UNIVERSITÀ
DEGLI STUDI
DI MILANO

ANALISI DELLE DENSITÀ CON OPTICS

Esplorazione dei dataset e identificazione dei cluster attraverso l'algoritmo OPTICS

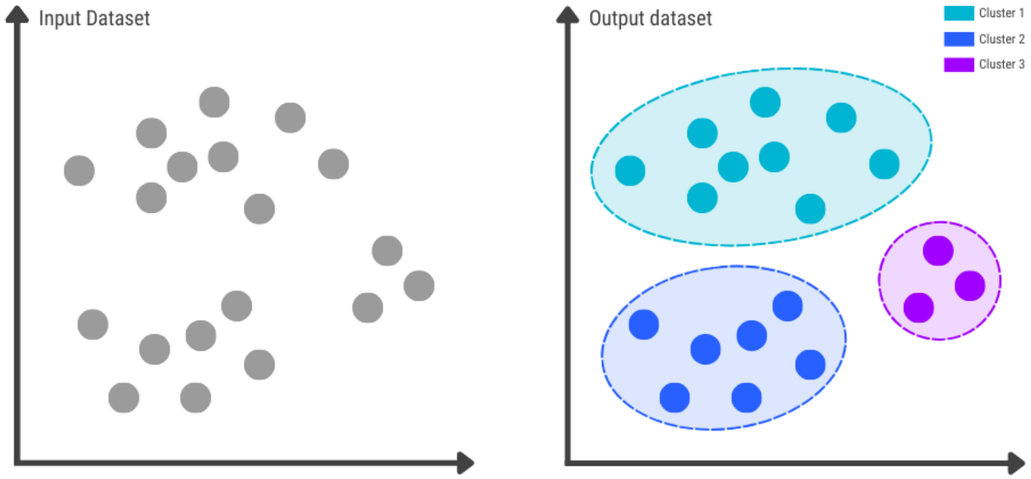
Presentato da:
Giulia M. Colombo
Paolo Gavagni
Laura Grosini

Corso:
Principi e modelli della percezione

Link GitHub:
https://github.com/giuliamartinacolombo/PrincipiEModelliDellaPercezione

IL CLUSTERING

Che cos'è?



Il **clustering** è una tecnica di machine learning non supervisionato che mira a raggruppare dati simili tra loro, senza l'uso di etichette predefinite.

È ampiamente utilizzato in ambito commerciale, scientifico e tecnologico.

Offre diversi benefici chiave nell'analisi dei dati, tra cui: analisi esplorativa, riduzione della complessità, pre-elaborazione per altre tecniche e applicazioni pratiche

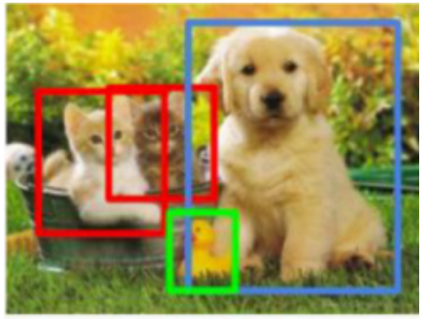
MA COSA VUOL DIRE “NON SUPERVISIONATO”?

Nel machine learning, esistono due grandi categorie di algoritmi:

01 **Apprendimento supervisionato**

↓

L’algoritmo impara da un insieme di dati etichettati. Significa che per ogni esempio fornito, si conosce già la “risposta corretta”.

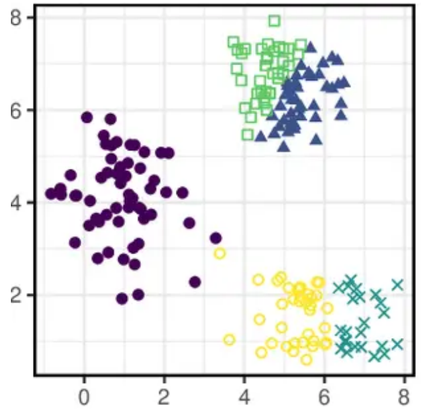


CAT, DOG, DUCK

02 **Apprendimento NON supervisionato**

↓

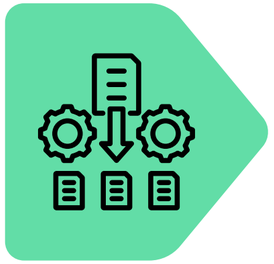
L’algoritmo riceve solo i dati grezzi (punti, immagini, segnali, ecc.) e deve scoprire da solo se esistono gruppi o strutture nascoste.



