STUDIO SULLA QUALITÀ DELL'ARIA NEL 2023

COMPONENTI DEL GRUPPO:

Iosca Federica [Mat. 697611]: f.iosca2@studenti.uniba.it

A.A. 2023-24

Github: https://github.com/FeJ01/Prog_23-24.git

Sommario

Introduzione

L'inquinamento ambientale, negli ultimi anni, rappresenta uno dei problemi più importanti a livello globale. Per inquinamento ambientale si intende la presenza di agenti inquinanti nell'aria che sono dannosi per gli esseri viventi sul pianeta Terra.

Esistono diversi tipi di inquinamento, tra cui inquinamento termico, acustico, atmosferico, idrico ecc…; ciascuno dei quali porta con sé una catena di effetti collaterali di diversa entità, tra cui problemi respiratori.

L'obbiettivo di questo progetto è quello di analizzare e comprendere come gli agenti inquinanti presenti nell'aria, influiscano sulla qualità dell'aria. Inoltre, capire l'interazione tra la qualità dell'aria ed i fattori climatici.

Lo sviluppo di questo progetto è stato strutturato nelle seguenti fasi:

- Fase esplorativa del dataset che ha l'obiettivo di individuare eventuali valori nulli o dati poco compatti;
- Studio delle feature per mettere in evidenza quelle che impattano maggiormente con l'indice della qualità dell'aria;
- Realizzazione della Knowledge Base (scritta in Prolog) e definizione dei fatti e delle regole da cui fare una 'Feature Extraction' per ampliare lo studio;
- Fase di addestramento di modelli di apprendimento supervisionato e non supervisionato e visualizzazione delle loro prestazioni;
- Fase di addestramento di un modello probabilistico e valutazione della sua capacità di inferenza.

Requisiti fondamentali

Il progetto è stato realizzato in Python, in quanto offre un vasto numero di librerie che permettono di trattare e manipolare i dati in modo semplice. L'IDE utilizzato è PyScripter.

Librerie utilizzate

Le seguenti librerie possono essere installate sia una per volta che scrivendo nel prompt dei comandi "pip install -r requirements.txt".

- Pandas
- Numpy
- Pgmpy
- Sklearn
- Matplotlib
- Seaborn

- Imblearn
- pyswip
- Torch
- pickeDB
- networkx

Dataset

Il dataset, acquisito dalla piattaforma Kaggle, contiene informazioni ambientali e climatici riguardo zone geografiche ben definite, nel corso dell'anno 2023. Il dataset è "**global_air.csv**"

Il dataset contiene le seguenti informazioni:

- **City**: Sono riportate le città in cui sono state fatte le misurazioni. Questa feature contiene le 20 più grandi e principali città nel mondo, come ad esempio New York, Parigi, Tokyo, ecc…
- **Country**: Contiene i paesi di appartenenza delle città. Ogni città appartiene ad un paese differente.
- **Date**: Indica le date in cui sono state effettuate le misurazioni. Queste sono state effettuate lungo tutto l'anno 2023.
- **PM2.5**: Riporta le misurazioni medie nell'arco della giornata di particolato fine. Il particolato fine (Particulate Matter PM) è costituito da particelle solide e liquide aventi diametro aerodinamico variabile fra 0,1 e circa 100 μm che tendono a rimanere sospese in aria. Il termine PM2.5 è relativo alle particelle con diametro aerodinamico inferiore o uguale ai 2.5 μm.
- **PM10**: La definizione di questo elemento è simile alla precedente, ma con la differenza che questa feature misura la quantità media giornaliera di particolato fine delle particelle di diametro aerodinamico inferiore o uguale ai 10 μm (1 μm = 1 millesimo di millimetro).
- **NO2**: Mostra la misurazione giornaliera media del biossido di azoto. Si forma in gran parte in atmosfera per ossidazione del monossido (NO), inquinante principale che si forma nei processi di combustione. Le emissioni da fonti antropiche derivano sia da processi di combustione che da processi produttivi senza combustione (produzione di acido nitrico, fertilizzanti azotati, ecc.)
- **SO2**: Riporta la quantità media di biossido di zolfo. Questo si forma nel processo di combustione per ossidazione dello zolfo presente nei combustibili solidi e liquidi. Le fonti di emissione principali sono legate alla produzione di energia, agli impianti termici, ai processi industriali e al traffico. L'SO2 è il principale responsabile delle "piogge acide".

- **CO**: Misura la quantità media giornaliera di monossido di carbonio, un gas inodore e incolore che si forma dalla combustione incompleta degli idrocarburi presenti in carburanti e combustibili. Le concentrazioni in aria di questo inquinante possono essere ben correlate all'intensità del traffico.
- **O3**: Misura la quantità di ozono mediamente presente nell'aria. L'ozono è un gas incolore ed inodore e la sua presenza al livello del suolo dipende fortemente dalle condizioni meteoclimatiche.
- **Temperature**: Riporta la temperatura media calcolata durante la giornata in gradi Celsius (C°).
- **Humidity**: Indica il livello di umidità medio giornaliero misurato in percentuale.
- **Wind Speed**: Indica la velocità media del vento misurata in metri al secondo (m/s).

Calcolo sulla qualità della aria (AQI)

Una volta stabiliti gli obbiettivi iniziali dello studio sono state aggiunte due colonne necessarie, ovvero la colonna per l'AQI (chiamata Air_Quality) e la colonna contenente la categoria dell'indice precedente (chiamata Air_Quality_Category).

Per il calcolo dell'AQI è stato selezionato il metodo standard, stabilito da IQAir.

Innanzitutto, si calcola il valore dell'indice di inquinamento per ciascun inquinante tramite la formula standard:

$$AQI = \frac{I_{hi} - I_{lo}}{C_{hi} - C_{lo}} \times (C - C_{lo}) + I_{lo}$$

Dove:

- **C**: è la concentrazione misurata dell'inquinante.
- $\mathbf{C_{lo}}$: è il limite inferiore dell'intervallo di concentrazione in cui rientra la concentrazione **C**.
- $\mathbf{C_{hi}}$: è il limite superiore dell'intervallo di concentrazione in cui rientra la concentrazione **C**.
- $\mathbf{I_{lo}}$: è il valore di AQI corrispondente a $\mathbf{C_{lo}}$
- $\mathbf{I_{hi}}$: è il valore di AQI corrispondente a $\mathbf{C_{hi}}$

I limiti sono specifici per sostanza.

L'AQI complessivo si ottiene prendendo l'AQI più alto fra tutti gli inquinanti.

Una volta effettuati i calcoli per ogni riga presente nel dataset, sono passata alla classificazione di ciascun valore di AQI riportato, seguendo la tabella fornita da *IQAir* che determina, per ciascuna categoria, quali sono i rischi per salute umana.

Il livello di preoccupazione per la salute della qualità dell'aria è considerato **buono** se è compreso tra 0-50, **moderato** se è compreso tra 51-100, **malsano per gruppi sensibili** se è compreso tra 101-150, **malsano** se è compreso tra 151-200**, molto malsano** se è compreso tra 201-300 ed infine **pericoloso** se supera i 300.

Questo calcolo viene effettuato nel file '**Calcolo_Aqi.py'** e i risultati sono memorizzati nel nuovo file csv "**global_air01.csv**".

Estrazione di nuove feature

Nel file "**Studio_Feature.py"** sono state effettuate delle modifiche mirate per l'estrazione di nuove feature nel dataset. Con l'obiettivo di semplificare i calcoli futuri, la feature **Date** viene scomposta nelle tre feature **Day, Month** e **Year**.

Una volta effettuata questa divisione è stato possibile calcolare con più semplicità la media della velocità del vento mensile e la media della temperatura mensile per città, rappresentate dalle feature **Monthly_Avg_Temperature** e **Monthly_Avg_Wind_Speed.** Queste saranno fondamentali per la determinazione delle feature ambientali **HasRained** e **Is_Stagnant**, che indicheranno se nel giorno delle misurazioni ha piovuto oppure se l'aria fosse "stagnante". Queste nuove feature verranno trattate nel paragrafo relativo alla Knowledge base.

Queste modifiche del dataset sono salvate in un nuovo file csv intermedio ("**global_air02.csv**").

Fase esplorativa del dataset

Dopo aver inserito le due nuove colonne **Air_Quality** e **Air_Quality_Category** è stato esplorato il dataset con lo scopo di individuare feature rilevanti ed eventuali informazioni mancanti.

In questa fase sono utilizzati dei grafici ottenuti tramite funzioni delle librerie *matplotlib e seaborn*, visualizzabili nel file **"Studio_Feature.py"**.

