



Università degli studi di Bari Aldo Moro

DIPARTIMENTO DI INFORMATICA
Corso di Laurea Triennale in Informatica

CASO DI STUDIO DEL CORSO IN INGEGNERIA DELLA CONOSCENZA

"Sistema di Monitoraggio Ambientale e Controllo della Qualità dell'Aria basato sull'Intelligenza artificiali"



Repository GitHub: <https://github.com/MARJANZAGHARI1993/Icon23-24.git>

Autori :

Zagharitafreshi Marjan 704820 m.zagharitafreshi@studenti.uniba.it

Robles Vincenza 728324 v.robles1@studenti.uniba.it

Anno Accademico 2023-2024

Indice

Obiettivo.....	3
Introduzione	4
Ontologia.....	5
Knowledge Base (KB)	7
Fatti e regole della KB	7
Query Knowledge Base (KB)	8
Aggiornamento Knowledge Base (KB).....	10
Apprendimento Supervisionato.....	11
Scelta del modello.....	13
Sommario.....	13
Strumenti utilizzati.....	14
Decisioni di Progetto.....	15
Valutazione.....	15
Rete Bayesiana.....	20
Sommario.....	20
Strumenti utilizzati.....	20
Decisioni di Progetto.....	21
Valutazione.....	21
Algoritmo di path finding con A*.....	23
Sommario.....	23
Strumenti utilizzati.....	23
Decisioni di Progetto.....	23
Modelli Non Utilizzati.....	24
Valutazione.....	24
Conclusione.....	25
Riferimenti Bibliografici	26

Obiettivo:

Implementare un sistema intelligente di monitoraggio ambientale per valutare la qualità dell'aria e gli inquinanti atmosferici in un'area specifica come città di Bari, fornendo avvisi tempestivi alle autorità competenti e al pubblico.

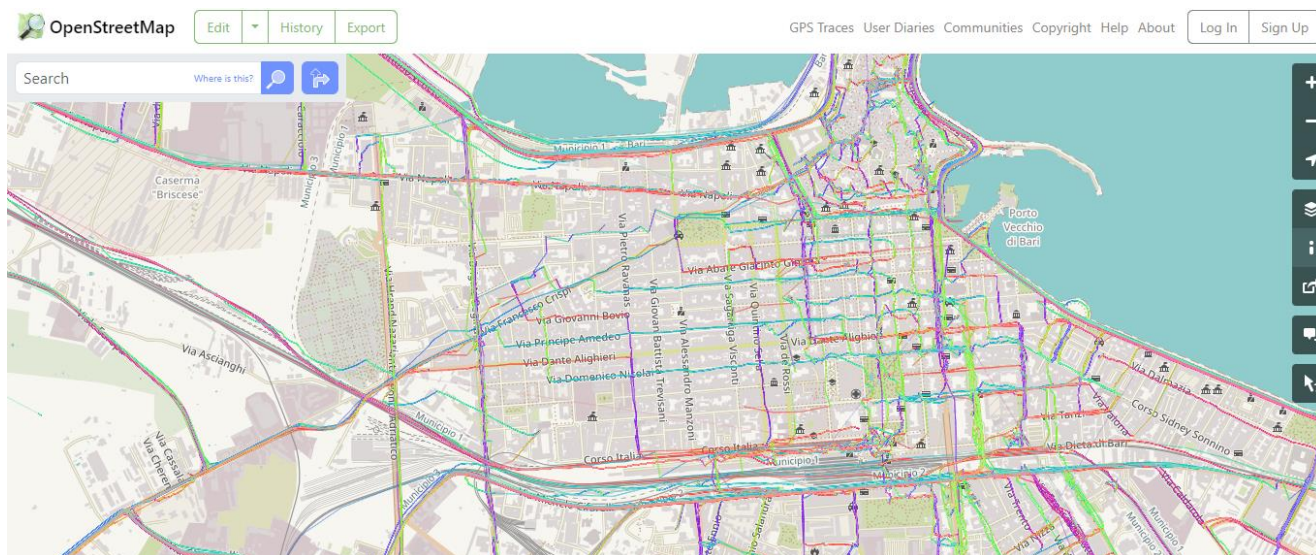
Introduzione:

Negli ultimi decenni, la crescente industrializzazione e l'urbanizzazione hanno portato a un aumento significativo dell'inquinamento atmosferico in molte aree del mondo. L'esposizione a inquinanti come PM2.5, PM10 , NO2 , CO ha gravi implicazioni sulla salute umana e sull'ambiente. Pertanto, è diventato sempre più cruciale implementare sistemi di monitoraggio ambientale avanzati per valutare la qualità dell'aria in modo accurato e fornire avvisi tempestivi per proteggere la salute pubblica. In questo contesto, questo progetto propone lo sviluppo di un Sistema di Monitoraggio Ambientale e Controllo della Qualità dell'Aria basato sull'Ingegneria della Conoscenza, che sfrutta dati in tempo reale da sensori ambientali, modelli atmosferici e analisi storiche per valutare e gestire l'inquinamento atmosferico in un'area specifica.

Ontologie

OpenStreetMap (OSM) è un progetto open source che permette agli utenti di creare e condividere dati geografici, come informazioni su strade, edifici e punti di interesse. I dati di OSM sono disponibili in vari formati, incluso XML, un formato di testo ampiamente utilizzato per la trasmissione e rappresentazione dei dati su Internet.

Nel contesto di un Sistema di Monitoraggio Ambientale e Controllo della Qualità dell'Aria, i dati di OSM in formato XML possono essere particolarmente utili. Questi dati possono essere convertiti in Prolog, un linguaggio di programmazione logica utilizzato per rappresentare conoscenza e interrogare basi di dati. La conversione dei dati XML di OSM in un formato compatibile con Prolog consente di creare una serie di clausole e classi che descrivono le relazioni tra gli elementi della mappa, facilitando così l'integrazione con sistemi di monitoraggio ambientale.



Un file XML di OpenStreetMap si compone di diversi elementi che descrivono gli oggetti sulla mappa, tra cui:

- `<node>`: rappresenta un punto geografico sulla mappa, definito da attributi come latitudine e longitudine.

Utilizzo Sensori Ambientali: Ogni sensore di qualità dell'aria può essere rappresentato come un `<node>`, con latitudine e longitudine che indicano la sua posizione esatta.

Punti di Riferimento: Puoi mappare punti chiave come parchi, scuole o industrie, che possono influenzare i livelli di qualità dell'aria.

- `<way>`: è una sequenza di nodi che rappresenta strade, sentieri, fiumi o altre linee.

Utilizzo Percorsi di Trasporto: Mappare strade o percorsi di trasporto può aiutare a comprendere le fonti di inquinamento legate al traffico.

Zone di Monitoraggio: Le strade possono essere utilizzate per delineare aree specifiche di monitoraggio e analizzare l'impatto delle emissioni stradali sulla qualità dell'aria.

- <relation>: definisce una relazione tra vari elementi (nodi, vie, altre relazioni).

Utilizzo Zone di Influenza: Può essere utilizzato per definire relazioni tra sensori e zone specifiche, come la relazione tra una strada trafficata e la qualità dell'aria nelle aree circostanti.

Reti di Monitoraggio: Può rappresentare reti complesse di sensori o la correlazione tra diverse fonti di dati (es. sensori e stazioni meteorologiche).

- <tag>: contiene metadati, come il nome di una strada, il tipo di edificio o altro.