COME FUNZIONA IL CLUSTERING

01. PRE-ELABORAZIONE DEI DATI

- Rimozione di valori mancanti, duplicati e feature irrilevanti
- Normalizzazione e ridimensionamento dei dati per evitare che variabili con valori più grandi dominino sulle altre

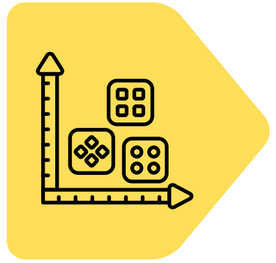


02. ESTRAZIONE DELLE CARATTERISTICHE

- Identificazione o creazione delle caratteristiche più rappresentative
- Permette all'algoritmo di concentrarsi sugli aspetti davvero rilevanti del dataset

03. MISURAZIONE DELLA DISTANZA

- Gli algoritmi raggruppano i dati utilizzando metriche di distanza o similarità
- Misurano quanto i punti dati siano vicini o lontani tra loro

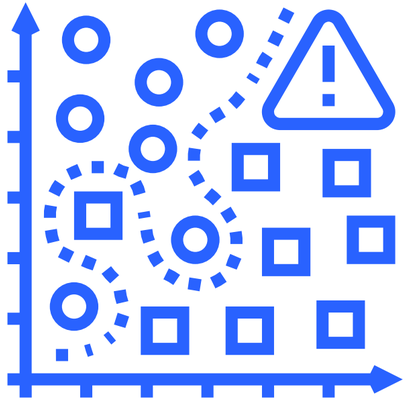


04. FORMAZIONE DEI CLUSTER

- Dopo il calcolo delle distanze, l'algoritmo raggruppa i punti in cluster
- Ogni cluster riunisce punti più simili tra loro che rispetto agli altri gruppi

05. VALUTAZIONE E PERFEZIONAMENTO

- I cluster vengono rivisti e perfezionati iterativamente
- L'algoritmo aggiorna i confini per renderli più coerenti e significativi



TIPOLOGIE DI CLUSTERING

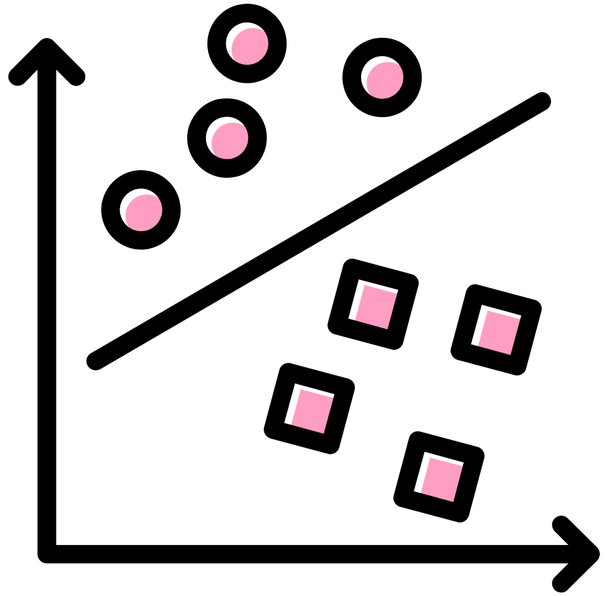
Gli algortimi di clustering vengono classificati in tre principali categorie:

01 Basati sulla **distanza** *(es. K-Means, K-Medoids)*

02 Basati sulla **densità** *(es. DBSCAN, OPTICS)*

03 Basati su **modelli** o **gerarchie** *(es. Agglomerative Clustering, Gaussian Mixture)*

CONCENTRIAMOCI SULL'ALGORITMO OPTICS!



PANORAMICA SU OPTICS

OPTICS (Ordering Points to Identify the Clustering Structure) è un algoritmo di clustering basato sulla **densità**, particolarmente utile per identificare cluster di **densità variabile**.

Clustering basato sulla densità

Gestione di densità variabili

Nessuna necessità di definire il numero di cluster

Reachability Plots

Gestione complessa dei dati

PRIMA DI SPIEGARE IL FUNZIONAMENTO...