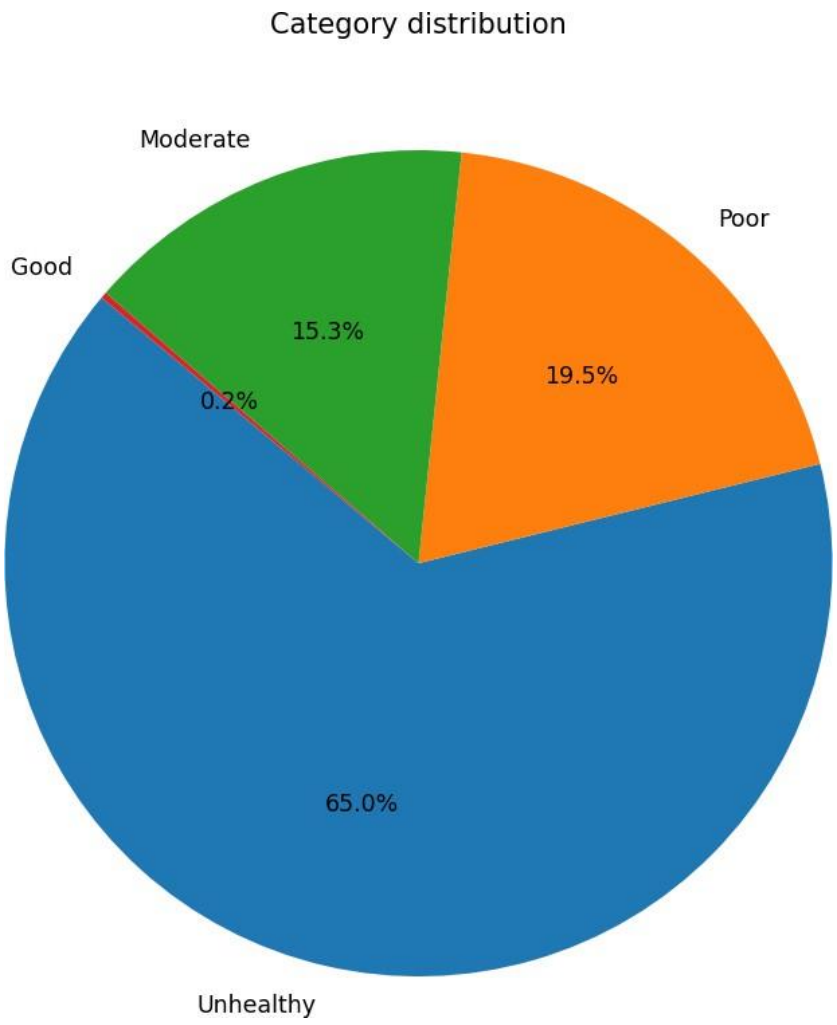
La matrice di correlazione ha permesso di individuare le feature che impattano maggiormente sull'AQI: **PM2.5** e **PM10**.

Le altre feature presentavano una scarsa o nulla correlazione con la qualità dell'aria.

Successivamente si è osservata la distribuzione degli inquinanti principali, individuate poco fa ed è emerso un andamento altalenante nel tempo.

Esplorando la distribuzione dell'Air_Quality_Category all'interno del dataset è emerso uno sbilanciamento delle classi. Come si può vedere in figura, il 65% dei dati è stato classificato come 'Unhealthy' e solo lo 0.2% dei dati è stato classificato come 'Good'.

Questo sbilanciamento rappresenta un problema per quanto riguarda l'addestramento dei modelli di ML. Per ovviare questo problema, verranno utilizzati dei metodi di bilanciamento del dataset, che saranno analizzati successivamente.



Knowledge Base e Prolog

La Knowledge Base è un sistema di gestione delle conoscenze che consente di creare, utilizzare e gestire le informazioni riguardanti un dominio di interesse. Per la realizzazione della Knowledge Base è stato utilizzato **Prolog** e la libreria *pyswip* per consentirne la gestione con il linguaggio di programmazione Python.

Questo progetto si concentra sul monitoraggio e sull'analisi dei dati ambientali, con un'attenzione particolare alla qualità dell'aria e alle condizioni meteorologiche in varie città del mondo.

L’obiettivo della knowledge base è estrarre nuove caratteristiche dalle relazioni tra le misurazioni, al fine di ottenere informazioni significative e utili.

I parametri utilizzati per la creazione della knowledge base sono in linea con il dataset sopra citato:

- City
- Country
- Date
- PM2.5
- PM10
- NO2
- SO2
- CO
- O3
- Temperature
- Humidity
- Wind Speed

Per unire i diversi concetti, sono stati configurati i seguenti predicati:

- Predicato che mette in relazione la città e il paese in cui si trova:
  geography_facts(City,Country)

- Predicato che mette in relazione le misurazioni degli inquinanti in un dato mese e giorno in una data città:
  pollutants_facts(City, Month, Day, PM2.5, PM10, NO2, SO2, O3, CO)

- Predicato che mette in relazione i fattori climatici misurati un dato mese e giorno in una data Città:
  climate_facts=(City, Month, Day, Temperature, Humidity, Wind Speed)

- Predicato che mette in relazione l’AQI numerico e l’AQI categorico per una data Città in un dato mese e giorno:
  aqi_facts(City, Month, Day, Air_Quality, Air_Quality_Category)

- Predicato che mette in relazione le medie mensili di temperatura e velocità del vento misurate in una data città:
  monthly_averages_facts(City, Month, Monthly_Avg_Temperature, Monthly_Avg_Wind_Speed)

Sono state definite delle **regole di base** e delle **regole più avanzate** con lo scopo di estrarre nuove informazioni sulle condizioni climatiche:

- Regola di base che rappresenta le misurazioni degli inquinanti, condizioni climatiche e AQI per una data città in un dato giorno e mese

```
get_measurements(City, Month, Day, PM2_5, PM10, NO2, SO2, O3, CO, Temperature, Humidity, Wind_Speed, AQI_value, AQI_category) :
        pollutants(City, Month, Day, PM2_5, PM10, NO2, SO2, O3, CO),
        climate_factors(City, Month, Day, Temperature, Humidity, Wind_Speed),
        aqi(City, Month, Day, AQI_value, AQI_category).
```

- Regola per l'ottenimento del valore massimo di AQI in assoluto all'interno del dataset

```
highest_aqi(City, Month, Day, Max_AQI) :
        findall(AQI_value, aqi(_, _, _, AQI_value, _), AQI_list),
        max_list(AQI_list, Max_AQI),
        aqi(City, Month, Day, Max_AQI, _).
```

- Regola che determina se in un dato mese e un giorno ha piovuto in una data città.

```
rainy_day(City, Month, Day,HasRained):
        climate_factors(City, Month, Day, Temperature, Humidity, Wind_Speed),
        monthly_averages(City, Month, Average_Temperature,Average_Wind_Speed),
        (   Temperature < Average_Temperature,
        Wind_Speed > Average_Wind_Speed,
```

```
    Humidity > 90
->  HasRained = true
;   HasRained = false
).
```

- Regola che determina se in dato mese e giorno l'aria era stagnante in una data città.

```
stagnant_air(City, Month, Day,Is_Stagnant):
        climate_factors(City, Month, Day, Temperature, Humidity,
        Wind_Speed),
         (   Temperature > 28,
         Wind_Speed < 3.5,
         Humidity > 70
->  Is_Stagnant = true
;   Is_Stagnant = false
).
```

- Regola che determina l'indice di dispersione degli inquinanti presenti nell'aria, calcolato in un dato mese e giorno per una Città.

```
pollutant_dispersion(City, Month, Day, Dispersion_Index) :-
        climate_factors(City, Month, Day, _, Humidity, Wind_Speed),
        Dispersion_Index is Wind_Speed * (1 / (1 + (Humidity / 100))).
```

Queste regole sono state utilizzate per definire nuove Features e ampliare il campo di studio, ponendo attenzione ai fattori climatici.

Infatti, è stato realizzato un nuovo dataset "**global_air_fin.csv**" che integra le feature estratte mediante Query in Prolog poste alla knowledge base.

Le nuove features introdotte sono:

- HasRained: campo booleano che, basandosi sulle misurazioni climatiche con una velocità del vento sopra la media mensile, la temperatura sotto la media mensile e un'umidità molto alta (al di sopra del 90%) determina se le misurazioni sono state effettuate in un giorno di pioggia.
- Is_Stagnant: campo booleano che determina, come una giornata con aria stagnante, sia una giornata che un'umidità e una temperatura molto alte

accentuate da un vento poco mosso, determinato dalla scala di Beaufort come brezza legger)
- Dispersion_Index: campo reale che indica l'indice di dispersione delle particelle inquinanti, basandosi sul rapporto dell'umidità e della velocità del vento, calcolati in una data città.