Utilizzo Attributi dei Sensori: Puoi usare i <tag> per aggiungere informazioni come il tipo di sensore, il tipo di inquinante misurato (es. PM2.5, NO2), la data di installazione, ecc.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <osm version="0.6" generator="CGImap 0.9.3 (4148004 spike-08.openstreetmap.org)" copyright="OpenStreetMap and contributors" attr
3 <bounds minlat="41.1148000" minlon="16.8481000" maxlat="41.1264000" maxlon="16.8563000"/>
4 <node id="68528585" visible="true" version="47" changeset="152112135" timestamp="2024-06-01T09:23:04Z" user="Buraddo" uid="2287
5 <tag k="capital" v="A" />
6 <tag k="gloss_id" v="6166" />
7 <tag k="name" v="Bari" />
8 <tag k="name:ar" v="بَارِي" />
9 <tag k="name:be" v="Bapu" />
10 <tag k="name:el" v="Μηδία" />
11 <tag k="name:en" v="Bari" />
12 <tag k="name:fa" v="باری" />
13 <tag k="name:ja" v="バリ" />
14 <tag k="name:ks" v="باری" />
15 <tag k="name:mk" v="Бапи" />
16 <tag k="name:ro" v="Bari" />
17 <tag k="name:ru" v="Бапи" />
18 <tag k="name:sp" v="Bapu" />
19 <tag k="name:uk" v="Бапи" />
20 <tag k="name:un" v="باری" />
21 <tag k="name:zh" v="巴里" />
22 <tag k="name:zh-Hant" v="巴里" />
23 <tag k="place" v="city" />
24 <tag k="population" v="326344" />
25 <tag k="rank" v="20" />
26 <tag k="source" v="geodati.gfoss.it" />
27 <tag k="wikidata" v="Q3519" />
28 </node>
29 <node id="139164298" visible="true" version="8" changeset="39173761" timestamp="2016-05-08T01:38:09Z" user="mcheckimport" uid="
30 <node id="139164301" visible="true" version="7" changeset="32978720" timestamp="2015-07-30T12:20:50Z" user="gfigliuolo97" uid="
31 <node id="139164303" visible="true" version="6" changeset="8253723" timestamp="2011-05-26T14:45:42Z" user="_mrp_" uid="218322"
32 <node id="139164304" visible="true" version="5" changeset="7089702" timestamp="2011-01-10T13:20:00Z" user="mcheckimport" uid="218322"
33 </node>
```

Classificazione delle Aree: I tag possono aiutare a classificare aree in base all'uso del suolo, come residenziale, industriale o commerciale, per analizzare l'impatto sull'ambiente.

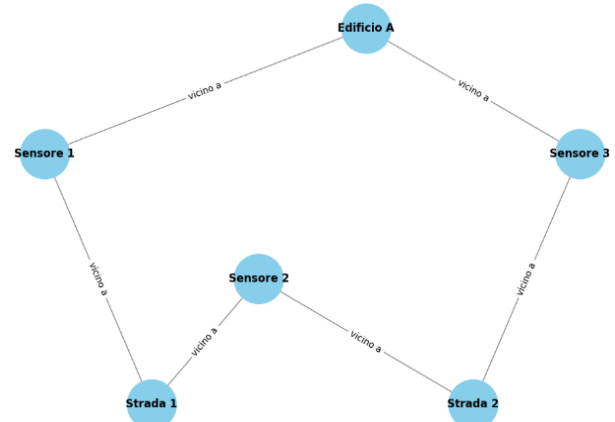
Questi dati sono organizzati in una struttura a grafo, con

ogni elemento identificato da un ID univoco e connesso ad altri elementi tramite relazioni. Questo schema facilita l'accesso e l'estrazione di informazioni utili per il monitoraggio ambientale e il controllo della qualità dell'aria. Ad esempio, è possibile utilizzare i dati di OSM per localizzare sensori di qualità dell'aria rispetto a strade e edifici, ottimizzando la raccolta e l'analisi dei dati ambientali.

I nodi blu rappresentano sensori di qualità dell'aria e altri elementi come strade e edifici.

Gli archi (le linee che collegano i nodi) rappresentano le relazioni di vicinanza tra i diversi elementi.

Le etichette sugli archi indicano la relazione "vicino a" tra un sensore e altri elementi della mappa, come strade o edifici.



Knowledge Base (KB)

Una Knowledge Base (KB) è un insieme di conoscenze organizzate in modo da poter essere utilizzate da un programma o sistema per rispondere a domande o prendere decisioni. una KB memorizza informazioni sui vari inquinanti, le misurazioni rilevate dai sensori, le condizioni ambientali e le regole per l'analisi e il controllo della qualità dell'aria, è importante definire chiaramente le classi e le loro sottoclassi, insieme alla descrizione di ciascuna di esse. Di seguito trovi la descrizione dettagliata delle classi e delle sottoclassi usate nella KB.

Fatti e regole della KB

Per strutturare una base di conoscenza (KB) in Prolog per un "Sistema di Monitoraggio Ambientale e Controllo della Qualità dell'Aria basato sull'Intelligenza Artificiale", è utile organizzare le informazioni in classi e sottoclassi. Questo approccio aiuta a modellare il dominio in modo più organizzato e semantico, facilitando la creazione di fatti e regole.

I fatti rappresentano dati concreti e informazioni specifiche che il sistema conosce. Sono affermazioni che descrivono lo stato del sistema, come le misurazioni dei sensori, le soglie di inquinanti e le condizioni ambientali in determinati luoghi e tempi.

- **Classe Sensore** : Rappresenta un dispositivo che raccoglie dati ambientali, come i livelli di inquinamento atmosferico,Attributi:

id_sensore: Identificativo univoco del sensore.

tipo: Tipo di sensore (es. PM10, NO2, CO2).

posizione: Coordinate geografiche del sensore (latitudine e longitudine).

stato: Stato operativo del sensore (attivo, inattivo).

Sottoclasse: Sensore Inquinamento,Attributi:

soglia_inquinante: Valore soglia per l'inquinante monitorato.

frequenza_campionamento: Frequenza con cui vengono raccolti i dati.

```
/* Classe sensore
*
* Contiene i seguenti attributi:
* - id: Identificativo del sensore
* - tipo: Tipo di sensore (es. NO2, PM2.5, CO2)
* - posizione: Posizione geografica del sensore (es. latitudine, longitudine)
* - stato: Stato del sensore (attivo, inattivo)
*/

/* Classe sensore_inquinamento sottoclasse di sensore */
prop(sensore_inquinamento, subClassOf, sensore).

/* Classe sensore_meteo sottoclasse di sensore */
prop(sensore_meteo, subClassOf, sensore).
```

- **Classe Misurazione**: Rappresenta una registrazione dei dati raccolti da un sensore in un dato.,Attributi:

id_misurazione: Identificativo univoco della misurazione.

inquinante: Tipo di inquinante misurato.

valore: Valore misurato dell'inquinante.

data: Data della misurazione.

sensore: Riferimento al sensore che ha effettuato la misurazione.

Sottoclasse: Misurazione Critica,Attributi:

livello_soglia_superato: Booleano che indica se il valore misurato supera la soglia di sicurezza.

```
/* Classe misurazione
*
* Contiene i seguenti attributi:
* - id: Identificativo della misurazione
* - inquinante: Tipo di inquinante misurato
* - valore: Valore della misurazione
* - data: Data della misurazione
* - sensore: Sensore che ha effettuato la misurazione
*/

/* Classe misurazione_critica sottoclasse di misurazione */
prop(misurazione_critica, subClassOf, misurazione).

/* Classe misurazione_normale sottoclasse di misurazione */
prop(misurazione_normale, subClassOf, misurazione).
```


- **Classe Condizione Ambientale:** Rappresenta le condizioni ambientali rilevate in un determinato luogo e tempo,Attributi:

```
/* Classe condizione_ambientale
*
* Contiene i seguenti attributi:
* - temperatura: Temperatura dell'aria (in gradi Celsius)
* - umidita: Percentuale di umidità relativa
* - velocita_vento: Velocità del vento (in km/h)
* - direzione_vento: Direzione del vento
* - pressione: Pressione atmosferica (in hPa)
* - data: Data della rilevazione
* - luogo: Luogo della rilevazione
*/

/* Classe condizione_critica sottoclasse di condizione_ambientale */
prop(condizione_critica, subClassOf, condizione_ambientale).

/* Classe condizione_normale sottoclasse di condizione_ambientale */
prop(condizione_normale, subClassOf, condizione_ambientale).
```

temperatura: Temperatura rilevata.

umidita: Percentuale di umidità.

velocita_vento: Velocità del vento.

direzione_vento: Direzione da cui proviene il vento.

pressione: Pressione atmosferica.

luogo: Nome del luogo in cui sono state rilevate le condizioni.

data: Data della registrazione.

Sottoclasse: Condizione Critica,Attributi:

allerta_vento_forte: Booleano che indica se è stato rilevato vento forte.

allerta_inquinamento: Booleano che segnala condizioni che possono peggiorare la qualità dell'aria.

Classe allerta: rappresenta gli avvisi emessi dal sistema di monitoraggio ambientale per segnalare situazioni di potenziale pericolo o condizioni che richiedono attenzione, Attributi:

livello: Indica il livello di gravità dell'allerta, che può essere ad esempio "basso", "medio", o "alto". Questo permette di classificare le allerte in base alla loro urgenza e pericolosità.

data: Registra la data in cui l'allerta è stata emessa, permettendo di tenere traccia temporale degli eventi.

sensore: (opzionale) Identifica il sensore che ha generato l'allerta, utile per localizzare rapidamente la fonte del problema.

```
/* Classe allerta
*
* Contiene i seguenti attributi:
* - livello: Livello dell'allerta (basso, medio, alto)
* - descrizione: Descrizione dell'allerta
* - data: Data dell'emissione dell'allerta
*/

/* Classe allerta_inquinamento sottoclasse di allerta */
prop(allerta_inquinamento, subClassOf, allerta).

