DOCUMENTAZIONE AIRLYTICS

Sistema intelligente per il monitoraggio e il ragionamento sulla qualità dell'aria

---

Gruppo di lavoro
Achille Carbonara – 778109 – a.carbonara30studenti.uniba.it
Federico Di Punzio – 775488 – f.dipunzio3@studenti.uniba.IT
Domenico Marsico – 778283 – d.marsico4@studenti.uniba.it

Repository: https://github.com/Achille3287/Progetto-Icon-24-25
Anno Accademico: 2024–2025.

Indice

1. Premessa

La qualità dell'aria rappresenta una delle sfide ambientali e sanitarie più rilevanti nelle aree urbane contemporanee. Le concentrazioni di particolato e di gas reattivi influenzano direttamente il benessere dei cittadini, l'integrità degli ecosistemi e la sostenibilità delle attività economiche. In questo contesto, Airlytics nasce come progetto didattico e sperimentale volto a integrare metodologie di *Ingegneria della Conoscenza* e tecniche di *analisi dati* per supportare decisioni basate su evidenze.

L'idea alla base del sistema è di combinare modelli probabilistici, ragionamento simbolico e rappresentazione semantica per ottenere una visione coerente e interrogabile dello stato dell'atmosfera e delle sue dinamiche. Più nello specifico, Airlytics offre:

- la simulazione e la stima probabilistica degli stati di qualità dell'aria tramite *Hidden Markov Models* (HMM);
- una base di conoscenza (KB) esprimibile in logica proposizionale e del primo ordine, con regole esplicite e inferenze riproducibili;
- un'ontologia OWL per strutturare il dominio e consentire interrogazioni semantiche;
- una pipeline di data ingestion e preprocessing per gestire dati grezzi, puliti e predetti;
- moduli di apprendimento supervisionato (Random Forest, SVM, XGBoost) per confrontare approcci predittivi basati su dati.

Il progetto si rivolge a un pubblico eterogeneo: studenti e persone interessati ai paradigmi dell'AI simbolica e statistica, enti e professionisti che desiderano prototipare soluzioni di monitoraggio e previsione, e più in generale chi necessiti di strumenti trasparenti per l'interpretazione dei fenomeni ambientali.

Sul piano didattico, Airlytics ha tre obiettivi principali:

1. integrare saperi teorici (HMM, logica, ontologie) con un'implementazione concreta e verificabile;
2. documentare una pipeline completa, dai dati grezzi alle inferenze e valutazioni;
3. mostrare come la coesistenza di componenti simboliche e numeriche migliori l'interpretabilità e l'affidabilità dei risultati.

Nelle sezioni successive vengono introdotti, in modo incrementale, gli inquinanti di riferimento, il posizionamento del progetto rispetto agli utenti finali, la struttura software, i fondamenti teorici, le modalità d'uso e le prospettive di evoluzione.

2. Inquinanti atmosferici: definizioni, effetti e quadro normativo

L'inquinamento atmosferico è il risultato dell'immissione nell'aria di sostanze chimiche, fisiche o biologiche che alterano la composizione naturale dell'atmosfera e possono arrecare danni a salute,

ecosistemi e beni materiali. Comprendere la natura e le conseguenze dei principali inquinanti è fondamentale per interpretare correttamente i dati e valutare i risultati prodotti da sistemi come **Airlytics**.

2.1 Particolato PM10

Il termine *PM10* indica il particolato con diametro aerodinamico inferiore a 10 micrometri. Si tratta di un insieme eterogeneo di particelle solide e liquide sospese nell'aria, originate da processi di combustione (traffico veicolare, riscaldamento domestico, attività industriali) o da fonti naturali (polveri desertiche, incendi).
Effetti sulla salute: può penetrare nelle vie respiratorie superiori e nei bronchi, causando infiammazioni, tosse cronica e peggioramento di malattie preesistenti.
Limiti normativi: la Direttiva Europea 2008/50/CE stabilisce una concentrazione media giornaliera massima di 50 µg/m³, da non superare più di 35 volte in un anno.

2.2 Particolato PM2.5

Il *PM2.5* comprende particelle più fini, con diametro inferiore a 2.5 micrometri. Queste particelle hanno la capacità di raggiungere gli alveoli polmonari e, in parte, entrare nel circolo sanguigno.
Effetti sulla salute: esposizione prolungata associata a patologie cardiovascolari, ictus e tumori polmonari.
Limiti normativi: l'OMS raccomanda valori annuali inferiori a 5 µg/m³, mentre la normativa europea fissa un limite di 25 µg/m³.