Predizione dell'AQI

In questa sezione verranno analizzate le performance dei vari modelli addestrati per la previsione dell'AQI, utilizzando sia algoritmi di apprendimento supervisionato che non supervisionato.

Random forest con misurazioni degli inquinanti

Il Random Forest è un algoritmo di apprendimento supervisionato che si basa su alberi decisionali. Si tratta di un modello ensemble, poiché unisce le previsioni di diversi alberi decisionali per ottenere un risultato più preciso e resistente.

Il Random Forest crea numerosi alberi decisionali, ciascuno dei quali viene addestrato su un sottoinsieme casuale dei dati di addestramento. Dopo l'addestramento, ogni albero fornisce una previsione. Nel caso della classificazione, il modello finale seleziona la classe che ha ricevuto il maggior numero di voti dagli alberi. Per la regressione, invece, il risultato finale viene ottenuto calcolando la media delle predizioni degli alberi. L'obiettivo è quello di ottenere previsioni per l'AQI categorico basate sulle misurazioni degli inquinanti atmosferici.

In questo progetto la costruzione del random forest richiede una serie di passaggi.

Come prima cosa, viene caricato il dataset contenente le misurazioni della qualità dell'aria.

Nella fase di pre-processing sono state rimosse alcune colonne non rilevanti per lo studio (tra cui Unnamed: 0, Year e City).

Per la gestione della variabile categoriale Country, è stato utilizzato "get_dummies" che esegue la codifica one-hot trasformando variabili categoriche in variabili numeriche binarie.

Per selezionare le feature e ridurre la dimensionalità del problema, sono stati esaminati la matrice di correlazione e i risultati della feature 'importance' del modello. Sono stati effettuati diversi tentativi, arrivando a determinare che 4 fosse il numero ottimale di feature da utilizzare per migliorare le performance del modello.

Successivamente, durante la fase di addestramento sono stati effettuati due tentativi. Il primo tentativo ha previsto l'addestramento del modello senza applicare alcun

bilanciamento del dataset. Questo approccio si è rivelato inefficace, poiché la curva di apprendimento mostrava evidenti segni di overfitting e la classe "Good" veniva esclusa dalla classificazione, a causa del numero limitato di campioni presenti nel test set.

I risultati ottenuti per il primo tentativo erano i seguenti:

Accuratezza del training: 1.0000

Accuratezza del test: 0.9980

Report di classificazione:

	precision	recall	f1-score	support
good	0.00	0.00	0.00	4
moderate	0.99	1.00	0.99	452
poor	1.00	1.00	1.00	564
unhealthy	1.00	1.00	1.00	1980

accuracy			1.00	3000
macro avg	0.75	0.75	0.75	3000
weighted avg	1.00	1.00	1.00	3000

Accuratezza media della cross-validation: 0.9972

Deviazione standard della cross-validation: 0.0004

L’accuratezza perfetta sul training set potrebbe indicare che il modello ha memorizzato i dati di training e potrebbe non essere in grado di generalizzare su nuovi dati.

L’approccio finale consiste nel bilanciamento del dataset per la classe ‘Good’ utilizzando il metodo **SMOTE** della libreria *imblearn*. Il numero di istanze della classe è stato aumentato fino a raggiungere la media delle altre classi, migliorando così le prestazioni del modello.

Dopo aver bilanciato il dataset, sono stati fatti alcuni tentativi per cercare di ridurre la complessità degli alberi così da evitare il rischio di overfitting. Infine, il modello è stato addestrato utilizzando 30 alberi, una profondità massima di 8 e un campionamento pari all’80% del dataset. Subito dopo, il dataset è stato suddiviso in training set e test set per valutarne le prestazioni.

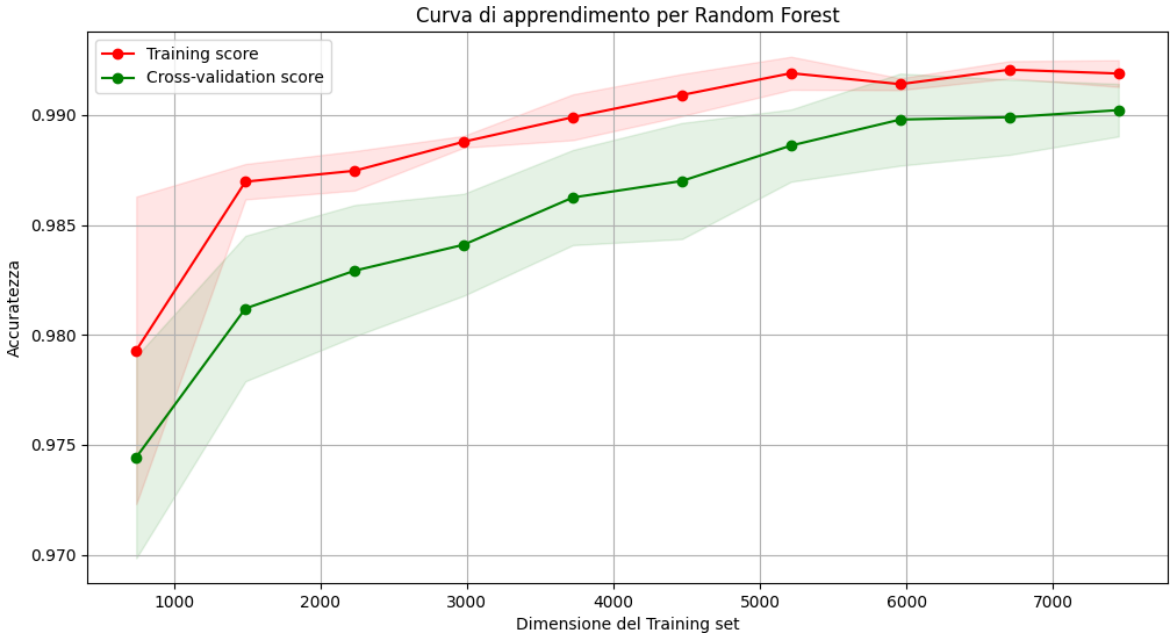
Di seguito lo screen dove si espongono i risultati ottenuti nella fase di valutazione:

```
***Metriche per il modello Random Forest:***
Accuratezza del training:  0.9935553168635876
Accuratezza del test:  0.9897243107769423
Precisione:  0.9897388338729615
Recall:  0.9897243107769423
F1-score:  0.9896668945089998
Matrice di confusione:
 [[ 980   11    0    0]
 [  28  420    1    0]
 [   1    0  581    0]
 [   0    0    0 1968]]
Report di classificazione:
              precision    recall  f1-score   support

        Good       0.97      0.99      0.98       991
    Moderate       0.97      0.94      0.95       449
        Poor       1.00      1.00      1.00       582
   Unhealthy       1.00      1.00      1.00      1968

    accuracy                           0.99      3990
   macro avg       0.99      0.98      0.98      3990
weighted avg       0.99      0.99      0.99      3990
```

Il report di classificazione conferma le ottime prestazioni del modello per tutte le classi, con un'accuratezza media molto elevata.




```
Random Forest Cross-Validation Accuracy: 0.9902 ± 0.0012
```

La curva di apprendimento del modello esprime l'andamento dell'accuratezza al crescere della dimensione del set di traning.

Il punteggio di accuratezza sul set di traning aumenta in modo rapido fino a 3000 campioni, dopodiché si stabilizza intorno al 99%.

La curva dell'accuratezza sui dati di cross-validation mostra un trend crescente, simile a quello del training, ma è sempre inferiore ,anche se leggermente, evidenziando un divario tra prestazioni sui dati di training e generalizzazione su nuovi dati. In ogni caso, questo divario si riduce all'aumentare dei campioni.

Per concludere, possiamo dire che il modello addestrato con bilanciamento dei dati ha ottime prestazioni e rispetto al modello precedente, presenta un ottimo equilibrio tra stabilità e generalizzazione su nuovi dati, minimizzando così il rischio di overfitting.

Regressione lineare

Il regressore lineare è un modello di machine learning utilizzato per prevedere una variabile continua in base a una o più feature. La regressione lineare cerca di trovare una relazione lineare tra le feature (variabili indipendenti) e la variabile target (variabile dipendente). La funzione lineare è rappresentata come:

$$y = \beta 0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

Dove:

- y è la variabile target
- $\beta 0$ è l'intercetta
- β_i (i = 1,2,…n) sono i coefficienti delle feature
- x_i sono le feature
- ε è l'errore

In questo contesto il regressore lineare è stato utilizzato per prevedere l'AQI sulla base delle feature 'Dispersion_Index', 'HasRained', 'Is_Stagnant'.