/* Classe allerta_meteo sottoclasse di allerta */
prop(allerta_meteo, subClassOf, allerta).
```

Query Knowledge Base (KB)

Le query alla Knowledge Base (KB) sono essenziali per estrarre informazioni specifiche e prendere decisioni informate riguardo alla qualità dell'aria. In questo contesto, le query permettono di identificare livelli pericolosi di inquinanti,ottenere informazioni sulle condizioni ambientali in un luogo e in un momento specifico,Recuperare misurazioni effettuate da specifici sensori.

Le query sono costruite utilizzando la sintassi di Prolog, che è un linguaggio di programmazione logica. Ogni query segue un formato predicato-argomento, dove:

Predicati: Rappresentano relazioni o funzioni.

Argomenti: Sono variabili o valori concreti.

Le regole (supera_livello_soglia, misurazioni_pericolose, condizioni_in_luogo_tempo) utilizzano fatti esistenti e condizioni logiche per derivare nuove informazioni.

Di seguito, spiego come sono state create alcune query, il motivo per cui sono state sviluppate e cosa fanno esattamente.

1. Il predicato `calcola_aqi` calcola l'Indice di Qualità dell'Aria (AQI) basato su diversi parametri ambientali

Temp: La temperatura ambientale. (Nota: In

questa definizione, la temperatura non viene effettivamente utilizzata nel calcolo dell'AQI, ma è inclusa nei parametri per coerenza con il contesto delle regole ambientali).

Umidita: L'umidità ambientale. (Simile alla temperatura, non viene utilizzata nel calcolo attuale dell'AQI).

CO2: Il livello di CO2 nell'aria.

PM25: Il livello di PM2.5 (particolato fine con diametro $\leq 2.5 \mu\text{m}$).

PM10: Il livello di PM10 (particolato con diametro $\leq 10 \mu\text{m}$).

AQI: L'Indice di Qualità dell'Aria calcolato.

La formula utilizzata per calcolare l'AQI è: $\text{AQI} \text{ is } (\text{PM25} + \text{PM10} + \text{CO2}) / 3$.

Questa formula semplicemente calcola la media aritmetica dei valori di PM2.5, PM10 e CO2 per ottenere l'AQI.

2. Il predicato `qualita_aria_localita` calcola e restituisce la qualità dell'aria di una località specificata

Localita: La località di cui si desidera conoscere la qualità dell'aria.

Qualita: La qualità dell'aria della località (restituito come risultato, può essere 'pericolosa' o 'sicura').

Recupero dei Dati Ambientali:

`prop(Localita, temperatura, Temp):`

Recupera la temperatura.

`prop(Localita, umidita, Umidita):` Recupera l'umidità.

`prop(Localita, co2, CO2):` Recupera il livello di CO2.

`prop(Localita, pm25, PM25):` Recupera il livello di PM2.5.

`prop(Localita, pm10, PM10):` Recupera il livello di PM10.

Calcolo dell'AQI (Indice di Qualità dell'Aria): Chiama il predicato `calcola_aqi` per calcolare l'AQI basato sui dati ambientali recuperati.

Verifica della Qualità dell'Aria: Chiama il predicato `verifica_allerta_qualita` per determinare se l'AQI è pericoloso o meno, se `Allerta` è true, imposta `Qualita` come 'pericolosa', altrimenti come 'sicura'.

```
/**
 * Calcola l'indice della qualità dell'aria basato sui parametri ambientali
 */
@param Temp: temperatura
@param Umidita: umidità
@param CO2: livello di CO2
@param PM25: livello di PM2.5
@param PM10: livello di PM10
@param AQI: indice della qualità dell'aria (viene restituito il risultato)
*/
calcola_aqi(Temp, Umidita, CO2, PM25, PM10, AQI) :-
    AQI is (PM25 + PM10 + CO2) / 3.
```

```
/**
 * Restituisce la qualità dell'aria di una località specificata
 */
@param Localita: località di cui si vuole conoscere la qualità dell'aria
@param Qualita: qualità dell'aria della località (viene restituito il risultato)
*/
qualita_aria_localita(Localita, Qualita) :-
    prop(Localita, temperatura, Temp),
    prop(Localita, umidita, Umidita),
    prop(Localita, co2, CO2),
    prop(Localita, pm25, PM25),
    prop(Localita, pm10, PM10),
    calcola_aqi(Temp, Umidita, CO2, PM25, PM10, AQI),
    verifica_allerta_qualita(AQI, Allerta),
    (Allerta == true -> Qualita = 'pericolosa' ; Qualita = 'sicura').
```

3. Il predicato `lat_lon` restituisce la latitudine e la longitudine di una località specificata

Localita: La località di cui si vogliono conoscere le coordinate.

Latitudine: La latitudine della località.

Longitudine: La longitudine della località.

Il predicato utilizza la funzione `prop` per accedere alle proprietà di `Localita`.

`prop(Localita, latitudine, Latitudine)` trova la latitudine della località e la unisce alla variabile `Latitudine`.

`prop(Localita, longitudine, Longitudine)` trova la longitudine della località e la unisce alla variabile `Longitudine`.

Il risultato è che, per una data località, vengono restituite le sue coordinate geografiche (latitudine e longitudine).

Aggiornamento Knowledge Base (KB)

L'aggiornamento della Knowledge Base (KB) è un processo essenziale per mantenere i dati sempre attuali e pertinenti. Consiste nell'aggiungere nuovi fatti, modificare quelli esistenti o eliminare fatti obsoleti. Per gestire la Knowledge Base, è stata creata una classe Python chiamata `KnowledgeBase` che utilizza la libreria `pyswip` per eseguire le query di interrogazione sulla KB viste precedentemente.

```
/**
 * Restituisce la latitudine e longitudine di una località specificata
 *
 * @param Localita: località di cui si vogliono conoscere le coordinate
 * @param Latitudine: latitudine della località
 * @param Longitudine: longitudine della località
 */
lat_lon(Localita, Latitudine, Longitudine) :-
    prop(Localita, latitudine, Latitudine),
    prop(Localita, longitudine, Longitudine).
```

```
def assegna_stazione_monitoraggio(self, stazione, configurazione):
    """
    Metodo assegna_stazione_monitoraggio
    -----
    Dati di input
    -----
    stazione: stazione di monitoraggio da assegnare
    configurazione: configurazione della stazione
    """
    self.dict_stazioni[stazione] = configurazione

    # Assegna la nuova stazione anche al file Prolog
    for zona, config in configurazione.items():
        query = f"assertz(prop({stazione}, {zona}, {zona}))"
        self.prolog.query(query)
    for key, value in config.items():
        query = f"assertz(prop({stazione}, {zona}, {key}, {value}))"
        self.prolog.query(query)
    print(f"Stazione {stazione} assegnata alla KB.")

def rimuovi_stazione_monitoraggio(self, stazione):
    """
    Metodo rimuovi_stazione_monitoraggio
    -----
    Dati di input
    -----
    stazione: stazione di monitoraggio da rimuovere
    """
    if stazione in self.dict_stazioni:
        del self.dict_stazioni[stazione]

        # Rimuovi la stazione anche dal file Prolog
        query = f"retractall(prop({stazione}, _, _))"
        self.prolog.query(query)
        print(f"Stazione {stazione} rimossa dalla KB.")
    else:
        print(f"Stazione {stazione} non trovata nella KB.")

def aggiorna_configurazione_monitoraggio(self, stazione, configurazione):
    """
    Metodo aggiorna_configurazione_monitoraggio
    -----
    Dati di input
    -----
    stazione: stazione di monitoraggio da aggiornare
    configurazione: nuova configurazione della stazione
    """
    if stazione not in self.dict_stazioni:
        print(f"Stazione {stazione} non trovata nella KB.")
        return

    self.dict_stazioni[stazione] = configurazione

    # Aggiorna la configurazione nel file Prolog
    for zona, config in configurazione.items():
        query = f"assertz(prop({stazione}, {zona}, {zona}))"
        self.prolog.query(query)
    for key, value in config.items():
        query = f"assertz(prop({stazione}, {zona}, {key}, {value}))"
        self.prolog.query(query)
    print(f"Configurazione di monitoraggio per {stazione} aggiornata.")

def rimuovi_stazione_monitoraggio(self, stazione):
    """
    Metodo rimuovi_stazione_monitoraggio
    -----
    Dati di input
    -----
    stazione: stazione di monitoraggio da aggiornare
    dati_ambientali: dizionario contenente i nuovi dati ambientali
    """
    if stazione not in self.dict_stazioni:
        print(f"Stazione {stazione} non trovata nella KB.")
        return