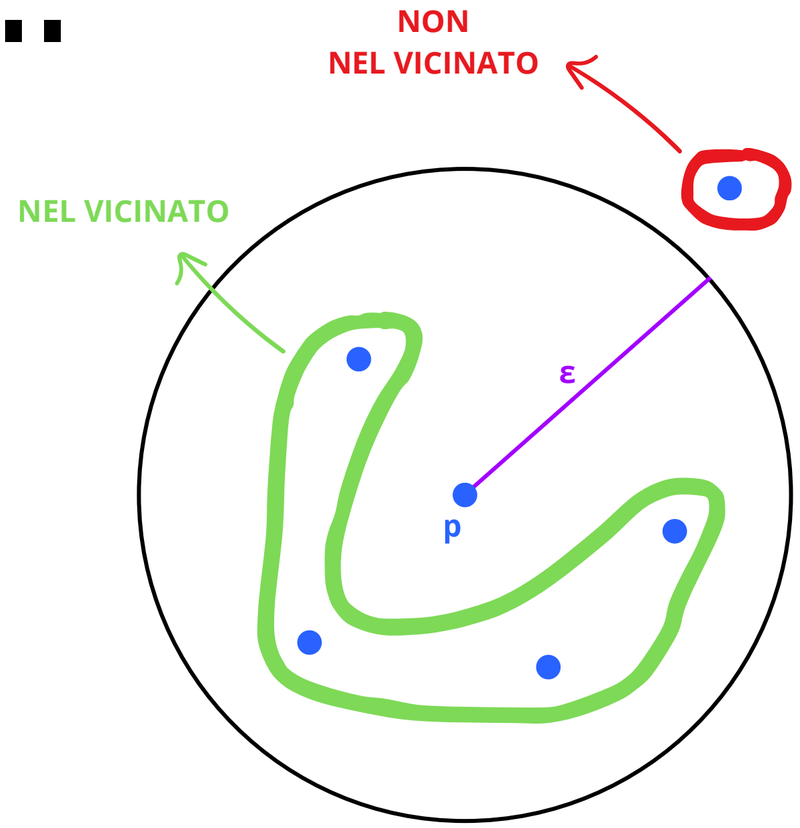
Qualche concetto chiave...

EPSILON (ε) e MINPTS

Epsilon (ε) è la massima distanza entro la quale si considerano i punti "vicini" rispetto a un punto dato p. In pratica, intorno a ogni punto si traccia un cerchio di raggio ε: tutti i punti che cadono dentro quel cerchio vengono considerati nel suo vicinato.

MinPts è il numero minimo di punti che devono trovarsi all'interno del cerchio di raggio ε affinché il punto p sia considerato un core point.

Ad esempio, se scegliamo *MinPts = 5*, allora p potrà essere considerato un core point.



CORE POINTS

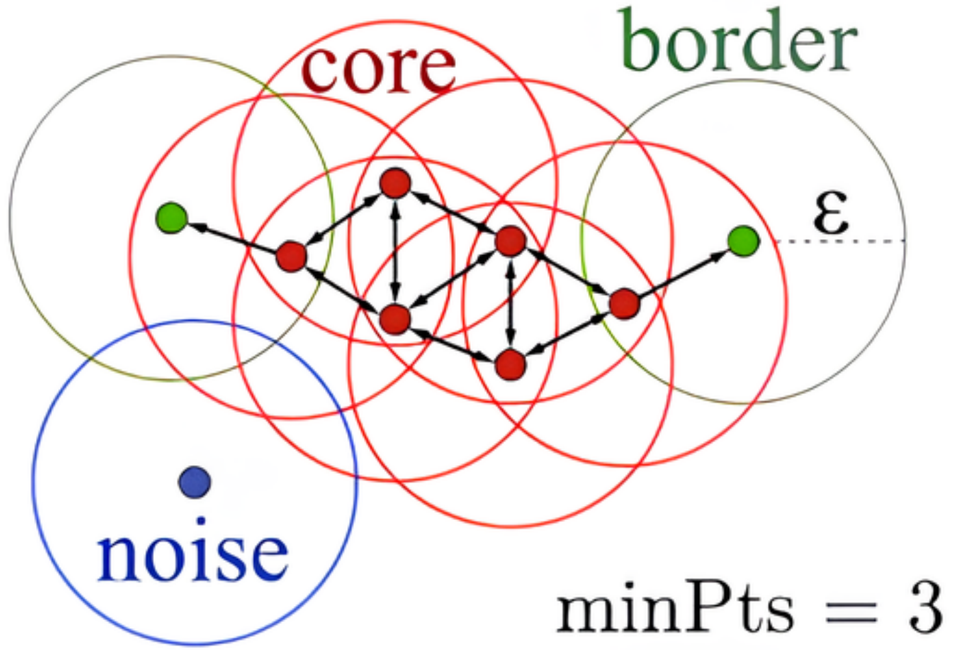
I **core points** sono considerabili la base su cui si costruiscono i cluster. Hanno un numero sufficiente di punti vicini (almeno MinPts in uno specifico raggio), che formano una regione densa. Sono fondamentali per definire dove un cluster ha inizio e come si estende.