La selezione di queste feature è motivata dal desiderio di scoprire le relazioni tra i fattori climatici e l'indice di qualità dell'aria. Ad esempio, un elevato indice di dispersione dell'aria potrebbe favorire la riduzione degli inquinanti, portando così a un miglioramento della qualità dell'aria.

Si è partiti con l'analisi della matrice di correlazione tra le feature selezionate e Air_Quality. L'analisi ha evidenziato una bassa correlazione, persino negativa per la feature 'HasRained'.

Nella fase di pre-processing sono state selezionate le feature di input sopra citate e il target, Air_Quality. Successivamente è stata applicata una normalizzazione delle caratteristiche mediante StandardScaler della libreria *sklearn.*

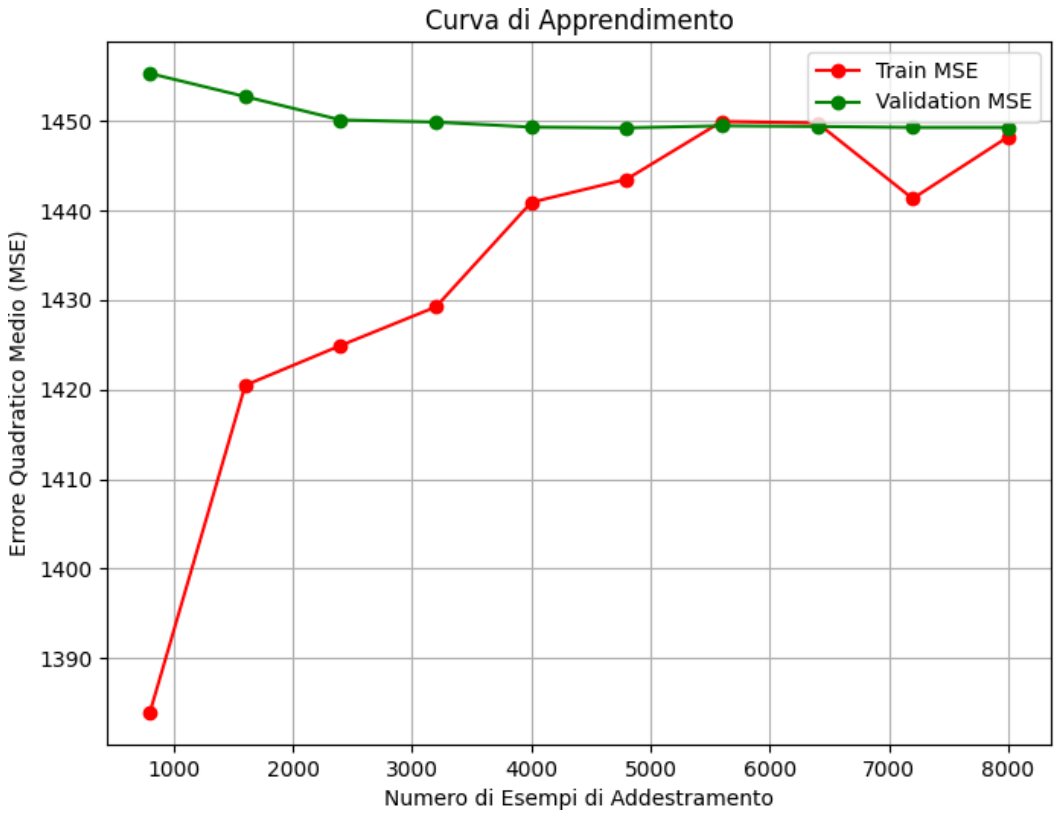
Dopo l'addestramento, il modello di regressione lineare viene valutato con cross-validation e sui dati di test.

I risultati ottenuti sono i seguenti:

```
***Linear Regression***
Mean Squared Error (CV): 1449.31
Mean Squared Error (Test): 1400.73
R-squared (Test): -0.00
```

Dai risultati emerge che il modello non riesce a identificare una relazione significativa tra le variabili climatiche selezionate e l'AQI. Questo è confermato dal valore di R-squared, nullo, che indica che il modello non è in grado di spiegare adeguatamente la variabilità della qualità dell'aria.

La curva di apprendimento per il modello indica che l'errore di addestramento (curva rossa) cresce con l'aumentare dei dati segnalando un adattamento limitato ai dati.



L’errore di valutazione, invece, resta costante intorno a 1450, suggerendo che il modello non migliora all’aumentare della dimensione del dataset.

Il fatto che le curve non convergano e che il validation MSE non scenda indica che il modello è troppo semplice per catturare i pattern complessi nei dati.

K-Nearest Neighbor Regression

Dati i risultati non ottimali ottenuti con il Regressore Lineare, ho scelto di provare l’addestramento del modello **K-Nearest Neighbor Regressor** (**KNN**), capace di individuare relazioni non lineari tra le variabili.

Il KNN è un modello di apprendimento supervisionato e non parametrico utilizzato per problemi di classificazione e regressione. L’obiettivo del KNN è fare previsioni per un nuovo esempio osservando i *K* punti più vicini in termini di distanza (ad esempio distanza euclidea).

Per questo progetto, le feature di input restano ‘Dispersion_Index’, ‘HasRained’, ‘Is_Stagnant’ e la feature di target è ‘Air_Quality’.

Dopo svariati tentativi, ho deciso di utilizzare la ricerca a griglia (GridSearchCV). Questo metodo è uno strumento di ricerca di iperparametri con l’obiettivo di individuare la combinazione migliore di iperparametri per un modello. Questo metodo ha permesso così di individuare i migliori parametri per il modello utilizzato:

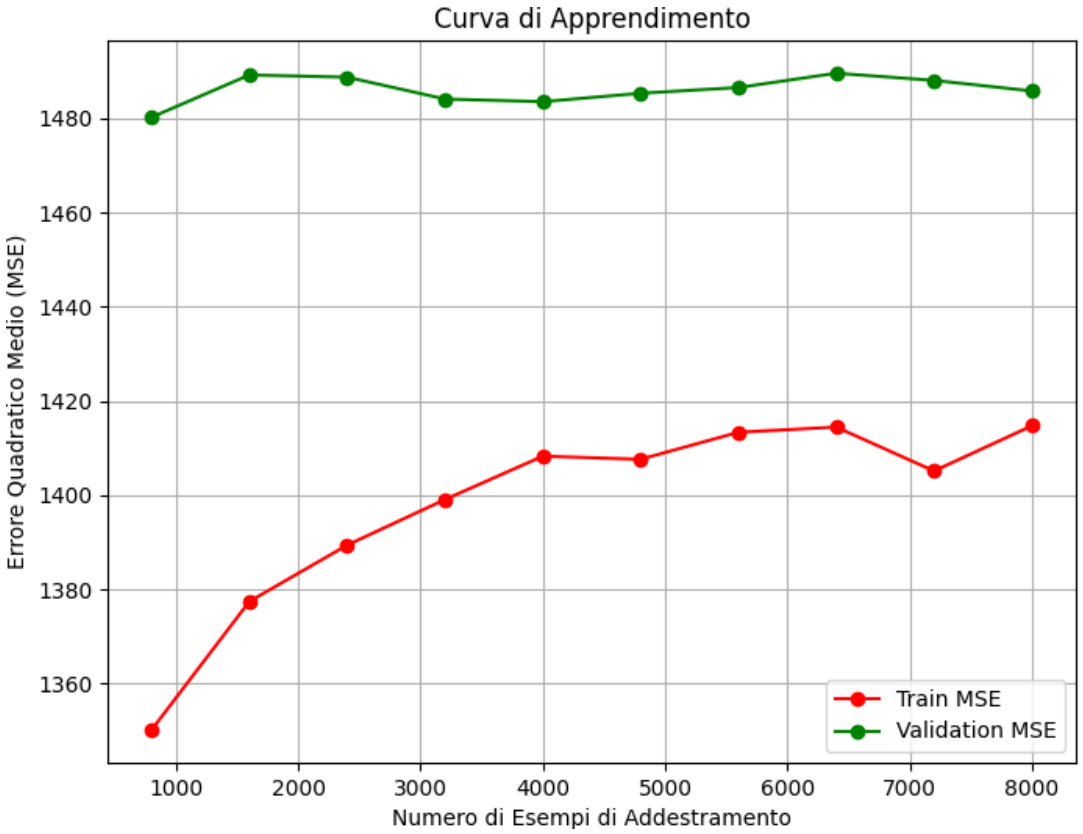
```
**K-neighbors Regression
Migliori parametri trovati: {'metric': 'euclidean', 'n_neighbors': 40, 'p': 1, 'weights': 'uniform'}
```

Dove:

- **Metric: Euclidian** -> definisce il tipo di metrica per calcolare la distanza tra i punti. In questo caso viene utilizzata la distanza euclidea. La distanza euclidea tra due punti $A(x_1\ y_1)$ e $B(x_2, y_2)$ è data dalla formula: $d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
- **n_neighbors: 40** -> indica il numero di vicini da considerare per fare una previsione. In questo caso il numero è 40.
- **P: 1** ->indica l’ordine della norma di Minkowski da usare. Tuttavia, in questo caso, dato che la metrica è ‘euclidian’, il parametro p viene ignorato.
- **Weights: uniform ->** specifica come vengono pesati i vicini nella previsione. Con ‘uniform’, tutti i vicini hanno lo stesso peso, indipendentemente dalla loro distanza dal punto di interesse.