    # Aggiorna i dati ambientali nel dizionario della KB
    self.dict_stazioni[stazione].update(dati_ambientali)

    # Aggiorna i dati nel file Prolog (se necessario)
    for key, value in dati_ambientali.items():
        query = f"assertz(prop({stazione}, {key}, {value}))"
        self.prolog.query(query)
    print(f"Dati ambientali per {stazione} aggiornati.")
```

la classe fornisce funzioni come **aggiorna_dati_ambientali**, **aggiorna_configurazione_monitoraggio**, **rimuovi_stazione_monitoraggio**, e **assegna_stazione_monitoraggio** per gestire i dati del controllo la qualità del ambiente nella KB di uno specifico zona.

aggiorna_dati_ambientali: Aggiorna i dati ambientali di una stazione di monitoraggio con i nuovi valori forniti. Modifica il dizionario interno della stazione e, se necessario, aggiorna anche il database Prolog.

aggiorna_configurazione_monitoraggio: Modifica la configurazione di monitoraggio di una stazione con una nuova configurazione specificata. Aggiorna il dizionario interno e, se necessario, il database Prolog con i nuovi dettagli della configurazione.

rimuovi_stazione_monitoraggio: Rimuove una stazione di monitoraggio dal dizionario interno e dal database Prolog, se presente. Elimina tutti i dati e le configurazioni associati alla stazione.

assegna_stazione_monitoraggio: Aggiunge una nuova stazione di monitoraggio al dizionario interno con la configurazione specificata. Inserisce anche i dettagli della stazione nel database Prolog per la gestione delle informazioni.

Apprendimento Supervisionato

Il sistema di monitoraggio ambientale raccoglie dati in tempo reale da sensori distribuiti nelle aree urbane. Questi sensori misurano vari inquinanti atmosferici come PM2.5, PM10, NO2 e CO, oltre a variabili meteorologiche come temperatura, umidità e velocità del vento. I dati vengono raccolti ogni giorno e memorizzati in un database centralizzato.

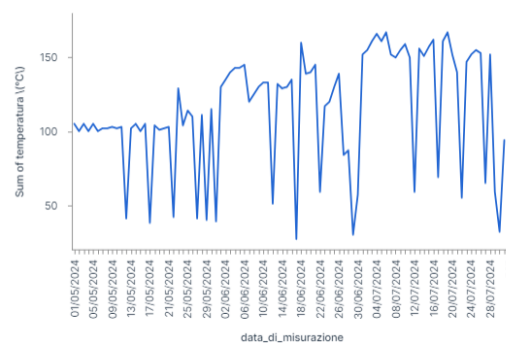
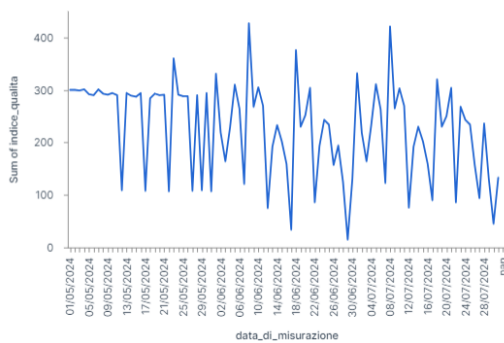
Il cuore del sistema è un modello di intelligenza artificiale basato su apprendimento supervisionato, progettato per predire i livelli di inquinamento atmosferico (AQI) per specifiche aree urbane. Il modello viene addestrato utilizzando un dataset storico che comprende:

- Livelli di inquinamento: Dati raccolti sui principali inquinanti atmosferici.
- Tipo di area: Residenziale, commerciale, industriale.
- Giorno della settimana e orario: Le condizioni di inquinamento possono variare significativamente a seconda del giorno e dell'orario.
- Condizioni meteorologiche: Temperature, umidità, velocità del vento, ecc.

Il modello predice l'AQI utilizzando la seguente formula:

$$AQI = \left(\frac{PM2.5 + PM10 + CO2}{3} \right) \times 100$$

Questa formula normalizza i livelli di inquinanti principali, restituendo un punteggio che rappresenta la qualità dell'aria per ciascuna area.



Calcolo delle Misure di Intervento

Una volta che l'AQI è stato predetto, il sistema calcola le misure di intervento necessarie per mantenere la qualità dell'aria entro limiti accettabili. La formula utilizzata per determinare il tempo di intervento è:

$$\text{Tempo di Intervento} = \max(30, (\text{AQI} \times 10) \times 2)$$

Dove:

- AQI: Indice di inquinamento previsto.
- 30: Tempo minimo di intervento in minuti per garantire misure efficaci.
- 10: Costante usata per la normalizzazione.
- 2: Fattore che amplifica l'intensità dell'intervento.

Tempo di Allerta

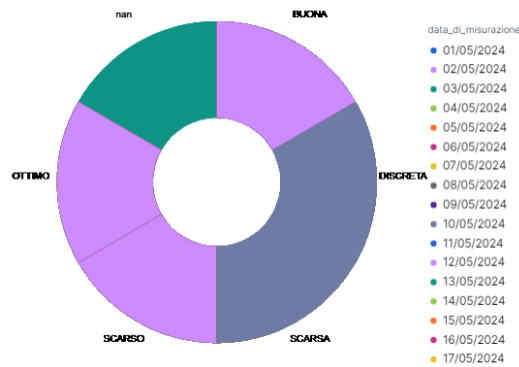
Il sistema aggiorna le previsioni e le azioni correttive ogni 10 minuti per garantire una risposta tempestiva. Il Tempo di Allerta è quindi di 10 minuti.

Il Tempo di Azione Totale per una data area è calcolato come:

$$\text{Tempo di Azione Totale} = \sum (T_{\text{intervento}(i)} + T_{\text{allerta}(i)})$$

Dove:

- $T_{\text{intervento}(i)}$: Tempo impiegato per intervenire nell'area i , influenzato dall'AQI previsto.
- $T_{\text{allerta}(i)}$: Fisso a 10 minuti, è il tempo durante il quale il sistema rivaluta la situazione e potenzialmente aggiorna le misure.



Visualizzazione

Il sistema include strumenti di visualizzazione per monitorare in tempo reale la qualità dell'aria e l'efficacia delle misure adottate. I report generate includono:

- Mappe di Inquinamento: Visualizzazione dei livelli di inquinamento nelle diverse aree urbane.
- Grafici delle Tendenze: Andamento storico e previsioni future della qualità dell'aria.
- Rapporti di Efficacia: Valutazione dell'impatto delle azioni di controllo e suggerimenti per miglioramenti.

Scelta del modello

Sommario

Per ottimizzare il monitoraggio ambientale e il controllo della qualità dell'aria, sono stati esaminati tre modelli di intelligenza artificiale per valutare le loro prestazioni in termini di accuratezza e tempi di risposta. Il nostro compito è una regressione, finalizzata a prevedere i livelli di inquinamento in relazione a variabili come la data.

- Il primo modello analizzato è l'Albero di Decisione. Questo modello di apprendimento supervisionato viene utilizzato per compiti di regressione e classificazione, sfruttando una struttura ad albero per effettuare previsioni. Gli alberi decisionali creano un modello predittivo basato su regole decisionali che segmentano i dati in base a variabili continue. Nel nostro caso, l'Albero di Decisione è stato impiegato per stimare i livelli di qualità dell'aria, utilizzando dati temporali e ambientali per fare previsioni sulla variabile target continua. Nel contesto del monitoraggio della qualità dell'aria:
 - Stima dei Livelli di Qualità dell'Aria: L'Albero di Decisione è utilizzato per prevedere i livelli di inquinamento atmosferico. Utilizzando dati ambientali (come temperatura, umidità, e livelli di inquinanti) e temporali (come il giorno della settimana), il modello stima i valori di qualità dell'aria.
 - Analisi delle Variabili Ambientali: L'albero decisionale aiuta a identificare come diverse variabili ambientali influenzano la qualità dell'aria. Ad esempio,

può determinare come l'aumento della temperatura o della umidità influisce sui livelli di PM2.5 o PM10.

- Decisioni Basate su Regole: Le regole decisionali generate dall'albero permettono di comprendere quali combinazioni di variabili ambientali sono più fortemente correlate ai cambiamenti nella qualità dell'aria, facilitando interventi mirati e tempestivi.
- Il secondo modello considerato è il K-means, che è un algoritmo di clustering non supervisionato. È un algoritmo di clustering che raggruppa i dati in cluster basati su similarità. K-means identifica gruppi di dati con caratteristiche simili, permettendo di scoprire pattern di inquinamento e tendenze nelle aree urbane. Il K-means. Sebbene non sia tradizionalmente usato per la regressione, può essere utile per analizzare i dati di inquinamento atmosferico

Nel contesto del monitoraggio della qualità dell'aria, K-means utilizza per:

- Identificare gruppi di aree con livelli di inquinamento simili.
 - Analizzare le tendenze di inquinamento nelle diverse zone.
 - Fornire una visione chiara delle aree con caratteristiche di inquinamento omogenee, facilitando l'analisi e l'intervento mirato.
-
- Il Terzo La regressione logistica è un modello di classificazione che prevede la probabilità di appartenenza a una certa classe. Anche se è più comunemente usata per compiti di classificazione binaria, può essere adattata alla regressione. Questo modello stima le probabilità di determinati livelli di inquinamento e contribuisce alla previsione della qualità dell'aria. Nel contesto del monitoraggio della qualità dell'aria, la regressione logistica può essere utilizzata per:
 - Stimare la probabilità che i livelli di inquinamento superino una soglia critica.
 - Fornire previsioni sui livelli di inquinamento basati su variabili ambientali e temporali. Classificare le aree in base ai loro livelli di inquinamento, aiutando a pianificare le azioni correttive.

Strumenti utilizzati

Per l'implementazione dei modelli nel nostro sistema di monitoraggio ambientale e controllo della qualità dell'aria, abbiamo utilizzato la libreria Scikit-learn, scegliendo le seguenti classi per ciascun tipo di modello:

Albero di Decisione: Per l'implementazione del modello relativo agli alberi di decisione, è stata utilizzata la classe `DecisionTreeRegressor` della libreria Scikit-learn. Questo modello di regressione costruisce un albero decisionale per effettuare previsioni sui livelli di qualità dell'aria, sfruttando dati temporali e ambientali per stimare la variabile target continua.

Regressione Logistica: Per l'implementazione della regressione logistica, è stata utilizzata la classe `LogisticRegression` della libreria `Scikit-learn`. Sebbene la regressione logistica sia comunemente impiegata per problemi di classificazione, può essere adattata per problemi di regressione attraverso tecniche come la regressione logistica multivariata. Questo modello è stato impiegato per stimare probabilità di livelli di qualità dell'aria basati su variabili ambientali e temporali.

K-means: Per l'implementazione del modello di clustering K-means, è stata utilizzata la classe `KMeans` della libreria `Scikit-learn`. Questo algoritmo di clustering è stato utilizzato per identificare gruppi o cluster nei dati di qualità dell'aria, che possono poi essere analizzati per comprendere i pattern di inquinamento e supportare le previsioni indirette

Decisioni di Progetto

Di seguito sono riportate le scelte degli iperparametri per i modelli utilizzati nel nostro sistema di monitoraggio ambientale e controllo della qualità dell'aria, impostati "manualmente" dopo una serie di test e valutazioni.

Albero di Decisione (Decision Tree Classifier)

L'Albero di Decisione è stato scelto per la sua capacità di visualizzare chiaramente il processo decisionale e le regole che il modello utilizza per classificare la qualità dell'aria. Abbiamo utilizzato l'albero di decisione per determinare le condizioni specifiche (combinazioni di temperatura, umidità, vento, livello di inquinante) che portano a ciascuna classe di qualità dell'aria.

Regressione Logistica (Logistic Regression)

La Regressione Logistica è stata scelta per prevedere la qualità dell'aria (classi: "Buona", "Moderata", "Scadente", "Molto Scadente") sulla base delle variabili misurate. È particolarmente adatta per problemi di classificazione multiclasse come il nostro. Il modello è stato utilizzato per determinare quali variabili ambientali (temperatura, umidità, vento, livello di inquinante) influenzano maggiormente la probabilità che l'aria sia di una certa qualità.

K-Means Clustering

Il K-Means Clustering è stato utilizzato per esplorare i dati in modo non supervisionato, per identificare gruppi (cluster) naturali basati su somiglianze nelle condizioni ambientali. È stato scelto per trovare schemi nascosti e raggruppare giorni o condizioni che tendono ad avere qualità dell'aria simile.

Valutazione

Per valutare le prestazioni dei modelli di Regressione Logistica, Alberi di Decisione e K-means nel contesto del nostro sistema di monitoraggio ambientale e controllo della qualità dell'aria,

Albero decisione

Parametri applicati:

qui per avere una classe piu

