Progetto stage metrica Density-Based Clustering Validation (DBCV) per la valutazione interna del clustering

Notebook

20/09/2024 → Google colab, python 3, CPU. Calcolo della metrica DBCV sui dataset. librerie utilizzate: dbcv: 0.1.0.

Dataset: “10_7717_peerj_5665_dataYM2018_neuroblastoma”
Best Parameters: min_cluster_size = 15, cluster_selection_epsilon = 0.01
DBCV: 0.763

Dataset: “Takashi2019_diabetes_type1_dataset_preprocessed”
Best Parameters: min_cluster_size = 3, cluster_selection_epsilon = 0.01
DBCV: 0.776

Dataset: “journal.pone.0148699_S1_Text_Sepsis_SIRS_EDITED”
Best Parameters: min_cluster_size = 10, cluster_selection_epsilon = 0.01
DBCV: 0.407

Dataset: “journal.pone.0158570_S2File_depression_heart_failure”
Best Parameters: min_cluster_size = 3, cluster_selection_epsilon = 0.01
DBCV Score with PCA: 0.459

Dataset: “journal.pone.0175818_S1Dataset_Spain_cardiac_arrest_EDITED”
Best Parameters: min_cluster_size = 3, cluster_selection_epsilon = 0.01
DBCV Score with PCA: 0.390

19/09/2024 → Google colab, python 3, CPU. Applicazione di PCA su ciascun dataset per ridurre la dimensionalità, seguita dall'esecuzione del clustering con l'algoritmo HDBSCAN utilizzando i parametri ottimali ottenuti tramite una ricerca dei migliori valori..librerie utilizzate: numpy: 1.26.4, matplotlib: 3.7.1, pandas: 2.1.4, scikit-learn: 1.3.2, hdbscan: 0.8.38.post1.

17/08/2024 → Google colab, python 3, CPU. Analisi esplorativa dei dataset, matrice di correlazione e barchart. librerie utilizzate: numpy: 1.26.4, matplotlib: 3.7.1, pandas: 2.1.4, seaborn: 0.13.1.

14/08/2024 → Google colab, python 3, CPU. Studio dataset
“journal.pone.0148699_S1_Text_Sepsis_SIRS_EDITED”
“journal.pone.0158570_S2File_depression_heart_failure” e
“journal.pone.0175818_S1Dataset_Spain_cardiac_arrest_EDITED”

13/08/2024 → Google colab, python 3, CPU. Studio dataset
“10_7717_peerj_5665_dataYM2018_neuroblastoma” e
“Takashi2019_diabetes_type1_dataset_preprocessed“

11/08/2024 → Google colab, python 3, CPU. Calcolo metrica DBCV con DBCV. Librerie utilizzate: scikit-learn: 1.3.2, hdbscan: 0.8.38.post DBCV from "git+https://github.com/christopherjenness/DBCV". Tempo totale di esecuzone 93.43 secondi sul dataset con 300 punti

09/08/2024 → Google colab, python 3, CPU. Confronto metriche DBCV, silhouette, dunn_index, davies_bouldin, calinski_harabasz. librerie utilizzate: sklearn.metrics,davies_bouldin_score, calinski_harabasz_score, dbcv from "git+https://github.com/FelSiq/DBCV".

08/08/2024 → Google colab, python 3, CPU. Calcolo metrica DBCV con Fast Density-Based Clustering Validation. Librerie utilizzate: scikit-learn: 1.3.2, hdbscan: 0.8.38.post1, dbcv from "git+https://github.com/FelSiq/DBCV". Tempo totale di esecuzione: 445.65 secondi sul dataset con 10000 punti. Tempo totale di esecuzione: 1.83 secondi sul dataset con 300 punti

07/08/2024 → Google colab, python 3, CPU. Calcolo metrica DBCV con formula matematica

```
FUNCTION calculate_dbcv_math(X, labels)
    n = length of X
    unique_labels = find unique values in labels

    IF length of unique_labels is less than 2 THEN
        RETURN -1

    distances = compute the square root of (((X[:, new axis, :] - X[new axis, :, :]) ^ 2).sum
along axis 2)

    dbcv_sum = 0
    FOR each i from 0 to n-1 DO
        internal_sum = 0
        FOR each j from 0 to n-1 DO
            IF i is not equal to j THEN
                max_dik = maximum of distances[i, k] for k from 0 to n-1 IF k is not equal to j
                internal_sum = internal_sum + distances[i, j] / max_dik
            END IF
        END FOR
        dbcv_sum = dbcv_sum + internal_sum / n
    END FOR

    dbcv = dbcv_sum / n
    RETURN dbcv
END FUNCTION
```

Tempo totale di esecuzione: 81.23 secondi sul dataset con 300 punti. Non eseguito con 10000 campioni per via delle tempistiche ottenute in primo test.

06/08/2024 → Google colab, python 3, CPU. Calcolo metrica DBCV con hdbscan.validity.validity_index Librerie utilizzate: scikit-learn: 1.3.2, hdbscan: 0.8.38.post.. Tempo totale di esecuzione: 176.28 secondi sul dataset con 10000 punti. Tempo totale di esecuzione: 2.13 secondi sul dataset con 300 punti

05/08/2024 →Google colab, python 3, CPU. Librerie utilizzate: scikit-learn: 1.3.2, hdbscan: 0.8.38.post1. Ricerca parametri migliori su entrambi i dataset di test tra 'min_samples': [1, 5, 10, 50] e 'cluster_selection_epsilon': [0.01, 0.1, 0.2, 0.5]. Risultati migliori ottenuti con min_samples = 10, 'cluster_selection_epsilon'= 0.01.

04/08/2024 → Google colab, python 3, CPU. Librerie utilizzate: scikit-learn: 1.3.2, hdbscan: 0.8.38.post1. Generati 10 dataset randomici con diversa tipologia di rumore. Utilizzo HDBSCAN con min_sample = 5. Primo test effettuato con dataset generati con make_moons:

n_samples=300, noise=0.0
n_samples=300, noise=0.056
n_samples=300, noise=0.111
n_samples=300, noise=0.167
n_samples=300, noise=0.222
n_samples=300, noise=0.278
n_samples=300, noise=0.333
n_samples=300, noise=0.389
n_samples=300, noise=0.444
n_samples=300, noise=0.5

Secondo test effettuato con dataset generati con make_moons:

n_samples=10000, noise=0.0
n_samples=10000, noise=0.056
n=10000, noise=0.111
n_samples=10000, noise=0.167
n_samples=10000, noise=0.222
n_samples=10000, noise=0.278
n_samples=10000, noise=0.333
n_samples=10000, noise=0.389
n_samples=10000, noise=0.444
n_samples=10000, noise=0.5

Requirements:

```
numpy: 1.26.4
scikit-learn: 1.3.2
matplotlib: 3.7.1
hdbscan: 0.8.38.post1
scipy: 1.13.1
```

https://github.com/christopherjenness/DBCV

https://github.com/FelSiq/DBCV

```
Dataset: Uguale anche con 10.000
    n_samples=300, noise=0.0
    n_samples=300, noise=0.056
    n_samples=300, noise=0.111
    n_samples=300, noise=0.167
    n_samples=300, noise=0.222
    n_samples=300, noise=0.278
    n_samples=300, noise=0.333
    n_samples=300, noise=0.389
    n_samples=300, noise=0.444
    n_samples=300, noise=0.5
```