Subito dopo l’ottimizzazione degli iper parametri, si passa all’addestramento del modello e alla sua valutazione.

La curva di apprendimento del modello è la seguente:



Da questo grafico è possibile notare che l'errore quadratico medio non diminuisce con l'aumentare degli esempi di training, stabilizzandosi intorno a 1400. Questo indica che il modello non sta imparando una rappresentazione migliore dei dati man mano che si aggiungono più esempi.
Inoltre l'errore di validazione, è più elevato rispetto all'errore di addestramento per tutta la curva. Questo conferma la difficoltà del modello di generalizzare. Infatti, la curva è piatta, indica che l'aggiunta di dati non migliora le prestazioni di generalizzazione del modello.

I risultati ottenuti sono i seguenti:

```
Mean squared error(CV): 1485.83
Mean Squared Error (MSE): 1430.61
R-squared: -0.02
```

Il valore dell'errore quadratico medio ottenuto attraverso la cross-validation indica che il modello fa errori significativi sulla stima dei valori target.
Il valore dell'errore quadratico medio mostra che il modello ha difficoltà a generalizzare i dati di test e che potrebbe essere vicino ad un caso di underfitting.
Inoltre, R-squared ha un valore negativo che indica che il modello non riesce a individuare il pattern sottostante dei dati.

In sintesi, si può affermare che il modello non è adatto per questo problema specifico, poiché non riesce a catturare né la complessità dei dati di training né quella dei dati di test.

K-means

Il **K-means** è un algoritmo che ricade nella categoria degli **algoritmi di apprendimento non supervisionato**. Questo è un tipo di apprendimento che prevede un addestramento del modello senza l'utilizzo di dati etichettati. Viene utilizzato prevalentemente per la ricerca di relazioni intrinseche tra i dati di training non facilmente denotabili senza uno studio approfondito. L'apprendimento non supervisionato si divide in due principali tipologie:

- Algoritmi di clustering: i dati di input vengono raggruppati in cluster differenti in base alla similarità delle loro caratteristiche. Questi cluster vengono formati in base alla distanza interna tra dati. Si dividono in due categorie: **hard clustering** e **soft clustering**.
- Algoritmi di regressione: sono dei tipi di algoritmi utilizzati principalmente nella fase di preelaborazione delle features, con l'biettivo di eliminare il "rumore" dei dati. La riduzione della dimensionalità può essere utile anche per la rappresentazione dei dati.

Nello specifico il K-means ricade nella categoria degli algoritmi di clustering, nello specifico nella categoria degli algoritmi di hard clustering. Questo assegna ogni dato del dataset ad uno ed un solo cluster, al contrario gli algoritmi di soft clustering, invece, forniscono un assegnamento probabilistico di un elemento ad ogni cluster.

In questo caso di studio l'addestramento non supervisionato viene utilizzato con lo scopo di clusterizzare le misurazioni ambientali e cercare una corrispondenza con le effettive etichette. Questo viene fatto mediante lo studio delle caratteristiche climatiche delle misurazioni.

È importante notare come questo algoritmo, basandosi sul calcolo delle distanze euclidee, necessiti di un bilanciamento dei dati in modo che tutte le caratteristiche forniscano il giusto contributo nel calcolo dei cluster. Infatti, prima dell'applicazione del k-means sono stati applicati due metodi di pre-processing dei dati: la gestione degli **outliers** e la **scalarizzazione** dei valori.

Con outliers si definiscono i valori, che per eccesso o difetto, si discostano significativamente dalla maggior parte dei valori del dataset. Una presenza eccessiva di questi valori può portare ad uno spostamento dei centroidi, portando ad una rappresentazione errata dei dati del dataset. Nella prima fase di pre-processing dei dati per il non supervisionato viene calcolato per ogni feature un intervallo

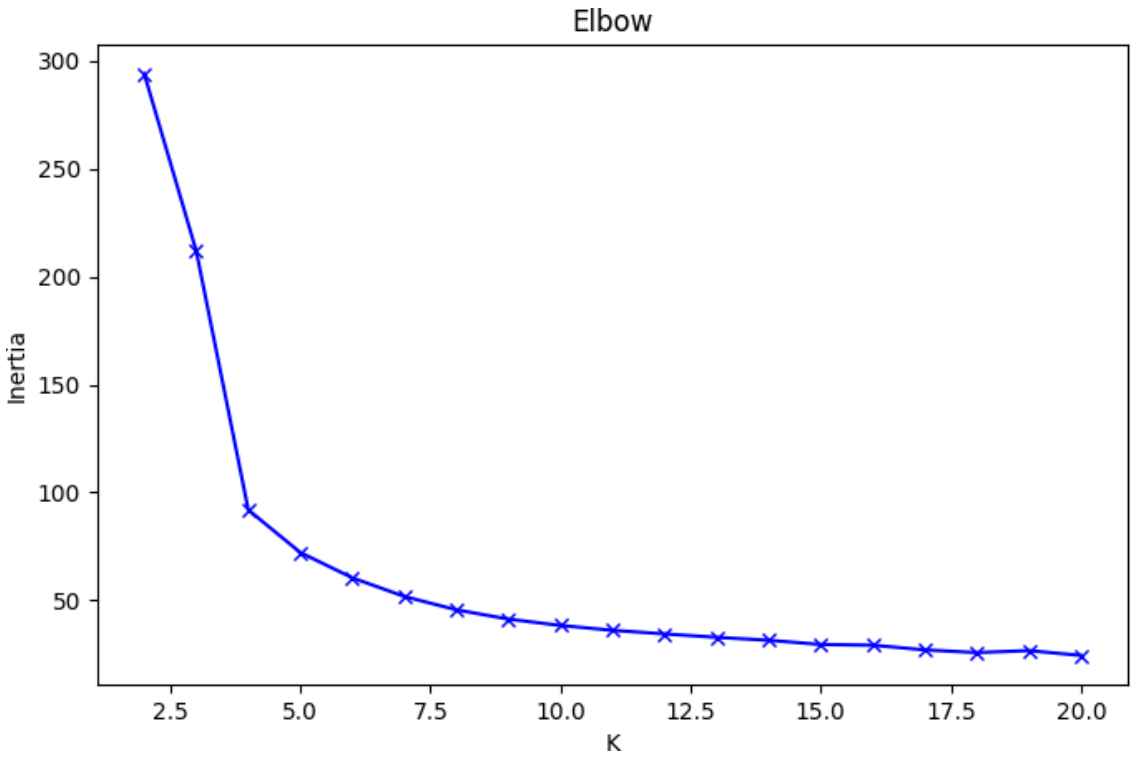
interquartile. Tutti i valori al di fuori dell'intervallo vengono considerati outliers e di conseguenza vengono eliminati.

La scalarizzazione, invece, serve a garantire che tutte le features abbiano lo stesso impatto nella determinazione delle distanze. Nel caso specifico di questo progetto la scalarizzazione dei dati è fondamentale in quanto il dataset riporta molte misurazioni con differenti unità di misura. Ci sono varie tipologie di scalarizzazione. In questo caso, quella selezionata è stata la scalarizzazione tramite normalizzazione, in quanto permette di ridimensionare i valori in un dataset in modo che rientrino in un intervallo comune che vada da 0 a 1.

Uno degli aspetti fondamentali per un buon rendimento dell'algoritmo di K-means è la selezione di un numero K ideale di centroidi. Per scegliere questo valore ottimale sono state utilizzate due strategie largamente utilizzate: il **metodo del gomito** (**Elbow method**) e il **Silhouette Score**.