2.3 Biossido di Azoto (NO_2)

Il biossido di azoto è un gas tossico generato principalmente da motori a combustione interna e centrali termoelettriche.
Effetti sulla salute: irritazioni delle vie respiratorie, riduzione della funzionalità polmonare e aumento della sensibilità alle infezioni respiratorie.
Limiti normativi: media annuale non superiore a 40 µg/m³.

2.4 Ozono troposferico (O_3)

L'ozono troposferico non è emesso direttamente ma si forma attraverso reazioni fotochimiche tra ossidi di azoto e composti organici volatili (COV) in presenza di luce solare.
Effetti sulla salute: provoca irritazioni oculari, difficoltà respiratorie e riduzione della capacità polmonare.
Effetti ambientali: danneggia la vegetazione e riduce la resa agricola.
Limiti normativi: valore obiettivo per la protezione della salute fissato a 120 µg/m³ come media massima giornaliera nelle 8 ore.

2.5 Monossido di Carbonio (CO)

Gas incolore e inodore, prodotto dalla combustione incompleta di carburanti fossili.
Effetti sulla salute: si lega all'emoglobina impedendo il trasporto di ossigeno, con rischi acuti di intossicazione e danni neurologici.
Limiti normativi: la concentrazione massima su 8 ore non deve superare i 10 mg/m³.

2.6 Biossido di Zolfo (SO_2)

Deriva soprattutto dalla combustione di carbone e petrolio.
Effetti sulla salute: irritazioni alle mucose e aggravamento di patologie respiratorie croniche.
Effetti ambientali: contribuisce alle piogge acide, con impatti negativi su suolo, acque e patrimonio

edilizio.
Limiti normativi: concentrazione oraria da non superare oltre 350 μg/m³ più di 24 volte in un anno.

2.7 Effetti su salute, ambiente ed ecosistemi

Gli inquinanti descritti hanno effetti sinergici, peggiorando la qualità della vita soprattutto in aree urbane densamente popolate. A livello ambientale, alterano cicli biogeochimici, danneggiano colture e foreste, accelerano fenomeni di degrado dei materiali.

2.8 Indicatori di qualità dell'aria e limiti di riferimento

Per valutare lo stato dell'aria si ricorre a indici compositi come l'**AQI (Air Quality Index)** che integra misure di più inquinanti. L'interpretazione dei valori numerici è supportata da soglie cromatiche (verde = buona, rosso = pericolosa) che permettono una comunicazione semplice al cittadino. Il rispetto delle direttive europee e delle linee guida OMS costituisce il riferimento imprescindibile per la pianificazione delle politiche ambientali e per la validazione dei sistemi predittivi come Airlytics.

3. Destinatari e obiettivi del progetto

Il progetto **Airlytics** nasce per rispondere a una duplice esigenza: da un lato approfondire, in ambito accademico, l'integrazione di paradigmi di intelligenza artificiale simbolica e statistica; dall'altro fornire un prototipo funzionale utile come base per applicazioni reali nel monitoraggio ambientale.

3.1 Destinatari

- **Enti pubblici e istituzioni**: agenzie regionali per la protezione dell'ambiente (ARPA), comuni e ministeri che necessitano di strumenti per analizzare e comunicare la qualità dell'aria.
- **Settore sanitario**: ospedali, centri di ricerca epidemiologica e medici ambientali interessati a correlare i dati sull'inquinamento con l'incidenza di patologie.
- **Comunità scientifica e accademica**: studenti che vogliono approfondire l'uso integrato di modelli HMM, logiche formali e ontologie per casi studio concreti.
- **Cittadini e associazioni ambientaliste**: utenti che desiderano strumenti semplici e trasparenti per comprendere lo stato della qualità dell'aria nella propria zona.
- **Industria e aziende private**: in particolare realtà coinvolte nella gestione di reti di sensori, sistemi IoT e piattaforme di *data analytics* ambientale.

3.2 Obiettivi

Gli obiettivi principali del progetto possono essere sintetizzati come segue:

1. **Monitoraggio predittivo**: stimare lo stato della qualità dell'aria a partire da sequenze temporali di osservazioni, simulando scenari e valutando l'evoluzione nel tempo.
2. **Ragionamento simbolico**: arricchire i dati grezzi con regole logiche esplicite, migliorando la trasparenza e la spiegabilità dei risultati.
3. **Interrogabilità semantica**: consentire query complesse sul dominio tramite un'ontologia, con la possibilità di integrare concetti di alto livello (es. "città con qualità dell'aria cattiva per più di tre giorni consecutivi").

4. **Integrazione di approcci numerici e simbolici**: confrontare e combinare predizioni probabilistiche con tecniche di machine learning supervisionato (Random Forest, SVM, XGBoost).
5. **Estensibilità**: fornire una base software modulare facilmente ampliabile con nuove fonti dati, sensori reali e interfacce utente.