```
best_depth = 10 # Puoi modificare la profondità in base alle tue esigenze
dt_model = DecisionTreeClassifier(max_depth=best_depth, random_state=42)
dt_model.fit(X_scaled, y)
```

complesso e piu completo abbiamo usato una regola basata sui livelli di diversi inquinanti presenti nell'aria, Regola per la Classificazione della Qualità dell'Aria Per classificare la qualità dell'aria in "Buona", "Moderata" o "Scadente", abbiamo creato una tabella che definisce intervalli specifici per ciascun livello di inquinante.

Livello di Inquinante	Buona	Moderata	Scadente
PM10 ($\mu\text{g}/\text{m}^3$)	≤ 20	> 20 e ≤ 50	> 50
NO2 ($\mu\text{g}/\text{m}^3$)	≤ 40	> 40 e ≤ 100	> 100
O3 ($\mu\text{g}/\text{m}^3$)	≤ 80	> 80 e ≤ 180	> 180
CO (mg/m^3)	≤ 4	> 4 e ≤ 10	> 10

Semplicità e Interpretabilità:

L'albero è relativamente profondo (profondità di 10), il che implica un modello complesso con molte regole di decisione. Mentre un albero complesso può catturare meglio la struttura dei dati, potrebbe anche essere più difficile da interpretare rispetto a un albero più semplice.

Decisione Basata su Caratteristiche Chiave:

Le decisioni principali sono prese sulla base del valore dell'inquinante misurato (valore_inquinante_misurato), seguito da variabili come temperatura (temperatura ($^{\circ}\text{C}$)), velocità del vento (Vento (km/h)), e umidità (Umidità (%)). Questo suggerisce che queste variabili sono molto rilevanti per la classificazione della qualità dell'aria.

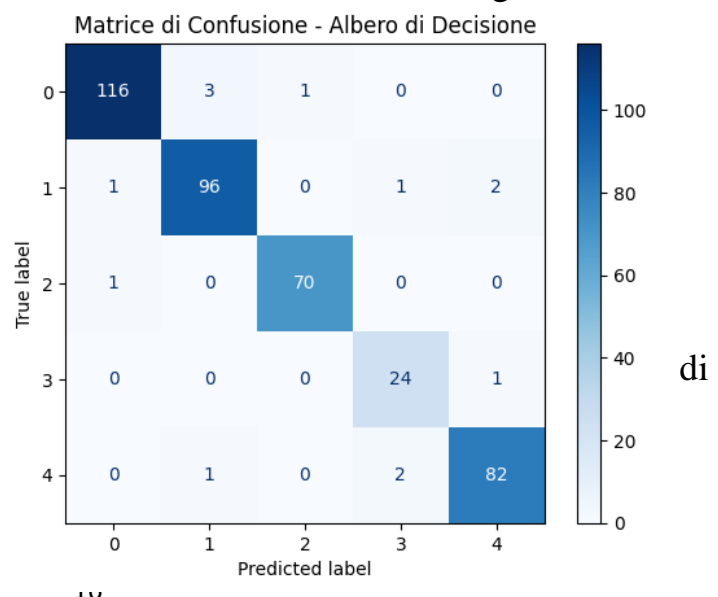
Potenziale Overfitting:

La profondità dell'albero di 10 potrebbe indicare un potenziale overfitting, soprattutto se il dataset è relativamente piccolo. Un albero così profondo potrebbe adattarsi troppo ai dati di addestramento e generalizzare meno bene su nuovi dati.

Utilità della Visualizzazione:

La visualizzazione dell'albero può essere utile per spiegare come il modello prende decisioni e per identificare eventuali nodi decisionali che sembrano irragionevoli o inaspettati. Questa matrice di confusione mostra i risultati del modello ad albero di decisione:

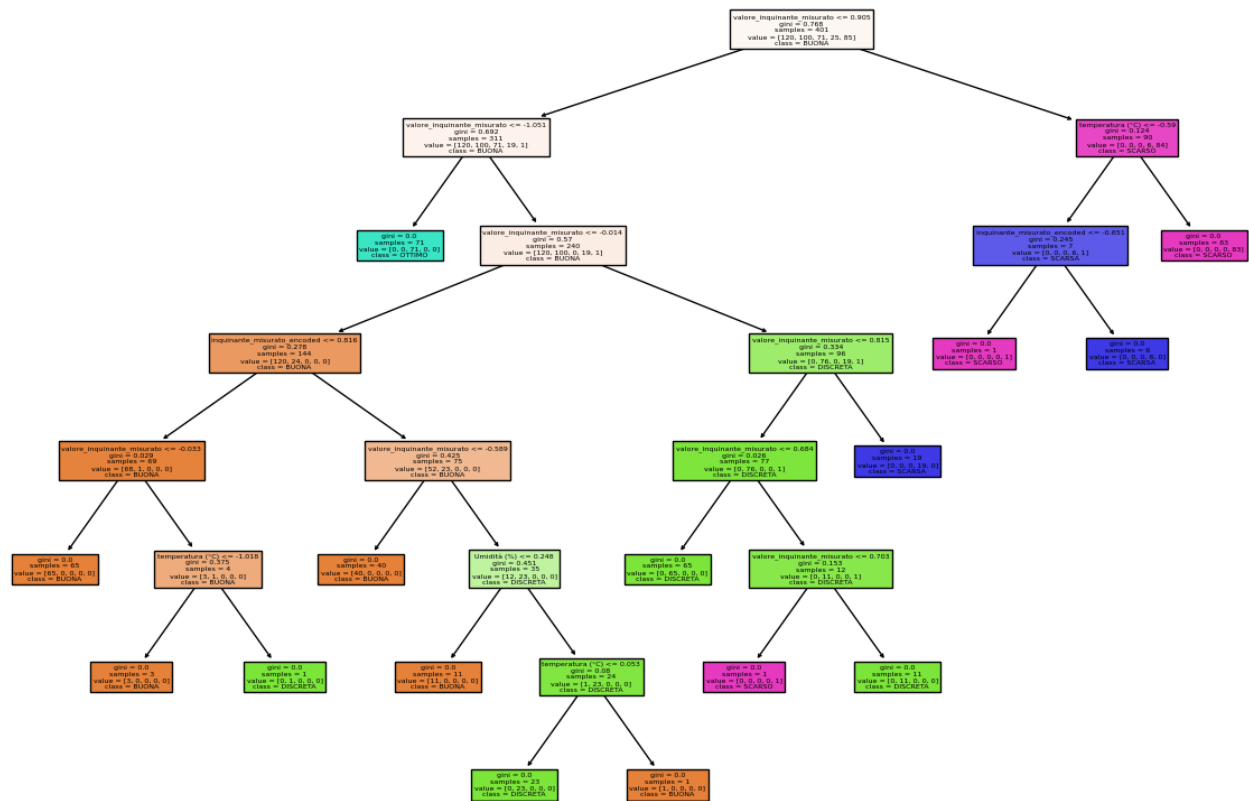
L'albero di decisione ha la più alta accuratezza tra i tre modelli, con molte delle classi classificate correttamente. Ad esempio, 116 campioni di classe 0 e 96 campioni di classe 1 sono correttamente classificati.



Gli errori sono minimi rispetto agli altri modelli, con pochissimi campioni mal classificati tra le diverse classi.

Questo risultato suggerisce che l'albero di decisione è molto adatto a questo dataset e riesce a catturare le relazioni tra le variabili per distinguere accuratamente le classi.

Albero di Decisione con Profondità 10



Regressione Logistica:

Abbiamo utilizzato l'algoritmo di Regressione Logistica per classificare la qualità dell'aria in base ai valori degli inquinanti e ad altre caratteristiche ambientali. La Regressione Logistica è un metodo statistico efficace per problemi di classificazione binaria e multiclasse, in cui si cerca di modellare la probabilità che un'osservazione appartenga a una certa classe.

L'algoritmo di Regressione Logistica è stato addestrato utilizzando una 10-fold cross-validation, che suddivide il dataset in 10 parti, addestrando il modello su 9 di esse e testandolo sulla parte rimanente. Questo processo è ripetuto 10 volte, fornendo una stima robusta delle prestazioni del modello.

Nel progetto, abbiamo utilizzato l'algoritmo di Regressione Logistica per classificare la qualità dell'aria. Per valutare le prestazioni del modello su un problema di classificazione multiclasse (qualità dell'aria: Classe 0: "Buona", Classe 1: "Moderata", Classe 2: "Scadente", Classe 3: "Molto Scadente", Classe 4: "Molto molto Scadente"), abbiamo calcolato le Curve ROC utilizzando una strategia One-vs-Rest (OvR).

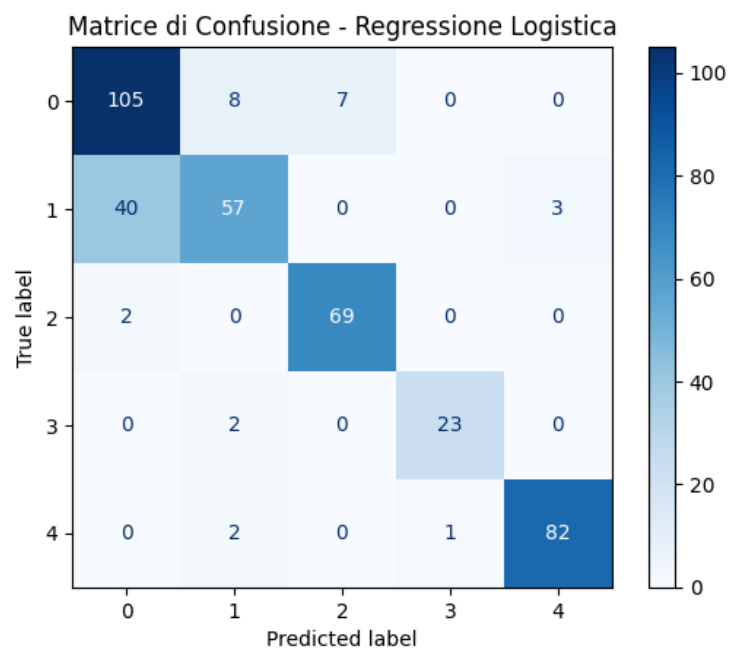
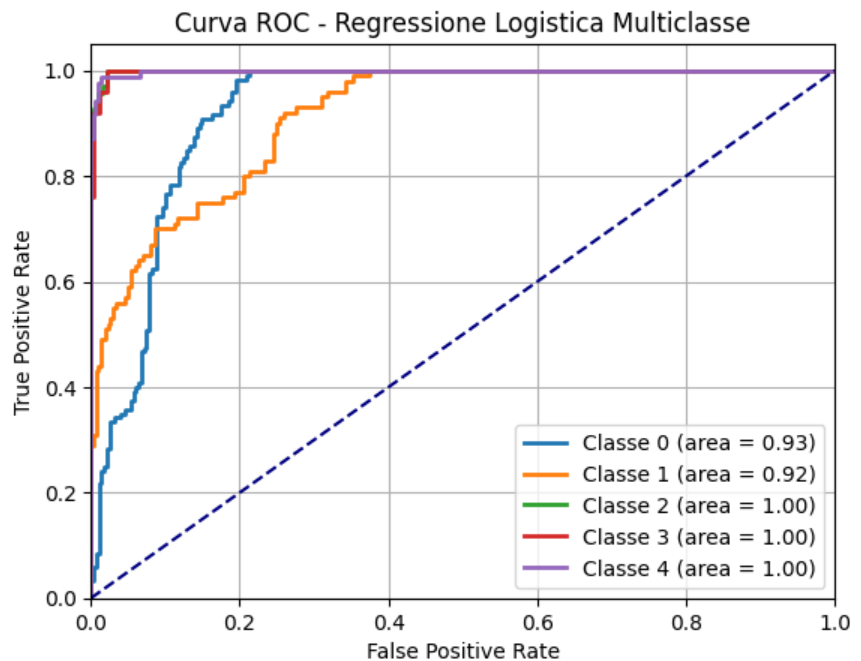
Questo approccio ci ha permesso di tracciare una curva ROC per ciascuna classe, mostrando quanto bene il modello distingue ciascuna classe dalle altre.

Il valore AUC (Area Under the Curve) calcolato per ogni curva ROC indica la capacità discriminante del modello: Un valore di AUC vicino a 1 indica che il modello classifica correttamente quella classe contro tutte le altre.

Un valore vicino a 0,5 suggerisce che il modello non ha capacità discriminante migliore di una classificazione casuale per quella classe.