Entrambi i risultati di questi due metodi sono interpretabili tramite uno studio dei grafici che producono.

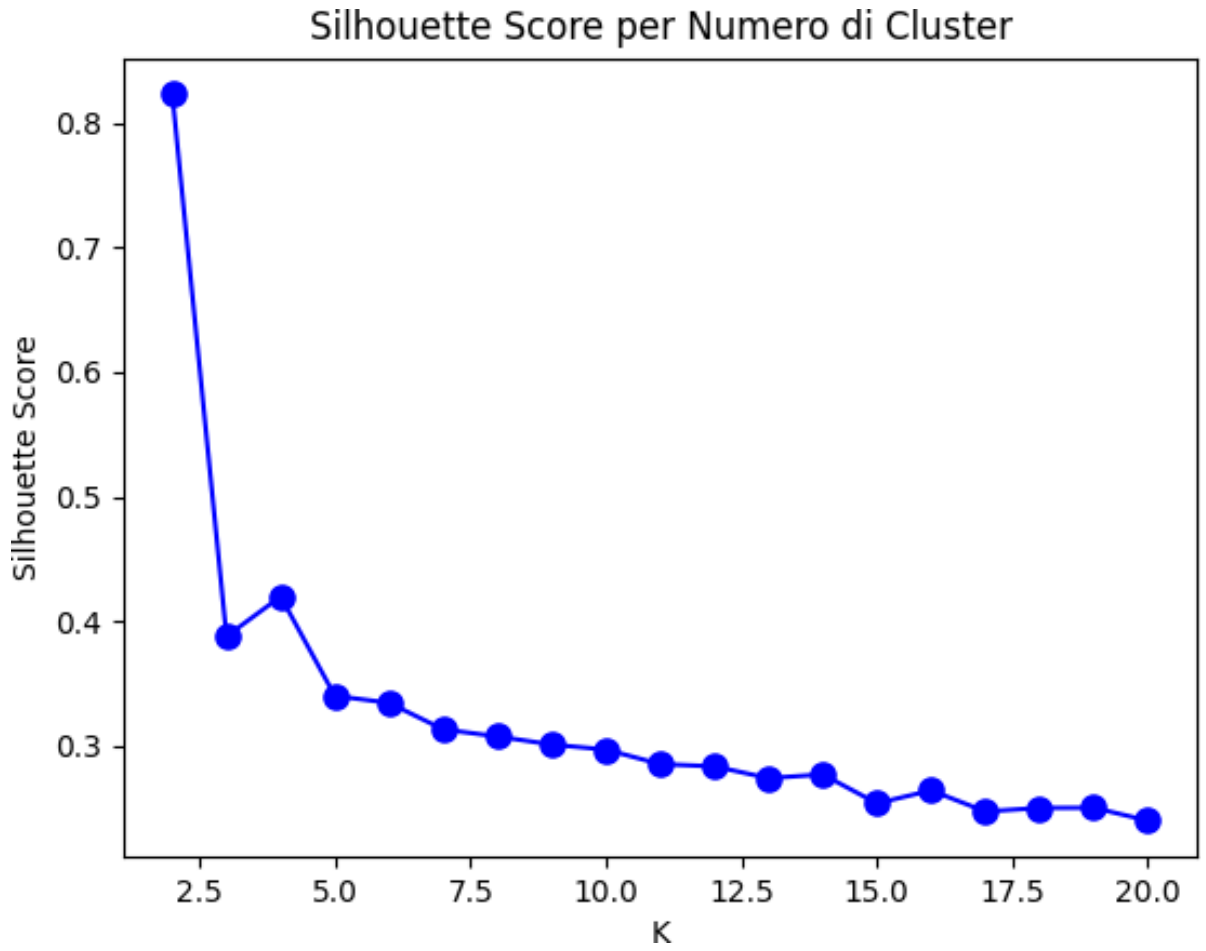
Il metodo del gomito mostra la variazione dell'inerzia (ovvero la somma delle distanze al quadrato tra ogni punto e il centro del cluster) al variare del numero di cluster. Queste misurazioni vengono fatte aumentando iterativamente il numero di cluster fino ad un valore massimo, che nel caso di questo progetto è 20. Una volta effettuati tutti i calcoli si costruisce un grafico che riporta i risultati ottenuti. Il "gomito" viene indicato come il punto del grafico in cui la riduzione dell'inerzia rallenta in modo significativo, ovvero dove la curva discendente si stabilizza mostrando un andamento più lineare.



Osservando questo grafico si riesce facilmente a comprendere come il numero k di centroidi sia 4, corrispondente al numero di etichette del dataset, il che risulta estremamente conveniente.

Per ottenere la conferma di questo risultato è stato applicato il metodo del **silhouette score.**

Il silhouette score calcola, con una scala che va da -1 a 1 la qualità dei cluster valutando contemporaneamente quanto bene i punti sono raggruppati all'interno dei loro cluster (coesione) e quanto sono separati dai punti di altri cluster (separazione), aiutando a determinare il numero ottimale di cluster.



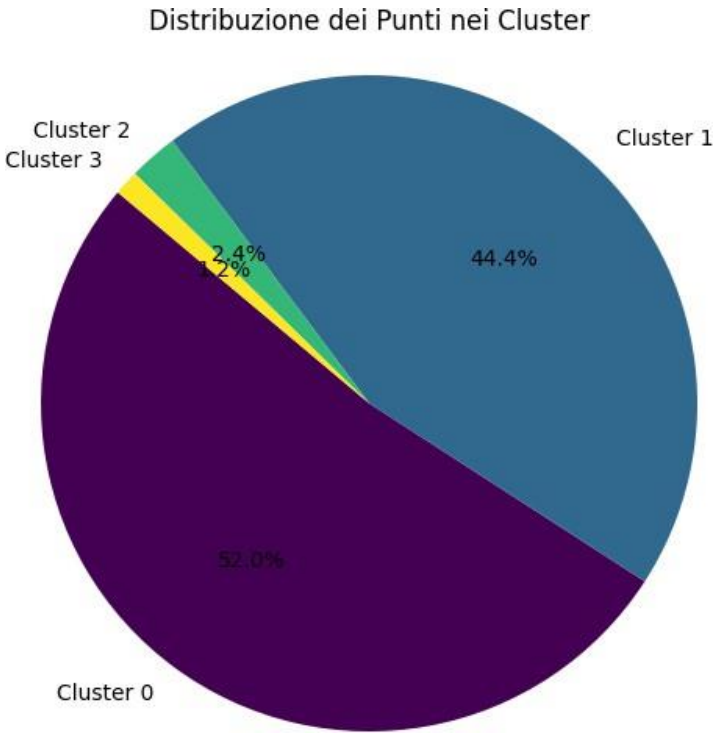
Pur osservando un silhouette score di circa 0.82 per k=2, questo non rappresenta un numero significativo del dataset data la formazione di cluster estremamente generici.

Questo lascia intendere che k=4 sia il valore migliore in quanto riporta come valore del silhuette score circa 0.42. Pur essendo uno score discreto viene ritenuto comunque sufficiente ai fini dello studio.

