:

- link ai dati per il 2020:
  `https://www.dati.lombardia.it/Ambiente/Dati-sensori-meteo/647i-nhxk/data`
- link ai dati sulla posizione dei sensori :
  `https://www.dati.lombardia.it/Ambiente/Stazioni-Meteorologiche/nf78-nj6b`

3 Approccio Metodologico

3.1 Fase Preliminare

Il primo passo preliminare è stato l'analisi delle informazioni sui sensori. Si è valutato quale fosse il numero di dispositivi a seconda dell'inquinante o della grandezza misurata, quali di questi fossero disponibili a partire dal primo anno considerato e tra questi quali fossero attivi tutt'oggi. Parallelamente si è valutata la distribuzione spaziale dei sensori, per studiare una strategia di integrazione spaziale tra dati meteo e dell'aria.
Il flusso di dati principale è quello dell'aria e quindi l'arricchimento si basa sull'aggiunta dei dati riguardanti i sensori meteo. Nella figura 1 si può osservare la distribuzione dei sensori meteo in Lombardia (punti blu), per ragioni di altitudine maggiore, si può notare una presenza più elevata di sensori meteorologici nelle zone alpine della regione, mentre mentre, analizzando i sensori evidenziati in rosso, si può notare una presenza maggiore nei pressi delle aree urbane.
Questa situazione si presta molto bene ad una integrazione spaziale. Per questa si sono valutate due strategie opposte. La prima consisteva nel valutare un dato meteo "provinciale" ovvero mediato tra le stazioni della provincia e poi integrato ai sensori dell'aria in base alla loro appartenenza ad una o l'altra provincia. Questo approccio porta a diverse problematiche, da un lato l'utilizzo di un'infformazione imprecisa per via dell'aggregazione grossolana, utilizzata in modo poco attento, vista la distribuzione disomogenea dei sensori dell'aria. Dall'altro lato, una provincia compare nei sensori della qualità e non in quelli del meteo, portando a complicazioni nella gestione dell'integrazione. La strategia alternativa adottata è stata quella di creare una mappa che associasse ad ogni stazione dell'aria un'unica stazione meteo in modo tale da minimizzarne la distanza.

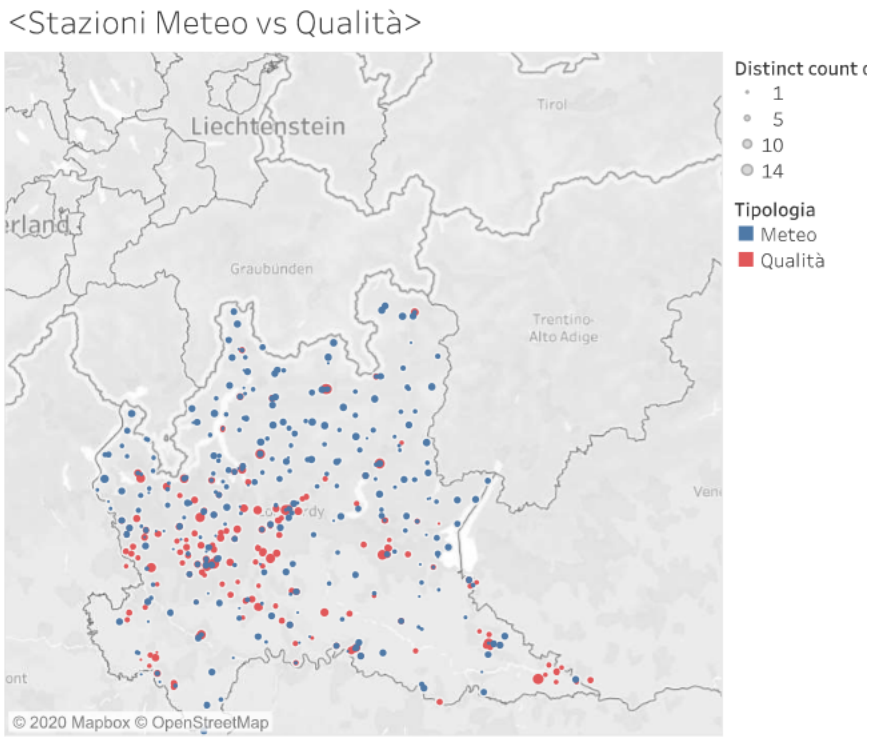

Figura 1: Distribuzione spaziale dei sensori di qualità dell'aria in rosso e meteo in blu.

