Relazione progetto di fondamenti di scienza dei dati (Progetto F)

Dataset utilizzati:

- anonymized_radio_space_WM1_4.csv
- anonymized_radio_space_WM1_5.csv

I precedenti dataset sono stati combinati in un unico data set chiamato merged_dataset.csv, che è il dataset utilizzato per l'analisi dei dati.

Descrizione delle features

ant: Indica l'antenna utilizzata per la trasmissione o la ricezione del segnale.

chan: Rappresenta il canale di comunicazione utilizzato per la trasmissione radio.

codr: Codifica del tasso di correzione degli errori (coding rate) utilizzato, ad esempio "4/5".

created_at: Data e ora di creazione del record nel dataset.

datr: Data rate o modalità di trasmissione dati, ad esempio "SF7BW125" che indica il fattore di spreading e la larghezza di banda.

dev_addr: Indirizzo del dispositivo che ha trasmesso i dati.

dev_eui: Identificatore univoco del dispositivo.

dev_nonce: Numero casuale generato dal dispositivo per sicurezza.

freq: Frequenza radio utilizzata per la trasmissione, espressa in MHz (es. 868.500).

gateway: Identificatore del gateway che ha ricevuto il messaggio.

lsnr: Signal-to-noise ratio (SNR), rapporto segnale/rumore del segnale ricevuto.

ns_time: Data e ora in formato network server time.

rssi: Received Signal Strength Indicator, misura della potenza del segnale ricevuto, in dBm.

rssic: Potenza del segnale ricevuto dal canale di controllo, in dBm.

rssis: Potenza del segnale ricevuto dal canale di servizio, in dBm.

rssisd: Deviation in RSSI, differenza di potenza del segnale ricevuto, in dBm.

size: Dimensione del pacchetto di dati trasmesso, in byte.

time: Timestamp del pacchetto di dati ricevuto, in formato ISO 8601.

tmst: Timestamp a livello di gateway, utilizzato per la sincronizzazione temporale.

type: Tipo di messaggio, ad esempio "Unconfirmed Data Up" che indica un messaggio di dati non confermato inviato verso l'alto.

FCnt: Frame Counter, un contatore che indica il numero di frame trasmessi dal dispositivo.

valueRaw: Valore grezzo del payload trasmesso, che potrebbe rappresentare dati sensoriali o altri dati specifici del dispositivo.

PIPELINE DI DATA PROCESSING

Data collection

Il dataset "anonymized_radio_space" contiene dati relativi a comunicazioni radio anonimizzate. Include dettagli sulla qualità del segnale, temporizzazione precisa, modalità di trasmissione, interazioni tra dispositivi e gateway, e informazioni di sicurezza. Queste caratteristiche suggeriscono che il dataset potrebbe essere utilizzato per analizzare le performance delle reti radio, valutare la copertura del segnale, studiare l'uso delle frequenze, e migliorare l'infrastruttura di comunicazione radio.

Dati Anonimizzati:

- Come indicato nel nome del dataset e confermato dai valori presenti, le informazioni identificabili, come gli indirizzi dei dispositivi (dev_addr) e i loro identificatori univoci (dev_eui), sembrano essere anonimizzati. Questo è evidente dai valori come "00000000000000f9", che non forniscono dettagli specifici sull'identità dei dispositivi.

Comunicazioni Radio:

- I dati raccolti sono relativi alle comunicazioni radio. Questo è evidente dalle feature come freq (frequenza radio), datr (data rate), chan (canale) e rssi (Received Signal Strength Indicator).

Temporizzazione Dettagliata:

- Le temporizzazioni sono registrate con precisione, includendo sia il tempo di creazione (created_at) che il tempo specifico del network server (ns_time) e del gateway (tmst). Questo indica un alto livello di dettaglio nella registrazione temporale degli eventi.

Qualità del Segnale:

- Le misurazioni della qualità del segnale, come lsnr (Signal-to-Noise Ratio) e varie forme di rssi, indicano che il dataset contiene informazioni dettagliate sulla qualità della trasmissione radio. Valori come rssi, rssic, rssis, e rssisd forniscono una visione completa della potenza del segnale ricevuto e delle sue variazioni.

Dettagli di Trasmissione:

- La presenza di feature come codr (coding rate), datr (data rate), e size (dimensione del pacchetto) suggerisce che il dataset include dettagli specifici sulle modalità di trasmissione dei dati.