I risultati hanno mostrato che il modello ha una buona capacità predittiva per ciascuna delle classi, confermando l'efficacia della Regressione Logistica per questo problema di classificazione.

Questa matrice di confusione rappresenta i risultati ottenuti utilizzando la regressione logistica:



La maggior parte delle classi è classificata correttamente, come si vede per la classe 0 con 105 campioni corretti, e la classe 4 con 82 campioni corretti.

Alcuni errori sono evidenti tra le classi 1 e 0, dove 40 campioni della classe 1 sono stati erroneamente classificati come 0.

Nel complesso, la regressione logistica sembra essere un modello efficace per questo dataset, con una buona performance nelle classi con più campioni.

K-Means:

Per il progetto, è stato applicato l'algoritmo di K-Means Clustering per raggruppare i dati relativi alla qualità dell'aria. Il clustering K-Means è un metodo non supervisionato che identifica i gruppi naturali nei dati, basandosi sulle somiglianze tra le caratteristiche.

Determinazione del Numero Ottimale di Cluster

Abbiamo utilizzato il Metodo del Gomito (Elbow Method) per determinare il numero ottimale di cluster. Questo metodo analizza la somma degli errori quadrati all'interno dei cluster (inertia) per diversi valori di "k".

Il numero ottimale di cluster è scelto in corrispondenza del "gomito" della curva, dove l'incremento di miglioramento dell'inertia inizia a ridursi.

Risultati e Valutazione del Clustering

Una volta determinato il numero ottimale di cluster (ad esempio,

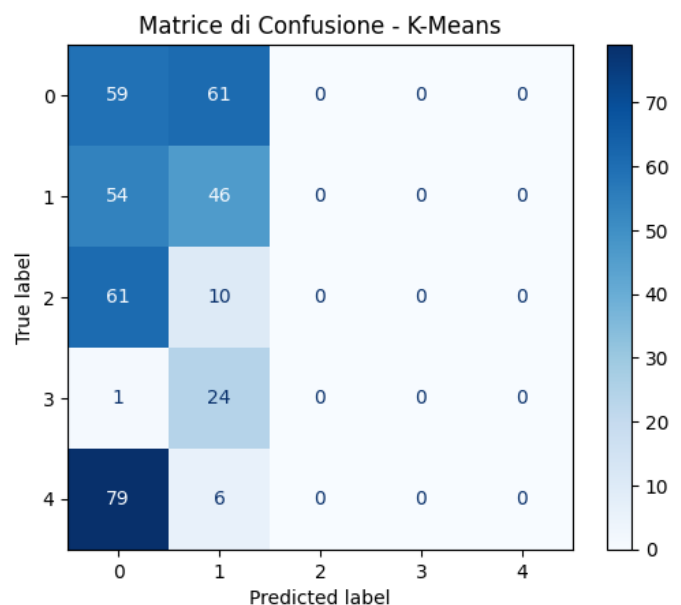
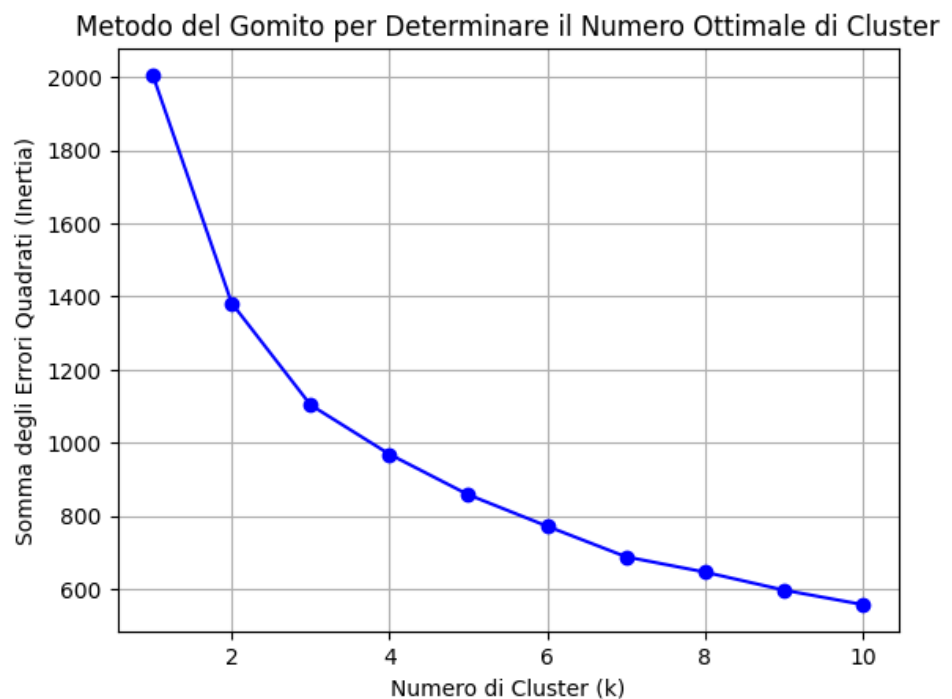
$k=3$), abbiamo applicato l'algoritmo K-Means per raggruppare i dati. Il punteggio silhouette è stato utilizzato per valutare la qualità del clustering. Un punteggio silhouette vicino a 1 indica che i cluster sono ben definiti e separati.

Il grafico risultante mostra i cluster identificati e i centroidi di ciascun cluster, fornendo una rappresentazione visiva della struttura dei dati. Questo risultato

dimostra che l'algoritmo K-Means è in grado di identificare gruppi naturali nei dati sulla qualità dell'aria, facilitando ulteriori analisi e interpretazioni.

Questa matrice di confusione mostra i risultati del modello K-Means. Il clustering K-Means ha difficoltà a distinguere correttamente le diverse classi:

Ci sono molti errori di classificazione, specialmente tra le classi 0 e 1, dove 59 campioni di classe 0 sono classificati



correttamente, ma 61 sono erroneamente assegnati a classe

La distribuzione dei valori suggerisce che il modello non è adatto per un problema di classificazione supervisionata con queste caratteristiche, poiché K-Means è un algoritmo non supervisionato e non ha informazioni su quali siano le etichette corrette durante l'addestramento.

Algoritmo	Cross-Validation Utilizzata	Metodo Utilizzato
Regressione Logistica	Sì	10-fold cross-validation con cross_val_predict
Albero di Decisione	Sì	10-fold cross-validation con cross_val_score
K-Means Clustering	No	Nessuna (clustering non supervisionato)

La cross-validation è stata utilizzata per entrambi gli algoritmi di apprendimento supervisionato (Regressione Logistica e Albero di Decisione) per garantire che le prestazioni del modello siano valutate in modo robusto. Per l'algoritmo K-Means, che è non supervisionato, non abbiamo utilizzato la cross-validation, poiché non esiste una suddivisione naturale dei dati in set di addestramento e test per un algoritmo che raggruppa semplicemente i dati in cluster.

Rete Bayesiana

Sommario

Il sistema di monitoraggio ambientale e controllo della qualità dell'aria si avvale dell'uso delle reti bayesiane per analizzare e prevedere i livelli di inquinamento atmosferico. Le reti bayesiane sono modelli grafici probabilistici che permettono di rappresentare le dipendenze condizionali tra variabili e gestire l'incertezza, offrendo una visione chiara delle relazioni causali tra variabili ambientali e indicatori di qualità dell'aria.

Strumenti Utilizzati

Librerie e Frameworks: pgmpy: Una libreria Python per la modellazione grafica probabilistica che fornisce gli strumenti per creare e gestire reti bayesiane, eseguire inferenze e apprendere dai dati. È utilizzata per costruire e addestrare la rete bayesiana. networkx (nx): È una libreria Python utilizzata per creare, manipolare e studiare la struttura, le dinamiche e le funzioni di grafi complessi. Qui è utilizzata per creare e gestire un grafo che rappresenta la rete bayesiana.