Questa prima parte è stata implementata attraverso script Python. A partire dalle API con le quali si sono acquisiti i dati. Come detto si sono utilizzate le API fornite da Socrata Open Data, che fornisce la possibilità di ottenere i dati con delle interrogazioni in un linguaggio chiamato "Socrata Query Language" o `SoQL` [2], molto simile al classico SQL. In questo modo è stato possibile filtrare i sensori non attivi nell'intervallo di tempo considerato, creando una tabella con le informazioni esclusivamente dei sensori utili. (sono stati anche selezionati solo i sensori meteo relativi alla pioggia). Vengono mostrati di seguito la parte iniziale dei dei *dataset* dei dati di meteo e aria rispettivamente:

```
In [1]: meteo.head()
Out[1]:
    IdSensore                 Data  Valore Stato  idOperatore
0        2103  01/01/2020 00:00:00     0.0    VA            4
1        2417  01/01/2020 00:00:00     0.0    VA            4
2        2452  01/01/2020 00:00:00     0.0    VA            4
3        2502  01/01/2020 00:00:00     0.0    VA            4
4        4065  01/01/2020 00:00:00     0.0    VA            4

In [2]: aria.head()
Out[2]:
   IdSensore                 Data  Valore Stato  idOperatore
0      10023  01/01/2019 00:00:00 -9999.0   NaN            1
1      10025  01/01/2019 00:00:00 -9999.0   NaN            1
2      11041  01/01/2019 00:00:00 -9999.0   NaN            1
3       5504  01/01/2019 00:00:00    75.6    VA            1
4       5507  01/01/2019 00:00:00    61.0    VA            1
```

I dati mostrati, contenenti le misure dei sensori, sono stati scaricati da open data sotto forma di csv, e per aderire il più possibile ad uno scenario reale di flusso in streaming, non sono stati filtrati se non attraverso Nifi. Ad essere filtrate a priori sono state le informazioni sui sensori, che non sono un dato in tempo reale e sono acquisite, come detto, tramite API. La funzione che mappa stazioni aria in stazioni meteo più prossime, invece, è stata creata utilizzando il modulo di Python `geopy` ed in particolare la distanza geodetica valutata a partire dalle coordinate (latitudine,longitudine) delle postazioni a cui i sensori appartengono. Più precisamente, sono stati aggregati i dati in modo da ridurre le informazioni alle sole stazioni, meno numerosi dei singoli sensori, e per ogni stazione dell'aria è stata valutata la stazione meteo la cui posizione minimizzava la distanza geodetica. A questo punto ad ogni sensore, in base alla sua postazione d'appartenza, è stata associata un postazione meteo di riferimento e la tabella di *merge* è stata salvata come file csv per le successive integrazioni.

Si evidenzia come si siano selezionati solo i sensori che misurano Biossido d'azoto, PM10 o Monossido di carbonio, sulla base di quanto detto nell'introduzione e sull'analisi della diffusione dei vari tipi di sensore. In seguito a questa prima selezione si sono conservate solo le postazioni contenenti il tris completo di sensori considerati[1].

3.2 Acquisizione

A seguito di questa prima parte preliminare si è proceduto con l'organizzazione dell'acquisizione. L'API purtroppo è disponibile solo per il 2020, di conseguenza le misure sono state scaricate dal sito di Open Data Lombardia in formato csv. Per simulare l'acquisizione in *streaming* si sono utilizzati altri due notebook di Python, uno per il meteo e uno per l'aria, che ,anno per anno, streammassero i dati. Si è utilizzato Apache Kafka per l'acquisizione e la gestione delle code, in modo da disaccoppiare la lettura e la scrittura e rendere asincrono il processo, azione necessaria vista la simulazione dei dati in tempo reale. In particolare si è utilizzato il modulo `kafka-python` per gestire il processo tramite script. Più precisamente su Jupyter è stato implementato un KafkaConsumer per tipo di dato, in modo da avere due *topic*, uno per i dati dell'aria e uno per i dati del meteo. La simulazione dello streaming avviene tramite la classe `DictReader` del modulo di `python csv`, che permette di non memorizzare tutto il file csv nella RAM, ma di leggerlo riga per riga, in particolare mappando l'informazione di ogni riga ad un dizionario, permettendo così di trasformare questa informazione in un messaggio di Kafka codificato come JSON.

[1] La procedura completa è esposta nel notebook reference_tables

Il paradigma *object oriented* prevede che le applicazioni siano composte da un insieme di componenti, chiamati oggetti oppure istanze, che collaborano tra loro per svolgere un lavoro o risolvere un problema. Vi sono i *model*, le *view* e i *controller*. I *model* sono oggetti che contengono i dati di un entità, corrispondono ai record delle tabelle, permettono di memorizzare temporaneamente i dati in memoria. Le *view* sono oggetti oppure viste, come ad esempio le pagine html, permettono la visualizzazione oppure l'inserimento di informazioni. I *controller* sono oggetti che hanno il compito di gestire un flusso logico applicativo, all'interno della stessa applicazione ci possono essere anche più *controller*.
I dati prodotti tramite i *notebook* finiscono in Apache Nifi tramite Kafka, infatti i primi due nodi implementati sono proprio due KafkaConsumer in ascolto sui *topic* di meteo e qualità dell'aria. Si è scelto di usare NiFi per diverse ragioni, la prima è la volontà di integrare il più possibile la Velocità e la Varietà eseguendo l'*enrichment* dei dati direttamente in *streaming*, task facilitato dalla gestione delle code e delle priorità dei dati in NiFi. Inoltre quest'ultimo fornisce una comoda interfaccia grafica che permette di stoppare il flusso, direzionare e disambiguare i risultati dei nodi tramite le relazioni, visualizzare il contenuto dei *flowfile* nelle code, tutte possibilità che facilitano la progettazione. In ultimo permette di utilizzare tutta una serie di nodi per eseguire *task* di ogni genere, con una possibilità discretamente ampia di programmazione di questi, come verrà esposto nel seguito della relazione. Un altro vantaggio è stato la possibilità di usare un paradigma detto *Record-Oriented* che permette di processare i dati record per record come dei messaggi, in un modo che è agnostico rispetto al formato dei dati stessi. In questo modo è stato mantenuto un formato json per tutto il *flow*, ma lo si è gestito sfruttando schemi scritti in formato Avro e contenuti in un registro.