Interazioni tra Dispositivi e Gateway:

- La feature gateway mostra che i dati includono identificatori dei gateway che ricevono i messaggi dai dispositivi. Questo permette di analizzare le interazioni tra dispositivi e infrastruttura di rete.

Tipologia di Messaggi:

- La feature type indica il tipo di messaggio trasmesso, come "Unconfirmed Data Up". Questo può essere utile per distinguere tra diversi tipi di traffico e analizzare le loro caratteristiche.

Sicurezza e Integrità dei Dati:

- La presenza di dev_nonce e FCnt suggerisce l'uso di meccanismi per la sicurezza e l'integrità dei dati, comuni nei protocolli di comunicazione come LoRaWAN, che utilizzano contatori e numeri casuali per prevenire la ripetizione e garantire l'autenticità dei messaggi.

DATA PREPROCESSING

Feature extraction

Per ogni dispositivo estraggo le seguenti features: 'dev_addr', 'chan', 'datr', 'lsnr', 'rssic', 'FCnt'

La seguente istruzione mi permette di identificare il numero di dispositivi unici

```
num_device = device_data['dev_addr'].nunique()  # Numero di dispositivi unici
```

Numero dispositivi unici: 110

La seguente istruzione serve per contare il numero di pacchetti per dispositivo

```
num_package_for_dev = device_data.groupby('dev_addr').size()
```

groupby('dev_addr'): Raggruppa il DataFrame in base alla colonna 'dev_addr', cioè l'indirizzo del dispositivo.

size(): Calcola il numero di occorrenze (o righe) per ciascun gruppo di dispositivi. In altre parole, conta quanti pacchetti sono stati trasmessi da ciascun dispositivo.

Data transformation

```
means = numeric_data.mean()
stds = numeric_data.std()

normalized_dataset = (numeric_data - means) / stds
```

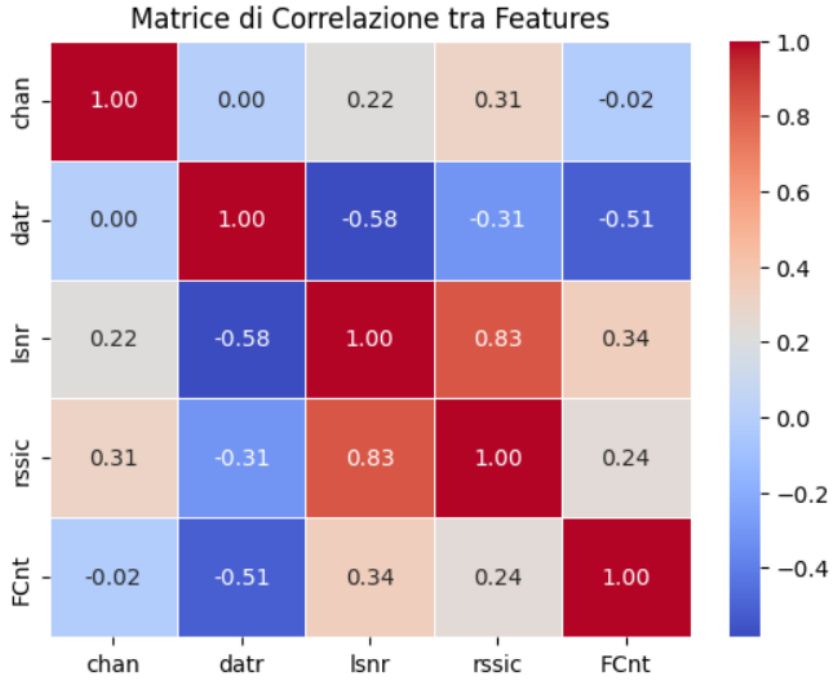
Il codice ha lo scopo di preparare il dataset per l'applicazione di tecniche di clustering, garantendo che le colonne numeriche siano standardizzate utilizzando la Z-score normalization. Questo serve per assicurarsi che tutte le feature abbiano la stessa scala e non siano influenzate dalle unità di misura originali, migliorando così l'efficacia dei metodi di clustering che utilizzano misure di distanza o similarità tra i dati.