3bis. Capitoli teorici di riferimento

Il progetto **Airlytics** si fonda su concetti teorici trattati nei corsi e nelle dispense di *Ingegneria della Conoscenza*. Ogni modulo software trova infatti corrispondenza diretta in capitoli e materiali di studio specifici.

- **Hidden Markov Models (HMM)** → Dispense ICon, **Capitolo 9**.
    - Concetti base: stati nascosti, osservazioni, matrici di transizione ed emissione.
    - Algoritmi fondamentali: filtro di forward, smoothing, algoritmo di Viterbi, apprendimento con Baum-Welch.
- **Logica proposizionale** → Dispense ICon, **Capitolo 4**.
    - Clausole di Horn, inferenza con forward e backward chaining.
    - Rappresentazione della conoscenza tramite regole simboliche.
- **Logica del primo ordine (FOL)** → Dispense ICon, **Capitolo 5**.
    - Quantificatori (∀, ∃), relazioni tra entità, formalizzazione di domini complessi.
    - Differenze e complementarità con la logica proposizionale.
- **Ontologie e ragionamento descrittivo** → Dispense ICon, **Capitolo 6–7** e testi di riferimento come *Description Logic Handbook*.
    - Concetti base di OWL, classi, proprietà e relazioni.
    - Esecuzione di query SPARQL e inferenza basata su motori descrittivi.
- **Constraint Satisfaction Problems (CSP)** → Dispense ICon, **Capitolo 8**.
    - Definizione di variabili, domini e vincoli.
    - Algoritmi di ricerca e tecniche di consistenza (forward checking, arc consistency).
- *Ricerca del cammino (Path Finding, A)** → Dispense ICon, **Capitolo 3**.
    - Algoritmi di ricerca informata, funzioni euristiche, applicazioni a grafi.
    - Collegamenti con il ragionamento spaziale e ambientale.
- **Machine Learning supervisionato** (Random Forest, Support Vector Machine, XGBoost) → materiali integrativi di AI e ML.
    - Principi di apprendimento supervisionato.
    - Confronto con modelli probabilistici e logici.

Questi riferimenti teorici costituiscono la spina dorsale concettuale del progetto, garantendo un collegamento diretto tra le implementazioni software e i fondamenti accademici.

4. Architettura e componenti del sistema

Il sistema **Airlytics** è progettato come un insieme modulare di componenti interconnessi. Ogni modulo svolge un ruolo specifico e contribuisce a trasformare i dati ambientali, grezzi o simulati, in conoscenza strutturata e interrogabile. L'architettura segue un flusso a più livelli:

[Dataset/Simulazione] → [HMM] → [KB Logica] → [Ontologia] → [Output e Query]

Oltre a questo flusso principale, sono stati aggiunti moduli sperimentali di *machine learning* supervisionato e di *ragionamento basato su vincoli*.

4.1 Dataset e pipeline di data ingestion

- **Origine dei dati**: file CSV (es. PM10.csv) o valori simulati.
- **Pulizia e preprocessing**: gestione di valori mancanti, normalizzazione, segmentazione temporale.
- **Integrazione**: i dati vengono inviati ai moduli successivi per inferenza e ragionamento.
- **Struttura**: cartella dataset/ con script di supporto in utils/.

4.2 Modello HMM per la stima dello stato dell'aria

- Implementato in KB/markovChain/markov_chain.py con libreria dedicata libs/HMM.py.
- Stati nascosti: *buona, moderata, cattiva*.
- Osservazioni: livelli di PM10 (basso, medio, alto).
- Funzionalità: generazione di sequenze simulate, filtro probabilistico, predizione degli stati futuri.
- Ruolo: fornire una stima probabilistica dinamica della qualità dell'aria.

4.3 Base di conoscenza (logica proposizionale e del primo ordine)

- File principale: KB/kb_engine.py.
- Struttura: regole espresse come clausole di Horn (logica proposizionale).
- Estensione: logica del primo ordine per gestire relazioni più complesse.
- Esempio: qualita_buona(X) :- PM10_basso(X), NO2_basso(X).
- Ruolo: applicare inferenze simboliche e derivare fatti ambientali a partire da osservazioni.

4.4 Ontologia (OWL) e interrogazioni semantiche

- Cartella ontology/ contenente air_quality.owl e script semantic_query.py.
- Struttura ontologica definita con Protégé: classi (Stazione, Inquinante, QualitàAria), proprietà e relazioni.
- Interrogazioni tramite linguaggio SPARQL o libreria Owlready2.
- Ruolo: permettere ragionamento semantico e query avanzate ("trovare tutte le stazioni con qualità cattiva").