3.3 Integrazione, Arricchimento e pulizia dei dati

A questo punto le fasi di integrazione, arricchimento e pulizia dei dati si intrecciano. Il procedimento di arricchimento dei dati, infatti, elimina contestualmente le misure dei sensori che si è deciso di escludere. Questo avviene perché il nodo che esegue l'integrazione in *streaming*, usa come riferimento le tabelle csv con le informazioni dei sensori di cui si è parlato nella fase preliminare. Il nodo LookUpRecord, utilizza una chiave, tramite la quale cercare il match corrispondente nel file. Visto che però nel file l'id dei sensori scartati non compare, il match avviene solo per i sensori effettivamente considerati. Lo schema di integrazione è il seguente:

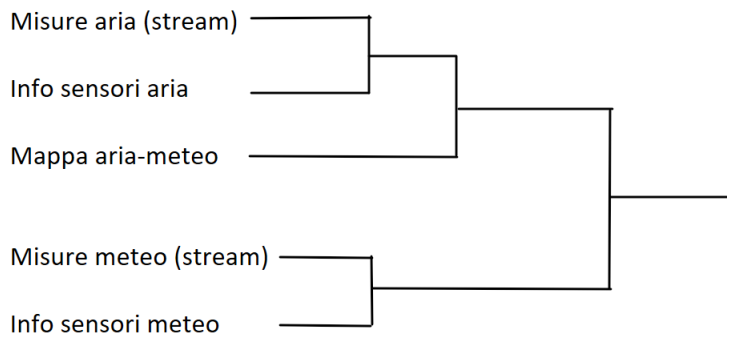

Figura 2: Distribuzione spaziale dei sensori di qualità dell'aria in rosso e meteo in blu.

I dati nel flusso dell'aria vengono arricchiti dalle informazioni sui sensori dell'aria e gli viene associato una postazione meteo a partire dai file csv citati nella fase preliminare. Nello stesso modo il flusso sui dati del meteo viene arricchite con le informazioni sui sensori del meteo. È possibile vedere l'inizio del *dataset* `aria` integrato di seguito:

```
In [1]: aria_integrated.head()
Out[1]:
  idsensore           nometiposensore  ...                 lng                                location
0     12577  Particelle sospese PM2.5  ...  10.015741513336042  {'latitude': '45.13194691285173',...
1      5572         Biossido di Azoto  ...   9.87014407497457  {'latitude': '46.16796681227828',...
2      5631         Biossido di Zolfo  ...   8.880210433125571  {'latitude': '45.46241579110661',...
3      5965         Biossido di Azoto  ...   9.863398419993635  {'latitude': '45.28397715446764',...
4     12610                    Piombo  ...   8.880210433125571  {'latitude': '45.46241579110661',...
```

Il match finale invece ha richiesto un approccio diverso. Questo perché a differenza delle prime quattro integrazioni in cui i dati di riferimento erano pochi e statici, in questo caso l'integrazione doveva avvenire tra due flussi e di conseguenza il riferimento aveva dimensioni molto maggiori ed era dinamico. A questo proposito si è utilizzato uno *storage* per immagazzinare temporaneamente i dati meteo arricchiti, in particolare un database (DB) in MongoDB (come verrà spiegato in seguito), e tramite un nodo di LookUp fare l'arricchimento delle misure dell'aria, sempre in *streaming*, con i dati arricchiti del meteo. La chiave utilizzata è una combinazione tra i campi relativi alla postazione della misura e l'ora della stessa. Questa chiave risulta univoca perché i dati meteo, essendo solo relativi alla pioggia, hanno una corrispondenza biunivoca tra sensore e postazione. E visto che la ricerca avviene record per record, ad ogni misura dell'aria è associata la corretta misura di precipitazione. In realtà in questo modo il processo di *look up* risultava piuttosto lento, cosa che ci ha spinto a trovare un modo per ridurre lo spazio di ricerca, con un *look up* gerarchico, ovvero sfruttando l'*expression language* e gli Attributi in NiFi, si andava a creare nel database una collezione per ogni postazione, così che il numero di confronti nel peggiore dei casi scendesse dal prodotto tra il numero di postazioni per le ore in un anno , alla loro somma. Questo perché creando una collezione col nome della postazione, quando il *flowfile* arrivava al nodo, questo automaticamente cercava solo nella collezione relativa alla postazione del nodo. Purtroppo, per quello che potrebbe essere un bug nel servizio di LookUp in MongoDB, non veniva valutato l'*expression language* nel campo del nome della collezione in fase di lettura, rendendo impossibile il processo. Si è ovviato a questo problema utilizzando un database più semplice vista la struttura dei dati meteo, ovvero HBase. In particolare come indice si è utilizzato direttamente la chiave con cui si faceva il *look up*, si è utilizzata una sola *column family* molto ridotta con solo il valore di precipitazione. In questo modo, insieme con la riduzione della dimensione del dataset attraverso la conservazione dei soli sensori effettivamente usati nel *look up*, si è ovviato al problema della lentezza nel match.