Correlazione tra feature

Per esaminare le relazioni tra le caratteristiche (features) all'interno del dataset, ho selezionato casualmente 5 dispositivi e calcolato il coefficiente di correlazione utilizzando il metodo di Pearson (selected_data.corr()). Questa analisi mi ha permesso di identificare se esistono correlazioni significative tra le caratteristiche selezionate, fornendo così insight sulla natura delle interazioni tra i dati dei dispositivi nel contesto specifico del progetto.

$$\rho_{AB} = \frac{\operatorname{cov}(A, B)}{\sigma_A \cdot \sigma_B}$$

Output:



Valori vicini a 1 indicano una forte correlazione positiva tra le feature. Ad esempio:

- Tra 'lsnr' e 'rssic' il valore è 0.83.
- Tra 'lsnr' e 'FCnt' il valore è 0.34.
- Tra 'rssic' e 'FCnt' il valore è 0.24.

Questi valori suggeriscono che quando una di queste feature aumenta, l'altra tende a aumentare proporzionalmente, indicando una relazione positiva tra di loro.

Valori vicini a -1 indicano una forte correlazione negativa tra le feature. Ad esempio:

- Tra 'lsnr' e 'datr' il valore è -0.58.
- Tra 'lsnr' e 'rssic' il valore è -0.58.

Questi valori indicano che quando una feature aumenta, l'altra tende a diminuire proporzionalmente, suggerendo una relazione inversa tra di loro.

Valori vicini a 0 indicano una correlazione debole o assente tra le feature. Ad esempio:

- Tra 'chan' e 'datr' il valore è 0.
- Tra 'chan' e 'rssic' il valore è -0.02.
- Tra 'datr' e 'FCnt' il valore è -0.51.

Questi valori suggeriscono che non c'è una relazione lineare chiara tra queste coppie di feature.

Riduzione della dimensionalità

```
for device, group_data in grouped_by_device:
    mean_features = group_data[['chan', 'datr', 'lsnr', 'rssic']].mean()
    fcnt_difference = group_data['FCnt'].max() - group_data['FCnt'].min()

    new_features_dict = {
        'dev_addr': device,
        'Mean_Chan': mean_features['chan'],
        'Mean_Datr': mean_features['datr'],
        'Mean_Lsnr': mean_features['lsnr'],
        'Mean_Rssic': mean_features['rssic'],
        'FCnt_Difference': fcnt_difference
    }
```

Ad ogni dispositivo associo 5 nuove caratteristiche ottenute tramite la media delle prime 4 feature (chan, datr, lsnr, rssic) e la differenza MAX-MIN per FCnt.

Calcolare la media delle feature e la differenza MAX-MIN è un approccio efficace per ridurre la quantità di dati e sintetizzare le informazioni più rilevanti. Questo processo semplifica l'analisi successiva senza perdere dettagli importanti. Avere un numero ridotto di caratteristiche per ogni dispositivo rende l'analisi e la visualizzazione dei dati più semplice e gestibile.

Analizzare le caratteristiche medie per ogni dispositivo può aiutare a identificare pattern o tendenze comuni tra i dispositivi stessi. Confrontando le medie delle feature tra dispositivi, possiamo rivelare somiglianze o differenze significative, facilitando la segmentazione dei dispositivi in gruppi simili.

La differenza MAX-MIN di 'FCnt' può essere utilizzata per individuare dispositivi che mostrano comportamenti anomali o inconsueti. Ad esempio, un valore molto elevato potrebbe indicare un problema con il dispositivo o una situazione eccezionale.

Inoltre, analizzare le medie delle feature per ogni dispositivo può aiutare a monitorare le performance generali e identificare dispositivi che potrebbero necessitare di manutenzione o

ottimizzazione. Questo permette di mantenere sotto controllo la qualità del servizio o delle operazioni monitorate dai dispositivi, garantendo così un funzionamento efficiente e affidabile.

Clusterizzazione di tipo k-means

L'obiettivo di questa analisi è determinare il numero ottimale di cluster per il dataset "anonymized_radio_space" utilizzando due metodi distinti: il Metodo Elbow e il Silhouette Score. Questi metodi sono stati applicati sulle feature Mean_Chan, Mean_Datr, Mean_Lsnr, Mean_Rssic, e FCnt_Difference per valutare l'efficacia della clusterizzazione k-means.