BORDER POINTS

I **border points** si trovano ai margini di un cluster, delineandone il bordo: non hanno punti abbastanza vicini per essere considerabili core points, ma rientrano nel "vicinato" di uno o più core points. Si trovano abbastanza vicini da essere parte di un cluster, ma non abbastanza per essere core points.

NOISE POINTS

I **noise points** stanno fuori dai cluster in quanto non hanno abbastanza vicini per essere definiti core points, e non rientrano nel "vicinato" di nessuno di essi. Per questo sono considerati **outlier**/anomalie e spesso corrispondono a dati irregolari o molto dispersi che non si inseriscono in nessun cluster. Riconoscerli è importante filtrare le informazioni irrilevanti o inaccurate e a migliorare la qualità dei cluster.



CORE DISTANCE

La **core distance** serve per capire se un punto può essere considerato parte del "cuore" di un cluster: *più vicini ha attorno a sé, più è probabile che appartenga a un cluster ben formato*.
Per un punto (p), la core distance è la distanza che lo separa dal suo MinPts-esimo vicino più vicino. Questo valore riflette il livello di densità locale:

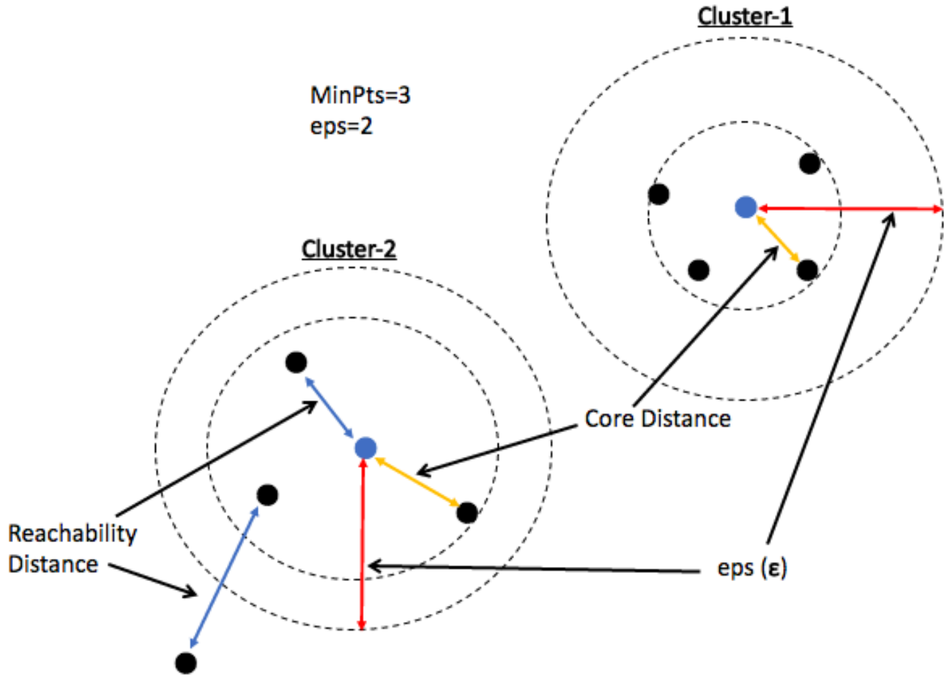
- se il punto **ha almeno MinPts vicini in un raggio ragionevole**, è abbastanza "circondato" da altri punti e diventa un core point
- se i **vicini sono pochi o troppo distanti**, il punto non ha sufficiente densità attorno a sé e non può dare origine a un cluster.