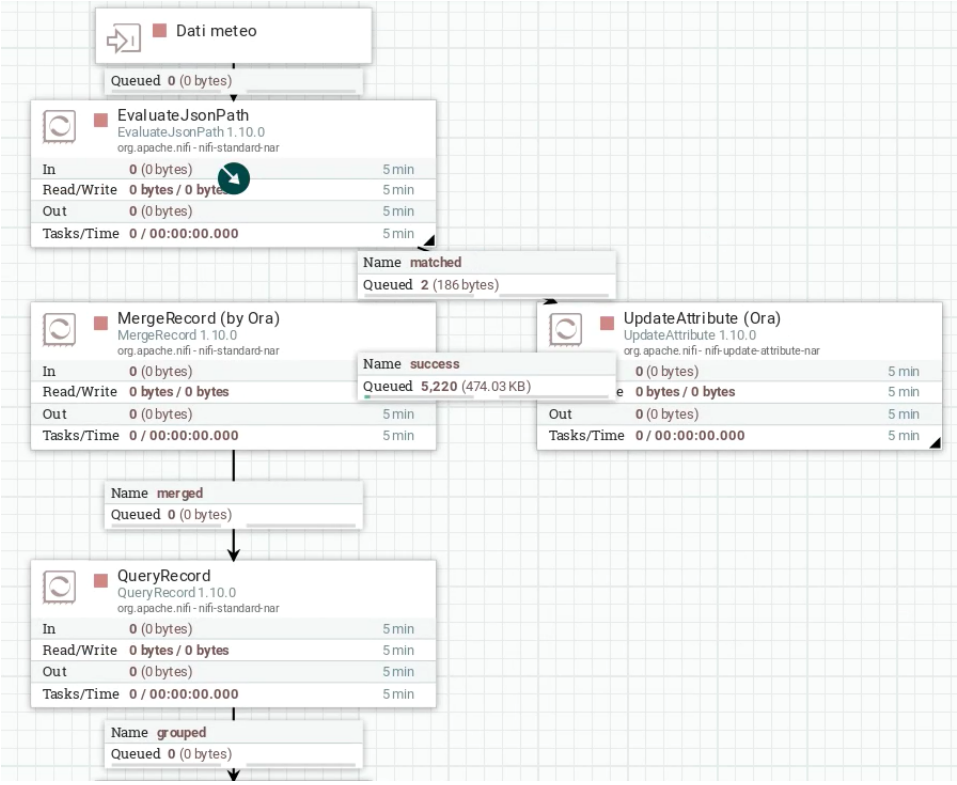

Figura 3: Interno del *group node* chiamato "Aggregazione oraria".

Un altro elemento fondamentale dell'integrazione, oltre alla componente spaziale, è stata la componente temporale. Infatti la granularità dei dati risultava differente tra le due fonti: oraria per la qualità e ogni 10' per la pioggia. Questo ha richiesto l'implementazione di un *group node* che gestisse l'aggregazione oraria dei dati. La difficoltà in questo caso sta nella gestione dello *streaming*, infatti, volendo mantenere i dati in memoria, senza l'uso di un database, era impossibile usare una *query*, perché questa agisce solo all'interno di un *flowfile* e nel caso si volesse fare un *merge* bisogna assicurarsi che tutte e sei le misure necessarie all'aggregazione siano presenti nel nuovo *flowfile* per ognuno dei sensori in gioco. La soluzione è stata la seguente: utilizzando un nodo JsonPath si è estratta l'informazione temporale dal contenuto di ogni *flowfile*, che in questo momento contiene una sola misura, ed è stata poi trasformata in una informazione oraria, così che tutte le misure avvenute all'interno della stessa ora avessero lo stesso valore "ora". Questa informazione è stata dunque inserita tra gli attributi del *flowfile*. A questo punto era possibile raggruppare i *flowfile* in un unico flowfile contenente tutte le misure avvenute alla stessa ora. Il nodo utilizzato è stato MergeRecord con una *policy* di *merge* basata sull'attributo "ora". Il processo di merge avviene nel seguente modo: i *flowfile* arrivano al nodo, il quale li inserisce in dei *bin* in base all'attributo "ora", in attesa che raggiungano una certa dimensione o numero di elementi. Per assicurarsi che tutti gli elementi fossero presenti in un *bin* prima dell'unione, si è utilizzato come *trigger* il numero massimo di *bin*, che associato alla produzione di un dato alla volta dal consumer di kafka insieme con una politica di priorità basata sul processare prima i file più vecchi nel sistema, assicura che i dati arrivino in ordine e che quando viene chiuso un *bin* perché è stato osservato un nuovo orario, tutti i dati del precedente orario sono già nel *bin* corretto. A questo punto ogni *bin* diventa un nuovo *flowfile* su cui si può eseguire una *query* aggregativa e tramite l'attributo si può ristabilire l'informazione temporale che adesso è l'ora della misura. Per assicurarci che i bin non si chiudessero prematuramente il numero massimo di bin è stato portato a 120, così che un occasionale cambio di ordine dei file non portasse a bin incompleti.