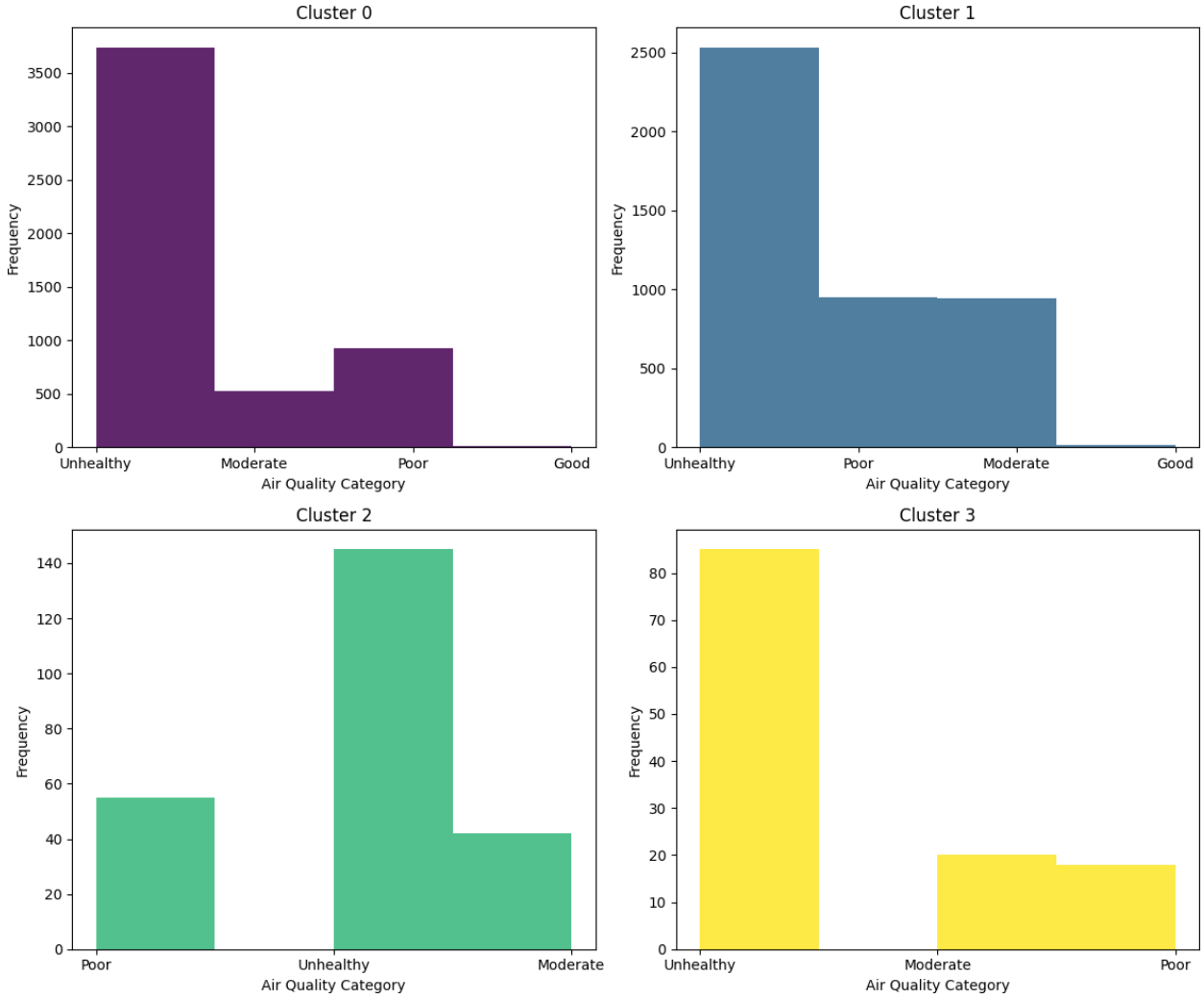
L'applicazione del K-means viene effettuata sul set di features rappresentanti le misurazioni metereologiche.

features= ['Temperature', 'Humidity', 'Wind Speed', 'HasRained', 'Is_Stagnant', 'Dispersion_Index']

Si ottiene il seguente risultato:



Inoltre, è stato riportato la distribuzione delle etichette delle misurazioni all’interno del clustering:



Per testare la precisione della corrispondenza tra cluster ed etichette è stato utilizzato la **V-measure**.

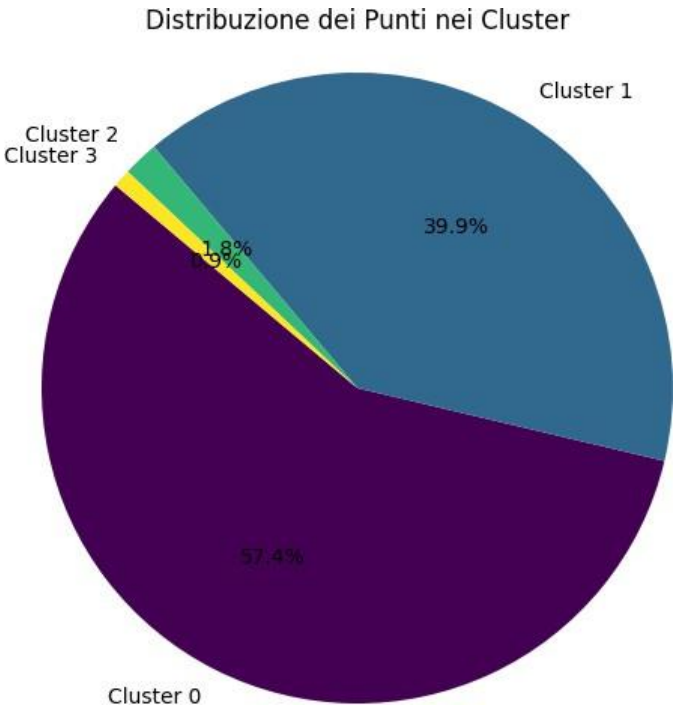
La **V-measure** è una metrica di valutazione utilizzata per misurare la qualità di un clustering, confrontando i cluster generati con le vere etichette di classificazione. La V-measure è la media armonica di omogeneità e completezza dei cluster e varia tra 0 e 1.

In questo caso il risultato ottenuto è stato:

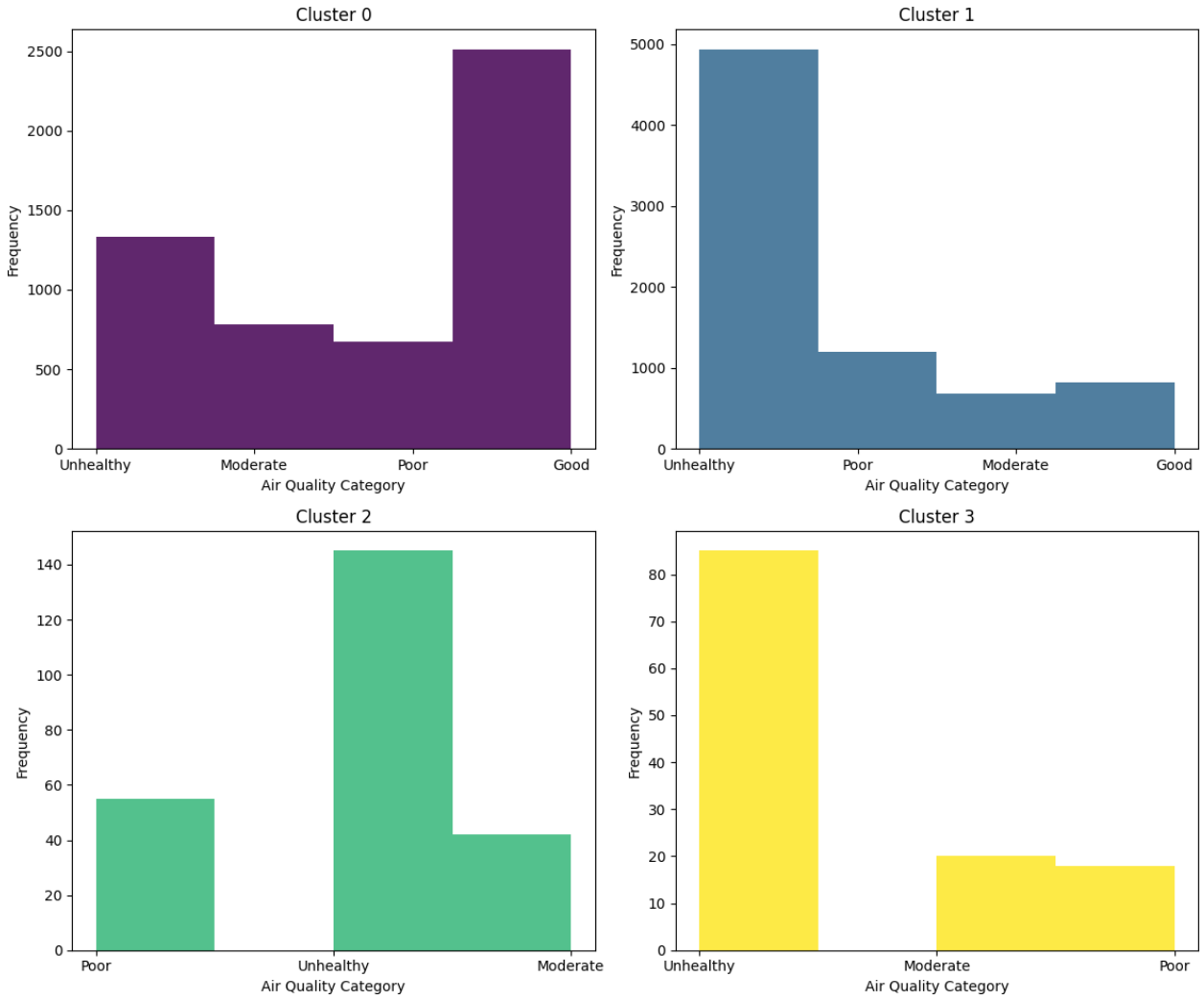
V-Measure: 0.017364694507662298

Si nota pertanto uno scarso risultato.