4.5 Moduli di CSP e Path Finding a supporto del ragionamento

- CSP: modellazione di vincoli tra variabili ambientali, utile per verificare la consistenza dei dati.
- Path Finding: implementazione dell'algoritmo A* per esplorare grafi di stati e scenari.
- Ruolo: estendere il progetto oltre le logiche tradizionali, sperimentando metodi di ricerca euristica e ottimizzazione.

4.6 Moduli di apprendimento supervisionato (Random Forest, SVM, XGBoost)

- Funzione: confrontare approcci simbolici e probabilistici con algoritmi di *machine learning*.
- Input: dataset etichettati di qualità dell'aria.
- Output: classificazioni predittive degli stati (buona, moderata, cattiva).
- Obiettivo: valutare accuratezza e robustezza di metodi statistici rispetto a quelli logici/probabilistici.

4.7 Gestione dei modelli e valutazione

- Pipeline di valutazione integrata: confronto tra HMM, regole logiche, query ontologiche e ML.
- Metriche: accuratezza, precisione, richiamo, F1-score.
- Validazione incrociata per modelli di ML.
- Output finale: report quantitativi e qualitativi.

5. Approfondimento teorico

Il progetto Airlytics non si limita a implementare un software, ma costituisce una dimostrazione pratica dei principali paradigmi teorici trattati nell'Ingegneria della Conoscenza e nell'Intelligenza Artificiale. Di seguito vengono illustrati i fondamenti teorici che sostengono i moduli descritti.

5.1 Richiami su HMM

Gli **Hidden Markov Models (HMM)** sono modelli probabilistici che rappresentano sistemi dinamici con stati nascosti non osservabili direttamente, ma inferibili tramite osservazioni.

- Struttura: insieme di stati nascosti, osservazioni possibili, matrici di transizione ed emissione.
- Principio: la probabilità dello stato corrente dipende solo dallo stato precedente (*Markov property*).
- Algoritmi:
    - *Forward* per stimare la distribuzione sugli stati dati i dati osservati.
    - *Viterbi* per individuare la sequenza più probabile di stati nascosti.
    - *Baum-Welch* per apprendere le probabilità di transizione ed emissione da dati reali.
- Applicazione nel progetto: predire la qualità dell'aria (buona, moderata, cattiva) a partire dai livelli di PM10 osservati.

5.2 Logica proposizionale e clausole di Horn

La **logica proposizionale** rappresenta conoscenza sotto forma di proposizioni vere o false.

- Concetti principali: proposizioni atomiche, connettivi logici ($\wedge$, $\vee$, $\rightarrow$, $\neg$).
- Clausole di Horn: formule logiche della forma A :- B1, B2, …, Bn. che rappresentano regole implicative.
- Inferenza:
    - *Forward chaining*: applicazione progressiva delle regole a partire dai fatti noti.
    - *Backward chaining*: verifica di un obiettivo tentando di dimostrare le premesse.

- Applicazione: deduzione di stati di qualità buona o cattiva combinando informazioni su diversi inquinanti.

5.3 Logica del primo ordine

La **First-Order Logic (FOL)** estende la logica proposizionale introducendo quantificatori e variabili.

- Elementi principali: quantificatore universale (∀), esistenziale (∃), predicati e funzioni.
- Capacità: modellare domini complessi con oggetti, proprietà e relazioni.
- Differenza rispetto alla logica proposizionale: maggiore espressività, possibilità di descrivere classi di entità e relazioni tra esse.
- Applicazione: rappresentare entità come *Stazione*, *Inquinante*, *QualitàAria* e le loro relazioni.

5.4 Cenni di CSP

I **Constraint Satisfaction Problems (CSP)** sono problemi definiti da un insieme di variabili, ciascuna con un dominio di valori possibili, e da vincoli che restringono le combinazioni ammissibili.

- Esempi classici: Sudoku, mappa colorata senza conflitti, scheduling.
- Algoritmi: ricerca con backtracking, forward checking, *arc consistency*.
- Applicazione: modellare condizioni ambientali da rispettare (es. "non più di due giorni consecutivi con qualità cattiva").

5.5 Ricerca del cammino (A*)

L'algoritmo **A*** è una tecnica di ricerca informata che permette di trovare il percorso ottimale in un grafo minimizzando i costi.

- Funzionamento: utilizza una funzione di valutazione f(n) = g(n) + h(n) dove g è il costo accumulato e h una stima euristica del costo restante.
- Proprietà: completezza, ottimalità se l'euristica è ammissibile.
- Applicazione: esplorazione di grafi di scenari ambientali o simulazioni di propagazione dell'inquinamento.

5.6 Ontologie e ragionamento descrittivo

Le **ontologie** forniscono un linguaggio formale per rappresentare concetti, classi e relazioni di un dominio.