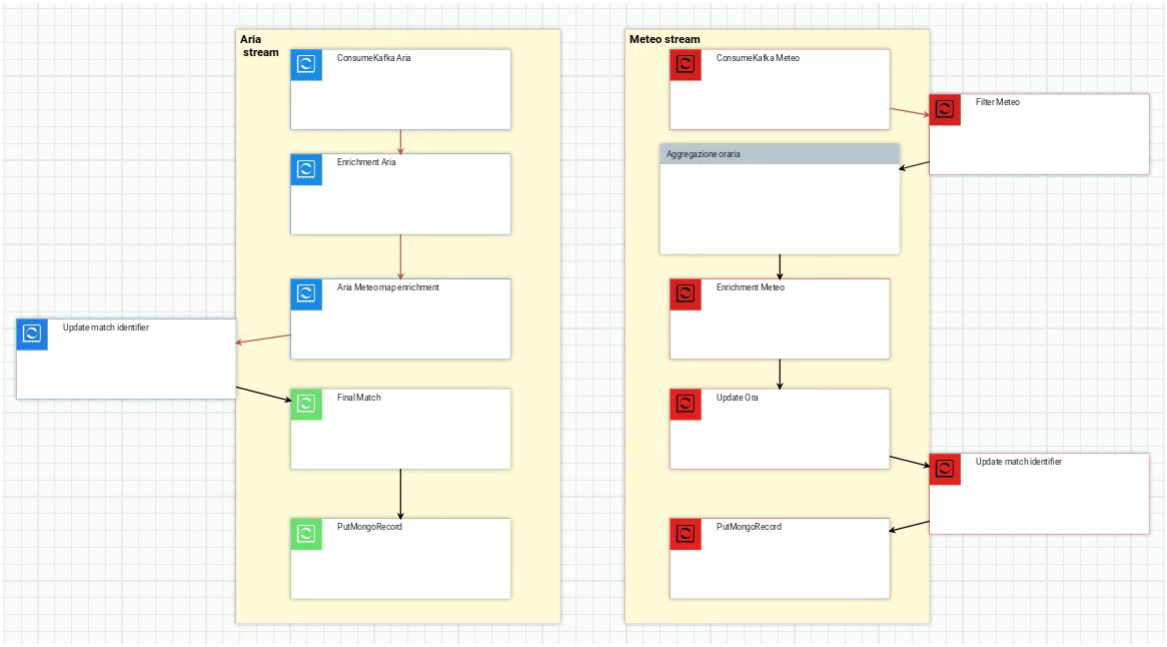

Figura 4: Nifi Flow.

3.4 Memorizzazione ed esplorazione

Per quanto concerne l'immagazzinamento dei dati in uscita dal *flow*, abbiamo scelto di affidarci a un database NoSQl di tipo MongoDB: si tratta di un sistema di gestione basato sui documenti (*Document Based Management System*), in cui i dati sono archiviati in formato JSON e letti tramite indici. Importante da ricordare è che MongoDB non prevede operazioni di *join*, già effettuate in NiFi e discusse in precedenza.
Per *dataset* di grandi dimensioni, risulta molto utile sfruttare le proprietà degli indici. Simili a un indice di un libro (o di una tabella SQL), gli indici di MongoDB consentono di recuperare le informazioni molto velocemente in quanto é possibile sapere quale parte di database non osservare. D'altro canto, l'uso di indici rallenta l'inserimento di nuovi dati (perché appunto devono essere indicizzati) ma velocizza notevolmente le ricerche. Attraverso l'indice si ha la possibilità di identificare in modo univoco il documento, come nei database relazionali attraverso la chiave primaria. Nel caso in cui questo campo non venga inserito manualmente, viene generato automaticamente dal sistema con un ObjectID, cioè un oggetto in formato Binary JSON (BSON). La scelta di questo *DBMS* é stata effettuata anche a seguito di una rapida analisi della struttura dei dati che permetteva una facile creazione di un DB a struttura documentale, la scelta di esportare il *file* dal *flow* di NiFi in formato JSON non é dunque casuale.