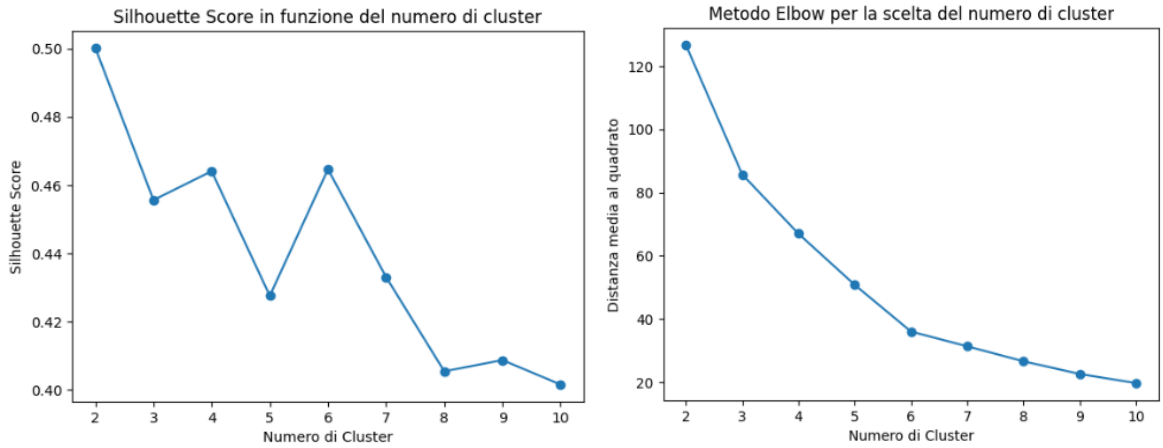


Grafico del Silhouette Score

Osservazioni:

- Il Silhouette Score è massimo per 2 cluster (circa 0.50).
- C'è una tendenza decrescente con l'aumentare del numero di cluster, con alcuni picchi minori (ad esempio, per 6 cluster).
- Il Silhouette Score si abbassa significativamente dopo 6 cluster.

Un valore di Silhouette Score più alto indica una migliore separazione e coesione dei cluster. Nel nostro caso, il massimo Silhouette Score si verifica con 2 cluster, suggerendo che questa potrebbe essere una buona scelta per ottenere una chiara distinzione tra i gruppi di dati. Tuttavia, il calo significativo del Silhouette Score dopo 6 cluster suggerisce che utilizzare più di 6 cluster potrebbe non migliorare la qualità del clustering e potrebbe anzi portare a una minore coesione e separazione tra i dati. Pertanto, l'analisi del Silhouette Score ci guida verso la considerazione di un numero di cluster compreso tra 2 e 6, con una particolare enfasi su 2 cluster come opzione iniziale ottimale.

Grafico del Metodo Elbow

Osservazioni:

- L'inertia (distanza media al quadrato) diminuisce rapidamente fino a circa 4 cluster, poi diminuisce più lentamente.
- Il "gomito" del grafico si osserva intorno a 4 cluster.

Il metodo Elbow suggerisce che 4 cluster potrebbero essere una scelta ottimale, poiché è il punto in cui la riduzione dell'inertia rallenta significativamente.
Anche se l'inertia continua a diminuire con più cluster, il beneficio aggiuntivo diminuisce dopo 4 cluster.

Considerazioni finali

Confluenza dei Risultati:

L'analisi condotta con i metodi Elbow e Silhouette Score offre indicazioni preziose per la scelta del numero ottimale di cluster. Mentre il Silhouette Score è più alto per 2 cluster, suggerendo una separazione e coesione ottimali, il metodo Elbow indica che 4 cluster rappresentano un buon compromesso tra complessità e qualità del clustering. Questo ci porta a considerare una gamma di opzioni: 2, 4 o 6 cluster, per ulteriori analisi e confronti.

Scelta del Numero di Cluster:

Iniziare con 2 cluster potrebbe essere una scelta conservativa, assicurando una buona separazione iniziale tra i dati. Tuttavia, è utile testare anche 4 e 6 cluster per verificare se queste configurazioni offrono insight migliori senza un significativo peggioramento del Silhouette Score. Tale approccio bilancia la semplicità con la possibilità di scoprire strutture più complesse nei dati, garantendo al contempo che la qualità del clustering rimanga alta.