Nel tentativo di migliorare le prestazioni del clustering è stato deciso di utilizzare il dataset bilanciato tramite SMOTE, pur mantenendo lo stesso numero di cluster. I risultati ottenuti sono:



Con distribuzione delle etichette:



Ottenendo un valore di V-measure pari a

V-Measure: 0.11339110721678153

Il risultato è ancora molto basso, migliorando sufficientemente le prestazioni precedenti.

Infine, per migliorare il valore del V-measure, è stato necessario aggiungere alle feature climatiche la feature con maggiore peso per la classificazione delle misurazioni, ovvero il PM2.5.

features= [‘PM2.5’, ‘Temperature’, ’Humidity’, ’Wind Speed’, ’HasRained’, ’Is_Stagnant’, ’Dispersion_Index’]

Questo tentativo porta ad un aumento del valore di V-measure:

V-Measure: 0.437450081298452

Pur essendo un valore discreto, fornisce un piccolo miglioramento della capacità del sistema di creare cluster significativi.

La rete Bayesiana

Una **rete Bayesiana** è un modello probabilistico che rappresenta un insieme di variabili e le loro dipendenze condizionali tramite un grafo aciclico diretto, in cui ogni nodo del grafo rappresenta una variabile, mentre gli archi indicano le relazioni di causalità o dipendenza tra queste variabili. Le relazioni sono quantificate da distribuzioni di probabilità condizionate.

La formula utilizzata dalle reti bayesiane segue la stessa logica del teorema di Bayes. Ma applicata all'intera struttura di dipendenze rappresentata dalla rete:

$$P(X_1, X_2, \dots, X_n) = \prod_{i=1}^{n} P(X_i | \, genitori(X_i))$$

Dove: $X_1, X_2, \dots, X_n$ sono le variabili della rete, $P(X_i | \, genitori(X_i))$ rappresenta la probabilità congiunta e g$enitori(X_i)$ rappresenta le variabili da cui X_i dipende direttamente.

Per la costruzione delle reti Bayesiane, per prima cosa è necessario identificare le variabili e i loro rapporti interni. Le feature selezionate per la costruzione del modello comprendono tutte le misurazioni, sia quelle relative agli agenti inquinanti, sia quelle relative ai fattori climatici.

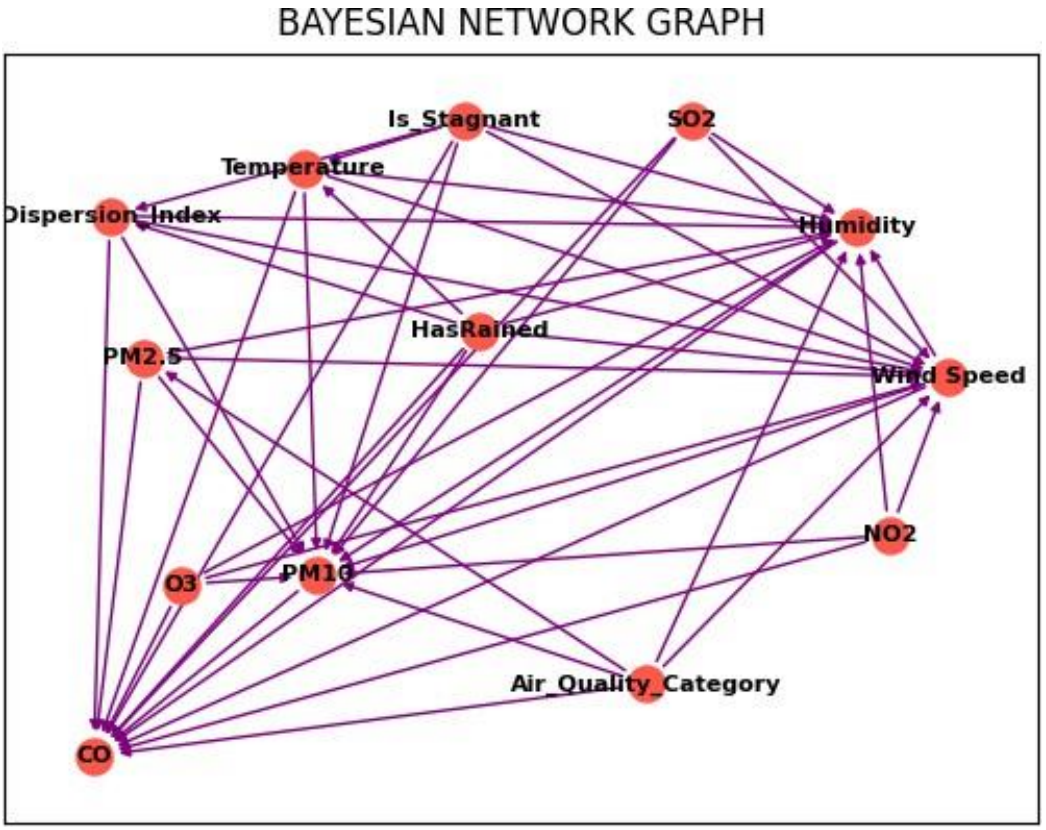
La feature 'Air_Quality' è stata esclusa ritenendo che la sua presenza diminuisse il grado di generalizzazione, in quanto eccessivamente preponderante per le previsioni del modello.

Prima di eseguire l'analisi della struttura del grafico è stato necessario effettuare una **discretizzazione** dei valori continui. Questo processo trasforma i dati continui in intervalli discreti, in modo da diminuire la complessità del modello e rendere possibile uno studio anche su un ampio spettro di valori. Come strategia di discretizzazione è stata usata la strategia quantile, che divide i valori in bin in base alla distribuzione.

I tentativi di costruzione della rete Bayesiana senza una precedente discretizzazione dei valori non sono stati portati a termine poichè l'eccessiva richiesta di memoria ha portato in alcuni casi a far crashare il sistema.

Quindi, una volta discretizzati i valori di interesse è stata utilizzata la tecnica di **Hill Climb Search** per l'apprendimento della struttura della rete. Questa tecnica utilizza una strategia di ottimizzazione locale effettuando svariati tentativi modificando la struttura della rete, cercando di migliorare il **K2 score** (il punteggio selezionato per il caso di studio).

È stato inoltre imposo un limite di 12 archi entranti per nodo in modo tale da evitare eccessivi carichi di memoria. La struttura che viene fuori è la seguente:



Inoltre, dallo studio del **CPD** (Conditional Probability Distribution) si può notare come al target feature Air_Quality_Category sia una variabile indipendente, ovvero il suo valore non dipende direttamente da altre variabili:

Air_Quality_Category (Good)	0.0025
Air_Quality_Category (Moderate)	0.0153
Air_Quality_Category (Poor)	0.1947
Air_Quality_Category (Unhealthy)	0.6498

Per verificare l'efficacia della rete Bayesiana è stata calcolata la categoria di appartenenza per un elemento del dataset che appartiene alla categoria 'Unealthy'.

Air_Quality_Category	**phi(Air_Quality_Category)**
Air_Quality_Category (Good)	0.0000

Air_Quality_Category (Moderate)	0.0000
Air_Quality_Category (Poor)	0.0000
Air_Quality_Category (Unhealthy)	1.0000

Ho provato a fare la stessa operazione con un elemento che appartiene alla categoria ‘Good’ che rappresenta la categoria meno frequente all’interno del dataset.

Air_Quality_Category	**phi(Air_Quality_Category)**
Air_Quality_Category (Good)	0.0451
Air_Quality_Category (Moderate)	0.0000
Air_Quality_Category (Poor)	0.9549
Air_Quality_Category (Unhealthy)	0.0000

Il risultato ottenuto è abbastanza deludente. Eseguendo altri test si è visto come i risultati tendessero a dare maggiore probabilità unicamente alle due classi più frequenti (‘Unhealthy’ e ‘Poor’).

Eseguendo altri test utilizzando dati ad hoc, sono stati ottenuti i risultati desiderati.

Sono stati eseguiti dei tentativi per la costruzione di una rete Bayesiana basata sul dataset bilanciato tramite SMOTE, ma senza successo. Infatti, da questo possiamo dedurre che una scarsa presenza di combinazioni di valori impedisce al modello di calcolare correttamente la CPD.

(Il file del modello bayesiano “Bayesian_model.plk” non è stato inserito nella documentazione poiché crea errori nel caricamento della cartella su GitHub, in ogni caso è possibile crearlo nuovamente)