Il file JSON ottenuto dal *flow* di NiFi conteneva oltre $1,700,000$ documenti la cui struttura era la seguente[2]:

[2]Si noti che, nel campo `unitamisura`, c'é un errore di formattazione l'unità di misura, nel caso di sensori che misurano `Biossido di Azoto` o `PM10`, é espressa come $\mu g/m^3$, mentre se il sensore misura la quantità di `PM10` l'unità sarà espressa in mg/m^3. Questo problema sarà presente ogni volta che comparirà questo campo del documento.

```
{
        "_id" : ObjectId("5ef48cd62ab79c003a6166b9"),
        "IdSensore" : "5601",
        "Data" : "01/01/2020 00:00:00",
        "Stato" : "VA",
        "idOperatore" : "1",
        "nometiposensore" : "Biossido di Azoto",
        "Valore" : 31.1,
        "unitamisura" : "[]g/m[]",
        "provincia" : "LO",
        "idstazione" : "598",
        "comune" : "San Rocco al Porto",
        "quota" : "50",
        "lat" : "45.081986696978625",
        "lng" : "9.70078817718398",
        "Precipitazioni" : 0,
        "meteo_station" : "898"
}
```


Il file JSON, sono stati importati con la funzione `mongoimport`, il formato del campo `Data` é stato trasformato da stringa, ad avere un formato standard `ISODate`, sfruttando la funzione di MongoDB `$dateFromString`, in modo da poter eseguire interrogazioni al database che considerino un dato intervallo di tempo. Questo step di preprocessing é cruciale perché la possibilità di eseguire interrogazioni relative al tempo fa parte del cuore di quest'analisi. Successivamente sono stati uniti i campi di `comune` e `provincia`, in modo da avere un campo denominato `Address` che contenesse entrambe le informazioni. Infine, per poter eseguire interrogazioni inerenti lo spazio abbiamo trasformato i campi `lng` e `lat`, in modo che assumano valore di un campo `GeoJSON`, costruendo questo nuovo campo `GeoLoc` é stato possibile fare interrogazioni anche in merito all'ubicazione dei sensori o rispetto ad un intorno (ad esempio Milano) date le proprie coordinate geografiche. Dopo aver rinominato i campi per avere un'uniformità linguistica, il nuovo schema aveva questa struttura:


```
{
        "_id" : ObjectId("5f4e71a9adbe1d0039686c75"),
        "lat" : "45.51933498138716",
        "lng" : "9.592009994888425",
        "Date" : ISODate("2020-01-01T13:00:00Z"),
        "GeoLoc" : {
                "type" : "Point",
                "coordinates" : [
                        NumberDecimal("9.592009994888425"),
                        NumberDecimal("45.51933498138716")
                ]
        },
        "Comune" : "Treviglio",
        "Provincia" : "BG",
        "SensorId" : "5790",
        "StationIdAddress" : "592 - Treviglio (BG)",
        "Altitude" : "191",
        "SensorType" : "Monossido di Carbonio",
        "Address" : "Treviglio (BG)",
        "MeteoStationId" : "137",
        "Precipitations" : 0,
        "UnityOfMeasure" : "mg/m[]",
        "Value" : 0.4,
        "StationId" : "592"
}
```


Infine sono stati indicizzati i campi `GeoLoc` e `Date` per ottimizzare le *query* al DB. Attraverso la piattaforma Mongo DB Compass é possibile visualizzare la struttura del Data Base ed anche eseguire interrogazioni e basiche aggregazioni, é stato possibile osservare che il conteggio dei sensori delle varie stazioni per ora varia nell'arco delle giornate avendo un picco a mezzanotte perchè i dati relativi al sensore che misura i PM10 sono giornalieri e non orari come gli altri. generalmente la quantità oraria di sensori per ora varia dai 60 ai 68 sensori.
Analizzando i dati, é stato possibile osservare che la quantità di *missing values* nel dataset era circa il $2,2\%$ (38197 documenti) per l'attributo `Valore`, mentre 561 osservaioni erano mancanti per l'attributo `Precipitazioni`. Per sua costituzione il campo `idOperatore` è uguale per tutto il dataset in quanto ha ereditato dal *dataset* iniziale `meteo`, è stato quindi rimosso come il campo `Stato` che trasportava

l'informazione relativa allo stato di funzionamento del sensore. Essendo quest'ultima già presente nei documenti in concomitanza di un *missing value* nel 100% dei casi (12262). Infine era presente il campo `Stato` senza alcun valore per i restanti 25935 documenti già segnati come "non presenti" nel campo `Valore` È stato inoltre possibile osservare la distribuzione delle stazioni, il *dataset* presenta 32 stazioni, dislocate in 12 province comprendendo un totale di 30 comuni. Mentre il numero di sensori é di 96 di tre tipologie (`Monossido di Carbonio`, `Biossido di Azoto` e `PM10 (SM2005)`) e le stazioni meteorologiche (pluviometri) sono in tutto 29.