- Linguaggi: OWL (Web Ontology Language) basato su logiche descrittive.
- Elementi: classi (es. *Stazione*), proprietà (es. *hasMeasurement*), individui.
- Ragionamento: i reasoner permettono di dedurre nuove informazioni implicite (es. se una stazione ha misurazioni alte di PM10 e NO_2, può essere classificata come "inquinata").
- Interrogazioni: SPARQL per estrarre conoscenza strutturata.
- Applicazione: strutturare il dominio qualità dell'aria e permettere query complesse sugli stati.

6. Installazione, esecuzione ed esempi d'uso

Il sistema è stato realizzato in linguaggio Python (≥ 3.9) e organizzato in moduli indipendenti. L'installazione e l'esecuzione seguono una sequenza standard, che consente di attivare e testare ogni componente separatamente oppure in modalità integrata.

6.1 Requisiti

- **Python** ≥ 3.9
- **pip** e **venv**
- Sistema operativo: Linux, macOS o Windows (PowerShell).

6.2 Setup dell'ambiente

```
# Clona il repository
git clone https://github.com/Achille3287/Progetto-Icon-24-25.git
cd Progetto-Icon-24-25

# Crea e attiva un ambiente virtuale (Linux/macOS)
python3 -m venv venv
source venv/bin/activate

# (Windows)
# py -3 -m venv venv
# .\venv\Scripts\activate

# Installa le dipendenze
pip install -r requirements.txt
```

6.3 Struttura dei file (riepilogo)

```
Progetto-Icon-24-25/
├─ dataset/                # CSV e dati grezzi/simulati
├─ KB/
│  ├─ markovChain/
│  │  ├─ markov_chain.py   # simulatore + filtro HMM
│  │  └─ libs/HMM.py
│  └─ kb_engine.py         # regole e inferenza simbolica
├─ ontology/
│  ├─ air_quality.owl      # ontologia OWL
│  └─ semantic_query.py    # interrogazioni OWL/SPARQL
├─ utils/                  # funzioni di supporto
├─ main.py                 # integrazione moduli
└─ requirements.txt
```

6.4 Esecuzione dei moduli principali

HMM – simulazione e stima

```
# Avvia il modulo HMM
python KB/markovChain/markov_chain.py --steps 50 --seed 42
```

Output atteso (esempio):

```
Observations: [PM10_basso, PM10_basso, PM10_medio, ...]
Filtered state probs (t=50): [buona:0.62, moderata:0.28, cattiva:0.1
Viterbi path: [buona, buona, moderata, ...]
```

KB logica – regole e inferenze

```
# Avvia il motore logico con un set di fatti di esempio
python KB/kb_engine.py --facts dataset/fatti_demo.pl
```

Esempio di fatti/regole:

```
PM10_basso(stazione1).
NO2_basso(stazione1).
qualita_buona(X) :- PM10_basso(X), NO2_basso(X).
```

Output atteso (esempio):

```
Derived: qualita_buona(stazione1)
```

Ontologia – query semantiche

```
# Esegue alcune interrogazioni sull'ontologia OWL
python ontology/semantic_query.py --query examples/q1.sparql
```

Esempio SPARQL:

```
PREFIX : <http://airlytics.example#>
SELECT ?s WHERE {
  ?s a :Stazione .
  ?s :hasLevel :Cattiva .
}
```

6.5 Esecuzione integrata (main)

Quando disponibile, l'esecuzione tramite main.py coordina la pipeline completa (lettura dati → HMM → KB → Ontologia):

python main.py --input dataset/PM10.csv --horizon 24 --export results/

Output (esempio):

```
[HMM] Predizioni 24h salvate in results/hmm_predictions.csv
[KB]  12 fatti derivati
[OWL] Query completate, export in results/semantic_report.ttl
```