```
from pgmpy.models import BayesianNetwork
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.inference import VariableElimination
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt

# Caricamento del dataset CSV
file_path = 'QUALITA_ARIA_MAGILU2024.csv' # Inserisci il percorso corretto del tuo file CSV
data = pd.read_csv(file_path, encoding='ISO-8859-1', delimiter=',')

# Pre-elaborazione del dataset: sostituzione delle virgole con punti e conversione a numerico
numeric_cols = ['valore_inquinante_misurato', 'temperatura (°C)', 'Vento (km/h)', 'Umidità (%)']
for col in numeric_cols:
    data[col] = pd.to_numeric(data[col].astype(str).str.replace(',', '.'), errors='coerce')

# Codifica delle variabili categoriali
data['classe_qualita_encoded'] = data['classe_qualita'].astype('category').cat.codes

# Definizione della struttura della Rete Bayesiana
# Struttura dove "Inquinamento" influenza altre variabili
model = BayesianNetwork([
    ('classe_qualita_encoded', 'valore_inquinante_misurato'),
    ('classe_qualita_encoded', 'temperatura (°C)'),
    ('classe_qualita_encoded', 'Vento (km/h)'),
    ('classe_qualita_encoded', 'Umidità (%)')])
```

matplotlib.pyplot (plt): È una libreria per la creazione di grafici in Python. Viene usata per visualizzare graficamente il grafo della rete bayesiana.

pgmpy.models.BayesianNetwork: È un modulo della libreria pgmpy che permette di creare e gestire modelli di reti bayesiane. Qui è usato per definire la struttura della rete.

Decisioni di Progetto

Le decisioni di progetto riguardano la selezione degli strumenti e delle librerie per costruire il sistema di monitoraggio ambientale. Queste includono la scelta di librerie come pgmpy, BayesPy e pomegranate per la modellazione e l'inferenza con le reti bayesiane, oltre alla selezione di tool di visualizzazione come Graphviz e Matplotlib. Ogni scelta influisce sulla precisione, efficienza e capacità interpretativa del sistema.

Valutazione

Una rete bayesiana è un modello probabilistico grafico che rappresenta le dipendenze condizionali tra un insieme di variabili attraverso un grafo diretto aciclico. In questo progetto, abbiamo utilizzato una rete bayesiana per modellare le relazioni tra le variabili ambientali (come temperatura, umidità, vento) e la qualità dell'aria (classe_qualita).

La struttura della rete bayesiana è stata definita in modo che la variabile classe_qualita (qualità dell'aria) influenzi altre variabili ambientali, come:

Valore inquinante misurato (valore_inquinante_misurato), Temperatura (temperatura (°C)), Vento (Vento (km/h)), Umidità (Umidità (%)) Questa struttura suggerisce che la qualità dell'aria (classe_qualita), come "Buona", "Moderata", "Scadente", "Molto Scadente", ha un impatto diretto sulle condizioni ambientali misurate. In altre parole, ipotizziamo che la qualità dell'aria sia una variabile latente che influenza direttamente le misurazioni di inquinamento e le condizioni atmosferiche.

La rete è stata addestrata utilizzando il Metodo di Massima Verosimiglianza (Maximum Likelihood Estimation) sui dati raccolti dal dataset CSV. Questo metodo permette di stimare le distribuzioni di probabilità condizionate per ogni variabile, dati i valori delle variabili genitore nella rete.

L'addestramento della rete bayesiana ha consentito di ottenere una rappresentazione accurata delle relazioni probabilistiche tra le variabili, facilitando l'inferenza di probabilità su diverse combinazioni di valori osservati.

Utilizzando l'algoritmo di Eliminazione delle Variabili (Variable Elimination), abbiamo eseguito inferenze sulla rete bayesiana addestrata. Ad esempio, abbiamo calcolato la probabilità della qualità dell'aria (classe_qualita) dato un valore specifico di temperatura (temperatura (°C)) e umidità (Umidità (%)).

Esempio di Inferenza:

Probabilità che la qualità dell'aria sia "Moderata" dato che la temperatura è di 25°C e l'umidità è del 60%:

Il modello ha restituito le seguenti probabilità condizionate:

$P(\text{classe_qualita} = \text{"Buona"}) = 0.6\%$

$P(\text{classe_qualita} = \text{"Moderata"}) = 0.25\%$

$P(\text{classe_qualita} = \text{"Scadente"}) = 0.04\%$

$P(\text{classe_qualita} = \text{"Molto Scadente"}) = 0.00\%$

$P(\text{classe_qualita} = \text{"Molto molto Scadente"}) = 0.05\%$