Analisi dei cluster

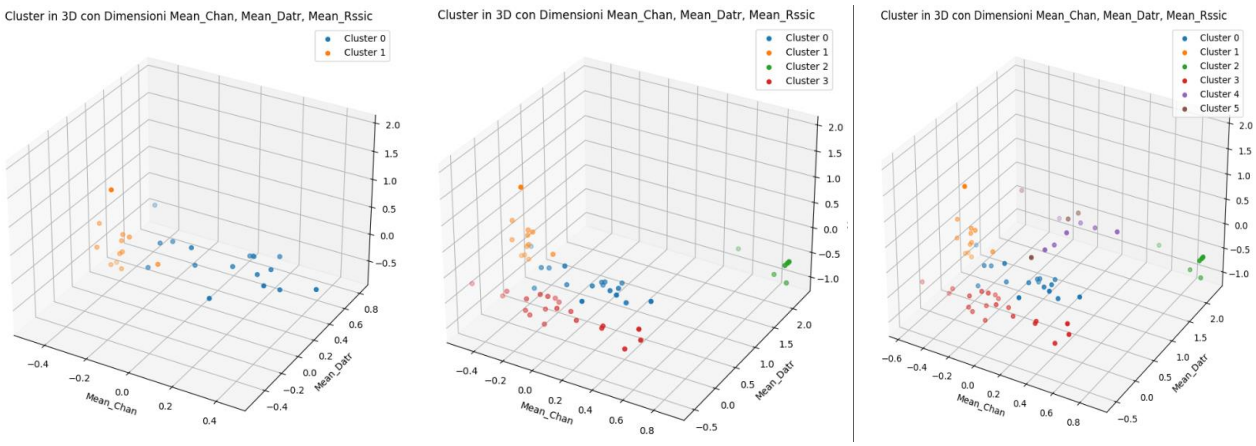


Grafico con 2 Cluster

- **Distribuzione**: Il primo grafico mostra una chiara divisione in due gruppi distinti. I punti del Cluster 0 e del Cluster 1 sono visivamente separati.
- **Densità**: Entrambi i cluster sembrano avere una densità simile, con una leggera sovrapposizione lungo alcune dimensioni.
- **Separazione**: La separazione tra i due cluster è abbastanza evidente, suggerendo che due cluster potrebbero essere sufficienti per una segmentazione iniziale dei dati.

Grafico con 4 Cluster

- **Distribuzione**: Aumentando a quattro cluster, si osserva una maggiore complessità nella distribuzione dei punti. I cluster aggiuntivi mostrano una maggiore granularità nella separazione dei dispositivi.
- **Densità**: Alcuni cluster appaiono più densi (ad esempio, Cluster 0 e Cluster 1) mentre altri sono più sparsi (ad esempio, Cluster 2).
- **Separazione**: I cluster sono ancora distinti, ma c'è una maggiore sovrapposizione rispetto al grafico con due cluster. Questo suggerisce che mentre quattro cluster catturano più dettagli, potrebbe esserci una leggera perdita di coesione all'interno dei cluster.

Grafico con 6 Cluster

- **Distribuzione**: Con sei cluster, la distribuzione dei punti diventa ancora più complessa. Ogni cluster contiene un numero inferiore di punti, indicando una segmentazione più fine dei dati.
- **Densità**: La densità varia notevolmente tra i cluster. Alcuni cluster (ad esempio, Cluster 0 e Cluster 1) rimangono densi, mentre altri (ad esempio, Cluster 4) sono molto sparsi.
- **Separazione**: La separazione tra i cluster è meno evidente rispetto ai grafici precedenti, con una maggiore sovrapposizione tra alcuni cluster. Questo indica che sei cluster potrebbero essere troppo dettagliati per ottenere una chiara distinzione tra tutti i gruppi.

Considerazioni finali

Man mano che il numero di cluster aumenta, la coerenza all'interno dei cluster tende a diminuire, portando a una maggiore sovrapposizione.
Aumentare il numero di cluster permette di catturare dettagli più fini nei dati, ma potrebbe complicare l'interpretazione e l'analisi successiva.
La scelta ottimale del numero di cluster dipenderà dall'obiettivo specifico dell'analisi. Due cluster offrono una segmentazione chiara e semplice, mentre quattro cluster bilanciano dettaglio e separazione. Sei cluster potrebbero essere utili per un'analisi molto dettagliata, ma a costo di una minore coesione all'interno dei cluster.

Gioacchino Augello
Francesco Rinaldi