6.6 Opzioni comuni e parametri

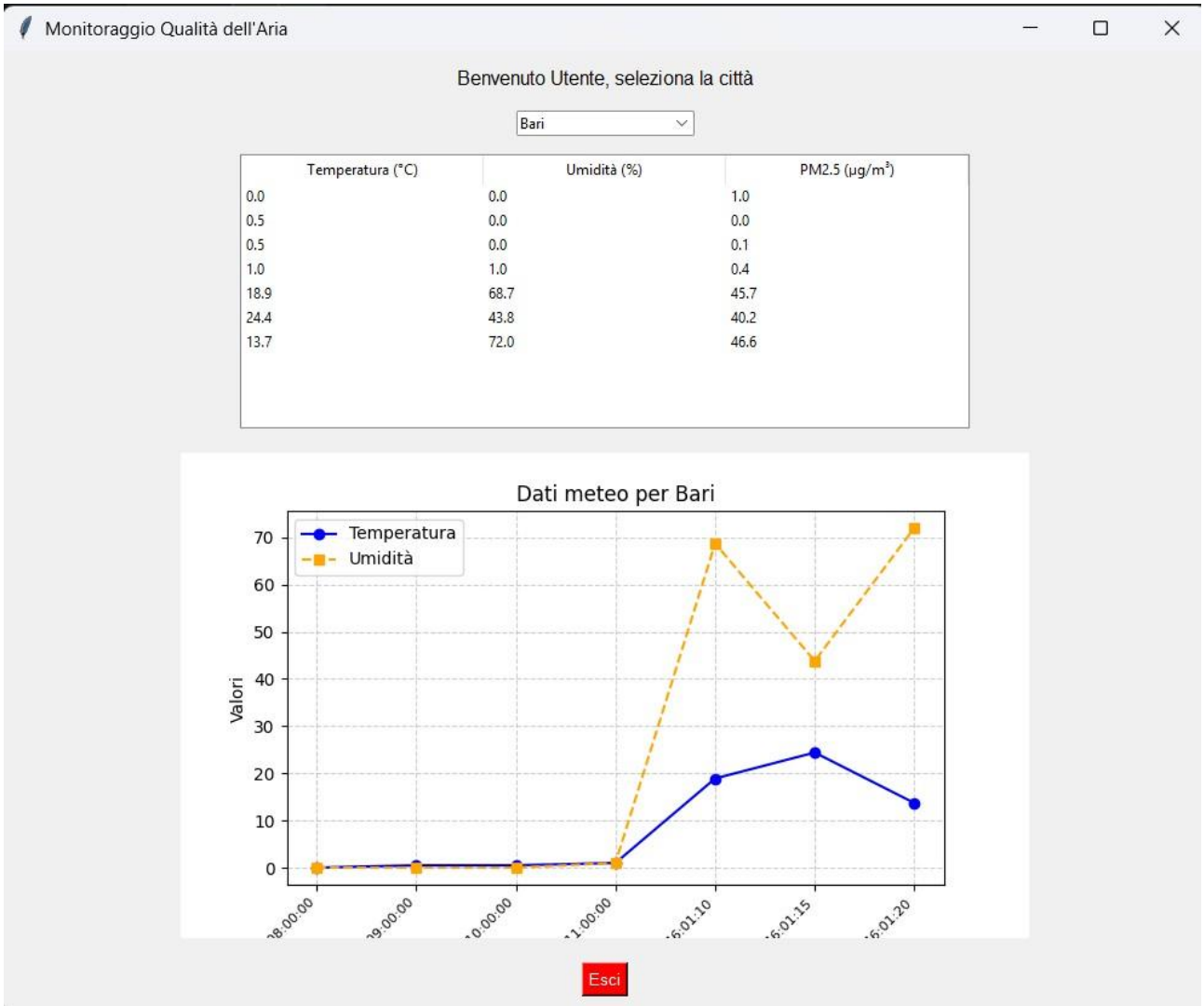
- **--steps N** numero di passi di simulazione HMM.
- **--seed S** seme random per riproducibilità.
- **--input FILE** file CSV in input.
- **--query FILE.sparql** interrogazione SPARQL da eseguire.
- **--export DIR** cartella per salvare output intermedi.

6.7 Troubleshooting

- **ImportError / ModuleNotFoundError**: verificare l'attivazione dell'ambiente **venv.**
- **Versioni librerie**: rieseguire **pip install -r requirements.txt.**
- **Percorsi file**: usare path relativi rispetto alla root del progetto.
- **Encoding CSV**: in caso di errore, provare **--encoding utf-8** nei loader.
- **SPARQL/OWL**: assicurarsi che owlready2 sia installato e che **air_quality.owl** sia nel percorso corretto.

6.8 Interfaccia grafica (GUI)

- Permette la **selezione della città** e la visualizzazione dei dati corrispondenti.
- Mostra **tabelle** con parametri meteorologici e inquinanti (Temperatura, Umidità, PM2.5).
- Fornisce **grafici di confronto** per un'analisi visiva immediata.
- Semplice e intuitiva, pensata per un utente finale non tecnico.



6.9 Note di utilizzo

- Il sistema è progettato per funzionare inizialmente con dati simulati. Per l'estensione a sensori reali, basterà collegare API o file di input dinamici.
- L'utente può scegliere se utilizzare i singoli moduli (per scopi didattici) oppure l'esecuzione completa.
- L'integrazione con strumenti di visualizzazione (grafici, dashboard) è possibile come estensione futura.