4 Visualizzazioni

Per visualizzare il database é stato utilizzato Tableau, il software é stato connesso a MongoDB e sfruttando l'organizzazione dei campi é stata creata una dashboard (`https://public.tableau.com/profile/stefano.boldrini#!/vizhome/ImpattolockdownLombardia/DBAPERTURA?publish=yes`).
Vengono riportate di seguito alcune infografiche.
Questa prima infografica nell'immagine 5 consiste in una mappa interattiva della distribuzione delle stazioni sul territorio regionale, che permette di selezionarle singolarmente e visualizzarne specifiche informazioni anche attraverso menu dropdown:

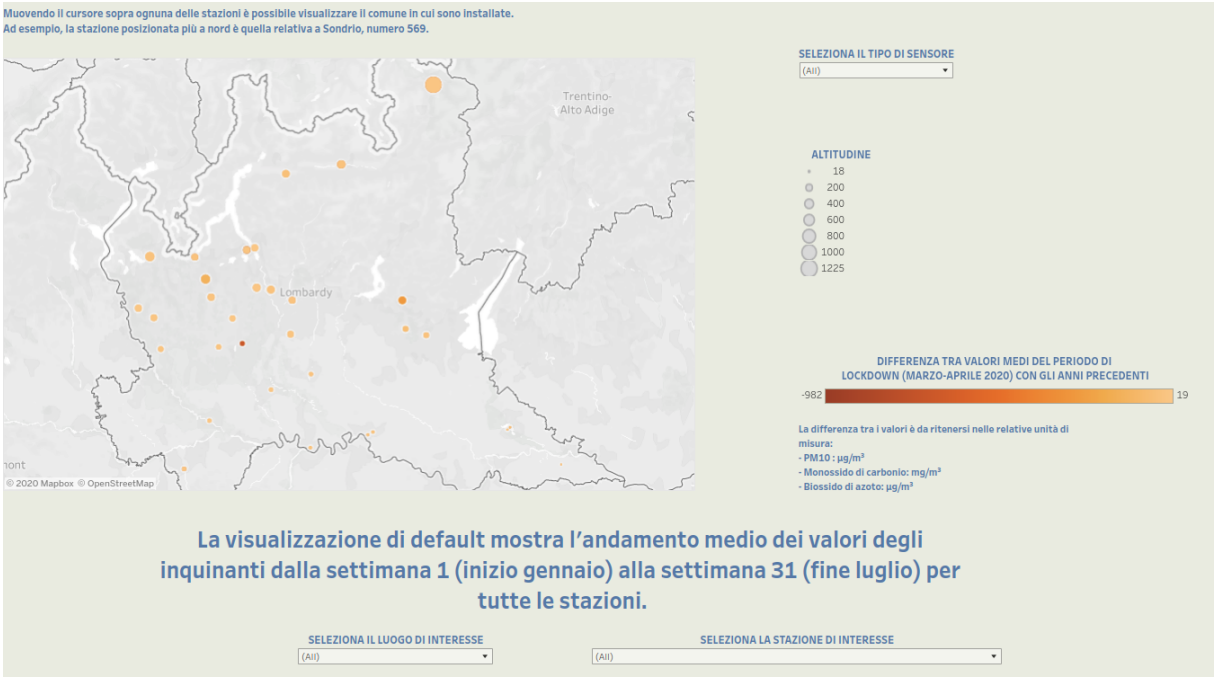

Figura 5: Mappa Stazioni

In seguito possiamo osservare la visualizzazione relativa al primo dei tre inquinanti nell'immagine 6, il PM10 (il medesimo confronto viene effettuato anche per il monossido di azoto e il biossido di carbonio). I valori medi settimanali afferenti al 2020 vengono messi a confronto con quelli degli anni passati, così come le loro differenze:

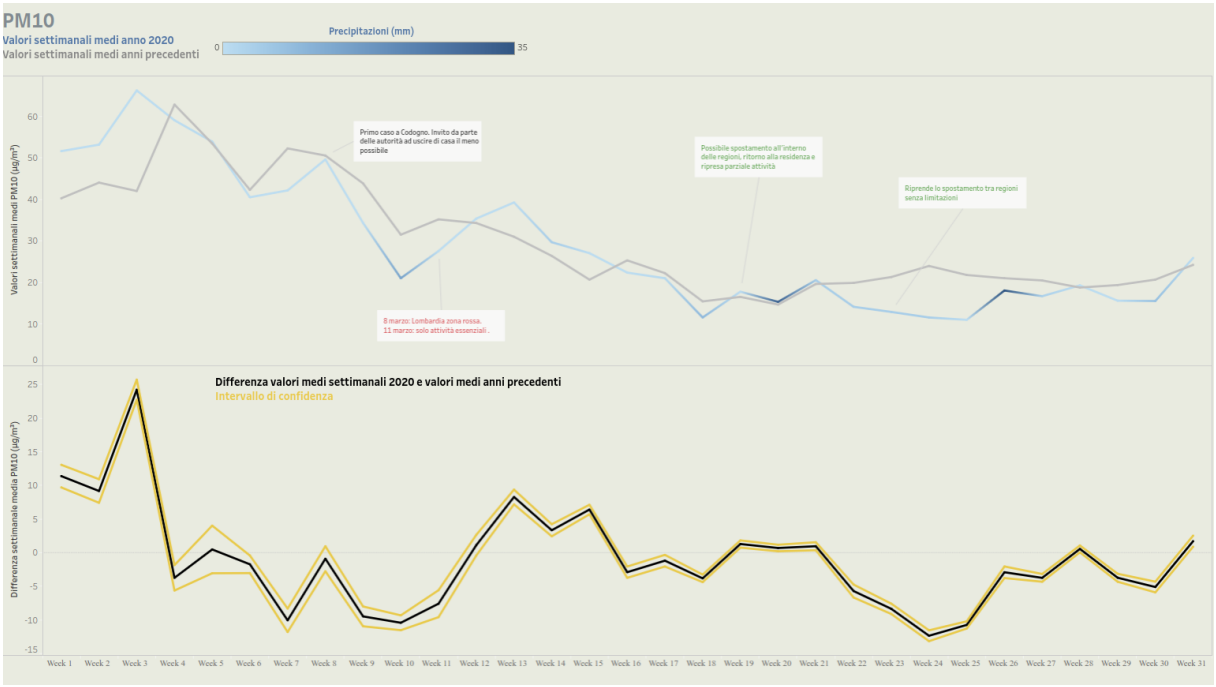

Figura 6: PM10 - Confronto valori settimanali medi e differenze

5 Conclusioni

L'integrazione dei dati proposta presenta vantaggi, quali la possibilità di valutare la qualità dell'aria tenendo conto dell'impatto delle precipitazioni, che spesso possono rendere i dati poco attendibili, come specificato in precedenza. L'utilizzo di una pipeline di integrazione in *streaming* permette di filtrare i dati in tempo reale e di avere per ogni misura la coppia inquinante e precipitazioni alla stessa granularità temporale e relative alla stessa zona geografica. I dati utilizzati, pur provenendo dalla stessa fonte, sono di natura completamente diversa e, per tale ragione, l'aggregazione non é stata semplice. Infatti i sensori si trovavano dislocati sulla superficie della regione in modo indipendente e anche la granularità temporale era diversa. La complessità dell'architettura è elevata, vista la varietà di strumenti impiegati, che però si sono resi necessari al fine di ottimizzare i vari task. Infatti si è scelto un approccio in linea con la cosidetta *polyglot persistence*, in particolare si è usato *Kafka* per salvare dati veloci in acquisizione, *MongoDB* per il database finale e hbase come storage momentaneo per l'integrazione in tempo reale, visto che, grazie alla sua struttura colonnare si prestava ad un recupero più veloce nel *lookup*. Un esame dei dati ci ha permesso di valutare che durante i mesi di *lockdown* si nota che la differenza fra la media settimanale del 2020 degli inquinanti rispetto alla media settimanale degli anni passati è significativa.

Riferimenti bibliografici

[1] Socrata
https://dev.socrata.com/

[2] SoQL documentation
https://dev.socrata.com/docs/queries

[3] Apache NiFi: RecordPath Guide
https://nifi.apache.org/docs/nifi-docs/html/record-path-guide.html

[4] Apache NiFi: User Guide
https://nifi.apache.org/docs/nifi-docs/html/user-guide.html

[5] Apache NiFi: Expression Language Guide
https://nifi.apache.org/docs/nifi-docs/html/expression-language-guide.html

[6] Open Data Lombardia - Aria e Meteo
https://www.dati.lombardia.it/Ambiente/Dati-sensori-aria-2016/7v3n-37f3
https://www.dati.lombardia.it/Ambiente/Dati-sensori-aria-2017/fdv6-2rbs
https://www.dati.lombardia.it/Ambiente/Dati-sensori-aria-2018/4t9j-fd8z
https://www.dati.lombardia.it/Ambiente/Dati-sensori-aria-2019/j2mz-aium
https://www.dati.lombardia.it/Ambiente/Dati-sensori-meteo-2016/kgxu-frcw
https://www.dati.lombardia.it/Ambiente/Dati-sensori-meteo-2017/vx6g-atiu
https://www.dati.lombardia.it/Ambiente/Dati-sensori-meteo-2018/sfbe-yqe8
https://www.dati.lombardia.it/Ambiente/Dati-sensori-meteo-2019/wrhf-6ztd

[7] TH Chan School of Public Health Study
https://www.hsph.harvard.edu/news/hsph-in-the-news/air-pollution-linked-with-higher-covid-19-death-rates/

[8] ARPA Study - Report
https://www.arpalombardia.it/sites/DocumentCenter/Documents/Aria\%20-\%20Relazioni\%20approfondimento/Analisi\%20preliminare\%20QA-COVID19.pdf