```
INFO:matplotlib.font_manager:generated new fontManager
+-----+
| classe_qualita_encoded | phi(classe_qualita_encoded) |
+-----+
| classe_qualita_encoded(-1) | 0.0074 |
+-----+
| classe_qualita_encoded(0) | 0.6325 |
+-----+
| classe_qualita_encoded(1) | 0.2530 |
+-----+
| classe_qualita_encoded(2) | 0.0475 |
+-----+
| classe_qualita_encoded(3) | 0.0000 |
+-----+
| classe_qualita_encoded(4) | 0.0595 |
+-----+
```

Questo tipo di inferenza è particolarmente utile per stimare la qualità dell'aria in base a condizioni osservate, fornendo informazioni utili per la pianificazione e la gestione ambientale.

Interpretazione e Utilizzo dei Risultati

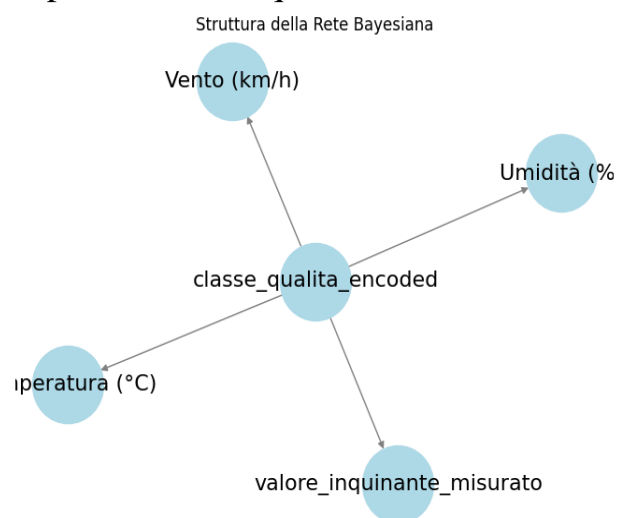
I risultati ottenuti dalla rete bayesiana permettono di:

Comprendere le Relazioni Causali: La struttura della rete mostra le relazioni di dipendenza tra le variabili, aiutando a identificare quali fattori influenzano maggiormente la qualità dell'aria.

Effettuare Previsioni: Utilizzando

l'inferenza, possiamo stimare la probabilità della qualità dell'aria in base a nuove osservazioni delle variabili ambientali.

Supportare Decisioni Ambientali: Le informazioni derivate dalla rete bayesiana possono supportare la definizione di politiche e interventi per migliorare la qualità dell'aria, monitorando le condizioni che maggiormente contribuiscono al deterioramento.



Algoritmo di path finding con A*

Sommario

Il progetto di monitoraggio ambientale e controllo della qualità dell'aria mira a fornire previsioni accurate e in tempo reale sulla qualità dell'aria in diverse aree geografiche, utilizzando avanzati modelli di machine learning e algoritmi di ottimizzazione. Tra i vari strumenti utilizzati, l'algoritmo di path finding A* svolge un ruolo chiave nell'ottimizzazione delle rotte per la raccolta dati e nella gestione delle emergenze ambientali. Questo documento fornisce una panoramica delle decisioni di progetto, degli strumenti utilizzati, dei modelli esclusi, e della valutazione delle prestazioni dell'algoritmo A*.

Strumenti Utilizzati

A*: Algoritmo di ricerca del percorso ottimale, utilizzato per ottimizzare le rotte di raccolta dati e per simulare la dispersione degli inquinanti.

K-means: Algoritmo di clustering utilizzato per identificare pattern nei dati di qualità dell'aria.

Regressione Logistica: Modello statistico per la classificazione dei dati relativi alla qualità dell'aria.

```
import networkx as nx
import matplotlib.pyplot as plt
from queue import PriorityQueue

def heuristic(a, b):
    """Funzione euristica per A* (distanza euclidea)"""
    return ((a[0] - b[0]) ** 2 + (a[1] - b[1]) ** 2) ** 0.5

def a_star_algorithm(graph, start, goal):
    """Algoritmo A* per trovare il percorso più breve"""
    # Coda di priorità per i nodi da esplorare
    open_set = PriorityQueue()
    open_set.put((0, start))

    # Dizionario per memorizzare il percorso
    came_from = {start: None}

    # Distanza attuale dal nodo iniziale
    g_score = {node: float('inf') for node in graph.nodes}
    g_score[start] = 0

    # Stima del costo totale dal nodo iniziale al nodo finale passando per il nodo corrente
    f_score = {node: float('inf') for node in graph.nodes}
    f_score[start] = heuristic(graph.nodes[start]['pos'], graph.nodes[goal]['pos'])

    while not open_set.empty():
        current = open_set.get()[1]
```

Decisioni di Progetto

L'integrazione dell'algoritmo A* nel sistema di monitoraggio è stata decisa per affrontare specifiche esigenze di ottimizzazione spaziale, come l'efficienza nella raccolta dati e la risposta rapida in situazioni di emergenza ambientale. A* è stato scelto per la sua capacità di trovare percorsi ottimali in spazi di ricerca complessi, mantenendo un equilibrio tra efficienza computazionale e precisione.

Ottimizzazione delle Rotte di Raccolta Dati:

A* è stato implementato per determinare le rotte ottimali per sensori mobili, droni o veicoli che raccolgono dati ambientali, riducendo il tempo e i costi operativi.

Simulazione della Dispersione degli Inquinanti: L'algoritmo è stato utilizzato per modellare e prevedere il percorso di dispersione di inquinanti nell'aria, in base a variabili ambientali come la direzione del vento e la topografia.

Gestione delle Emergenze Ambientali: A* è stato applicato per ottimizzare la risposta alle emergenze, trovando i percorsi più sicuri e rapidi per evacuazioni o interventi di emergenza.

Modelli Non Utilizzati

Nonostante il potenziale di altri algoritmi di ricerca del percorso, come Dijkstra o Breadth-First Search (BFS), A* è stato preferito per la sua efficienza superiore e la capacità di incorporare euristiche per migliorare la ricerca del percorso. Modelli di machine learning come Support Vector Machines (SVM) sono stati considerati ma esclusi in favore di Regressione Logistica, che ha dimostrato prestazioni migliori nel contesto specifico.

Valutazione

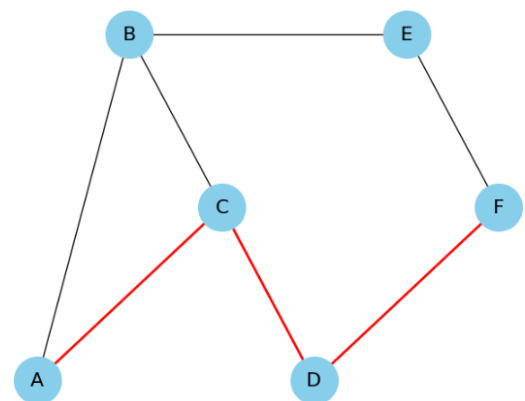
L'efficacia dell'algoritmo A* è stata valutata attraverso simulazioni e scenari reali, dove l'algoritmo ha mostrato una capacità notevole di ottimizzare percorsi di raccolta dati e di risposta a emergenze ambientali. Le prestazioni sono state misurate in termini di:

Tempo di Calcolo: A* ha dimostrato un'efficienza superiore rispetto ad altri algoritmi di path finding, grazie all'uso di euristiche ben progettate.

Accuratezza del Percorso: L'algoritmo ha fornito percorsi ottimali che hanno minimizzato il tempo di viaggio e i costi operativi, garantendo al contempo la copertura completa delle aree monitorate.

Robustezza nelle Simulazioni: Durante la simulazione della dispersione degli inquinanti, A* ha mostrato una buona capacità di adattamento alle variabili ambientali, fornendo previsioni accurate dei percorsi di dispersione. L'integrazione di A* nel sistema ha quindi contribuito a migliorare significativamente l'efficienza operativa e la capacità del sistema di rispondere in modo rapido e preciso a cambiamenti ambientali critici.

Percorso trovato da A a F: ['A', 'C', 'D', 'F']



Conclusione

Il progetto "Sistema di Monitoraggio Ambientale e Controllo della Qualità dell'Aria basato sull'Intelligenza Artificiale" ha dimostrato l'efficacia di un approccio integrato, che combina vari algoritmi avanzati per migliorare il monitoraggio e la previsione della qualità dell'aria in tempo reale. Gli algoritmi utilizzati, tra cui Alberi di Decisione, Regressione Logistica, K-means, Reti Bayesiane e l'algoritmo di path finding A*, hanno permesso di affrontare in modo efficace le complesse sfide legate al monitoraggio ambientale. Questo approccio integrato ha permesso di costruire un sistema altamente efficace e flessibile, in grado di fornire previsioni accurate e supportare decisioni rapide in contesti critici. La capacità del sistema di adattarsi e migliorare continuamente grazie all'apprendimento dai dati rappresenta un significativo passo avanti nella gestione ambientale e nel controllo della qualità dell'aria. L'uso combinato di diversi algoritmi ha non solo aumentato la precisione delle previsioni, ma ha anche fornito una comprensione più profonda dei fenomeni ambientali complessi che influenzano la qualità dell'aria.

In conclusione, questo progetto dimostra l'importanza di un approccio multidisciplinare che combina algoritmi di intelligenza artificiale e tecniche di ottimizzazione per affrontare problemi complessi come il monitoraggio della qualità dell'aria. Tale sistema non solo migliora la capacità di risposta alle emergenze ambientali, ma contribuisce anche a una gestione più sostenibile e informata delle risorse ambientali, con potenziali benefici a lungo termine per la salute pubblica e l'ambiente.

Riferimenti Bibliografici

1. <https://dati.puglia.it/ckan/dataset/dati-qualita-aria-2023>
2. <https://www.geeksforgeeks.org/learn-data-structures-and-algorithms-dsa-tutorial/?ref=home-articlecards>
3. <https://www.ilmeteo.it/meteo/Bari>
4. <https://scikit-learn.org/stable/>
5. <https://www.kaggle.com/code/vbmokin/air-quality-city-prediction-mapping>
6. <https://artint.info/3e/resources/index.html>
7. <https://it.wikipedia.org/wiki/Prolog>
8. <https://artint.info/2e/html2e/ArtInt2e.Ch7.S3.SS1.html>
9. <https://artint.info/AIPython/>
10. <https://www.researchgate.net/>
11. <https://www.openstreetmap.org/export#map=15/41.12060/16.86820&layers=G>