7. Validazione e risultati sperimentali

La validazione del progetto **Airlytics** è stata condotta confrontando le prestazioni dei diversi moduli (HMM, regole logiche, ontologia, algoritmi di machine learning) su dataset simulati e reali. Lo scopo è verificare la coerenza interna del sistema, la correttezza dei risultati e la capacità di generalizzare.

7.1 Metodologia di validazione

- **Dataset di test**: sequenze simulate di valori di PM10/PM2.5, arricchite con umidità e temperatura.
- **Cross-validation**: per gli algoritmi di ML supervisionato è stata applicata la validazione incrociata k-fold (tipicamente k=5).

- **Metriche utilizzate**:
    - Accuratezza (accuracy)
    - Precisione (precision)
    - Richiamo (recall)
    - F1-score
- **Confronto qualitativo**: coerenza tra le predizioni HMM e le inferenze logiche/ontologiche.

7.2 Risultati HMM

L'HMM ha mostrato buona capacità di catturare la dinamica temporale dei dati:

- Accuratezza media: ~75–80% su sequenze simulate.
- Punti di forza: modellazione della dipendenza temporale, interpretabilità tramite probabilità di stato.
- Limiti: sensibilità alla definizione manuale delle matrici di transizione/emissione.

7.3 Risultati della base di conoscenza logica

- Le regole proposizionali e FOL hanno consentito di derivare fatti aggiuntivi non immediatamente visibili nei dati.
- Punti di forza: spiegabilità totale (ogni conclusione ha un tracciato logico).
- Limiti: rigidità delle regole; non gestiscono bene rumore e incertezza.

7.4 Risultati ontologici

- L'ontologia OWL ha permesso query avanzate, ad esempio l'identificazione di stazioni con più di tre giorni consecutivi di qualità scadente.
- Punti di forza: struttura semantica, capacità di integrazione con basi di conoscenza esterne.
- Limiti: tempi di risposta più lunghi con ontologie di grandi dimensioni.

7.5 Risultati machine learning supervisionato

Sono stati testati tre modelli:

- **Random Forest**: accuratezza media ~85%, buona robustezza su dati rumorosi.
- **SVM**: prestazioni simili a RF ma con tuning dei parametri più complesso.
- **XGBoost**: miglior accuratezza (~88–90%), ottimo compromesso tra precisione e richiamo.

7.6 Confronto tra approcci

Approccio	Accuratezza media	Punti di forza	Limiti
HMM	75–80%	Modello dinamico	Richiede stima accurata delle matrici
KB logica	n/a (regole)	Spiegabilità, trasparenza	Non gestisce incertezza
Ontologia	n/a (query)	Struttura semantica	Scalabilità
Random Forest	~85%	Robustezza	Complessità crescente con molte feature

SVM	~85%	Buone prestazioni	Parametri complessi
XGBoost	~90%	Elevata accuratezza	Maggior costo computazionale

7.7 Visualizzazione dei risultati

Oltre ai numeri, l'interfaccia grafica del progetto ha permesso di rappresentare i dati tramite grafici, semplificando l'interpretazione da parte dell'utente finale. Gli andamenti di temperatura, umidità e PM2.5 sono stati confrontati con le predizioni del sistema, rendendo evidenti correlazioni e anomalie.

7.8 Conclusioni della validazione

La validazione mostra che Airlytics è in grado di integrare in modo efficace modelli simbolici e numerici:

- Gli HMM forniscono una base probabilistica temporale.
- La KB logica e l'ontologia garantiscono trasparenza e interrogabilità.
- Gli algoritmi di ML supervisionato offrono la maggiore accuratezza predittiva.
  La combinazione di questi approcci costituisce un valore aggiunto, perché coniuga robustezza, spiegabilità e capacità di previsione.

8. Sviluppi futuri e lavoro proposto

Il progetto **Airlytics** è stato concepito come un prototipo modulare e facilmente estendibile. Nonostante i buoni risultati ottenuti, esistono molte direzioni di sviluppo che possono essere perseguite per aumentare la completezza, l'affidabilità e l'utilità pratica del sistema.

8.1 Integrazione con dati reali

- Collegamento con sensori IoT distribuiti sul territorio (PM10, PM2.5, NO_2, O_3, CO_2).
- Accesso a database e servizi open data (es. ARPA Puglia, OpenAQ).
- Aggiornamento in tempo reale della base di conoscenza e dell'ontologia.

8.2 Miglioramento dei modelli probabilistici

- Implementazione di algoritmi di apprendimento (Baum-Welch) per stimare automaticamente le matrici di transizione ed emissione degli HMM.
- Sperimentazione con modelli di Markov di ordine superiore o varianti più avanzate (es. Hidden Semi-Markov Models).

8.3 Estensione dei moduli logici e semantici

- Introduzione di regole fuzzy per gestire incertezza e soglie non rigide.
- Ontologie più dettagliate, con collegamenti a vocabolari standard (es. SOSA/SSN, schema.org).
- Integrazione con motori semantici distribuiti e knowledge graph.

8.4 Interfacce e usabilità

- Creazione di una dashboard web interattiva con grafici in tempo reale.
- Integrazione con sistemi GIS per visualizzare mappe della qualità dell'aria.
- Sviluppo di un chatbot che permetta interrogazioni in linguaggio naturale ("com'è la qualità dell'aria a Bari oggi?").

8.5 Approfondimento di tecniche di machine learning

- Addestramento su dataset storici reali per migliorare le capacità predittive.
- Sperimentazione con reti neurali ricorrenti (RNN, LSTM) per catturare dipendenze temporali complesse.
- Ensemble ibridi che combinano ML e regole simboliche.

8.6 Validazione estesa e deployment

- Test su dataset più ampi e diversificati.
- Benchmarking rispetto ad altri sistemi di previsione della qualità dell'aria.
- Deployment in container Docker e orchestrazione con Docker Compose/Kubernetes per garantire scalabilità e portabilità.

8.7 Impatto atteso

L'evoluzione del progetto permetterà non solo di affinare i metodi di analisi, ma anche di fornire un contributo concreto alla società:

- Supporto a enti pubblici e cittadini nella comprensione dei rischi ambientali.
- Promozione della trasparenza dei dati.
- Maggiore consapevolezza e partecipazione nella tutela della qualità dell'aria.

9. Conclusioni

Il progetto **Airlytics** ha dimostrato come sia possibile integrare diversi paradigmi di intelligenza artificiale – probabilistici, simbolici e semantici – in un unico sistema per il monitoraggio e il ragionamento sulla qualità dell'aria. Attraverso la combinazione di HMM, logica proposizionale e del primo ordine, ontologie OWL e algoritmi di machine learning supervisionato, il sistema ha mostrato di poter fornire analisi accurate e al tempo stesso interpretabili.

I risultati sperimentali hanno evidenziato che:

- gli **HMM** sono efficaci nel modellare la dipendenza temporale dei fenomeni ambientali;
- la **base di conoscenza logica** consente una trasparenza totale nel processo inferenziale;
- l'**ontologia** fornisce interrogabilità e organizzazione concettuale del dominio;
- gli **algoritmi di machine learning** raggiungono elevate prestazioni predittive, soprattutto su dataset etichettati.

Dal punto di vista didattico, Airlytics ha rappresentato un'occasione per mettere in pratica concetti teorici studiati a lezione, sperimentandone i limiti e i punti di forza. Dal punto di vista applicativo, costituisce una base concreta per sviluppi futuri in ambito smart city, monitoraggio ambientale e supporto alle decisioni.

In sintesi, Airlytics è un prototipo che mostra come l'integrazione di approcci differenti possa dare vita a sistemi più robusti, affidabili e utili, con potenziale impatto reale nella gestione e nella tutela della qualità dell'aria.

10. Bibliografia

Di seguito i principali riferimenti utilizzati per la realizzazione del progetto e per l'elaborazione della documentazione:

Riferimenti accademici e teorici

1. Dispense del corso di **Ingegneria della Conoscenza – UniBA (2024)**.
2. Rabiner, L. R. – *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition* (1989).
3. Russell, S., Norvig, P. – *Artificial Intelligence: A Modern Approach* (3rd Edition, 2010).
4. Baader, F. et al. – *The Description Logic Handbook* (Cambridge University Press, 2010).
5. Dechter, R. – *Constraint Processing* (Morgan Kaufmann, 2003).

Riferimenti su machine learning e modelli predittivi

6. Breiman, L. – *Random Forests* (Machine Learning Journal, 2001).
7. Cortes, C., Vapnik, V. – *Support Vector Machines* (Machine Learning Journal, 1995).
8. Chen, T., Guestrin, C. – *XGBoost: A Scalable Tree Boosting System* (KDD, 2016).

Riferimenti su ontologie e web semantico

9. Antoniou, G., van Harmelen, F. – *A Semantic Web Primer* (MIT Press, 2008).
10. Hitzler, P. et al. – *Foundations of Semantic Web Technologies* (CRC Press, 2010).

Riferimenti normativi e ambientali

11. Direttiva 2008/50/CE del Parlamento Europeo sulla qualità dell'aria.
12. Linee guida dell'**Organizzazione Mondiale della Sanità (OMS)** per la qualità dell'aria (2021).
13. Agenzia Europea dell'Ambiente – rapporti annuali sulla qualità dell'aria.
14. ARPA Puglia – dati e bollettini sulla qualità dell'aria regionale.