REACHABILITY DISTANCE

La **reachability distance** misura quanto un punto è "raggiungibile" da un altro in una zona densa, considerando, oltre alla distanza geometrica, anche la densità locale.

È definita come il massimo tra la core distance del punto di partenza e la distanza effettiva tra i due punti.

- Valori **bassi** → continuità in regioni dense;
- Valori **alti** → transizioni verso bordi o zone sparse, permettendo a OPTICS di distinguere cluster e rumore.



REACHABILITY PLOT

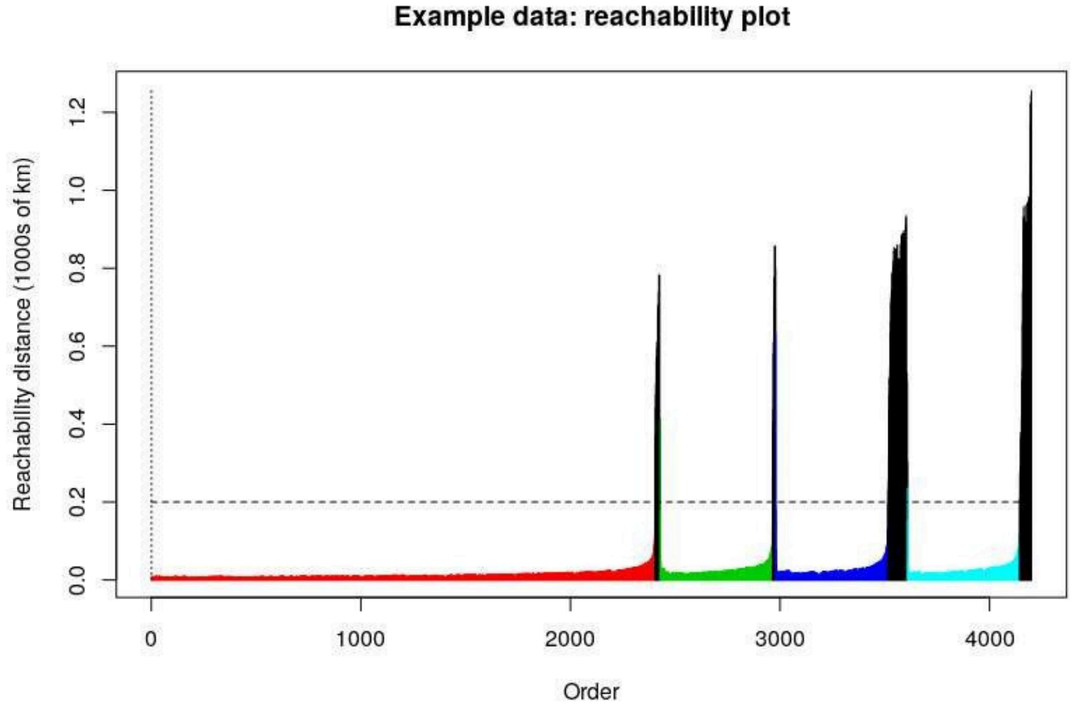
Il **reachability plot** è uno degli elementi più caratteristici e potenti di OPTICS. Si tratta di un grafico dove i punti vengono ordinati secondo la sequenza con cui l'algoritmo li visita, e rappresentati in base alla loro reachability distance. Visualmente:

- Le **"vallate"** indicano regioni a bassa reachability distance, quindi zone dense → corrispondono ai cluster. I drop significativi segnano l'ingresso in una zona densa e i minimi locali ne rappresentano il "cuore".

- I **"picchi"** rappresentano punti con alta reachability distance → spesso indicano rumore, outlier, o transizioni tra cluster, cioè i confini tra una regione densa e l'altra.

Questo tipo di rappresentazione è estremamente utile perché permette di osservare:

- cluster di forma qualsiasi,
- cluster con densità molto diverse tra loro,
- cluster annidati uno dentro l'altro,
- e la distribuzione del rumore nel dataset.



FUNZIONAMENTO ALGORITMO

01. IDENTIFICARE I CORE POINTS

l'algoritmo identifica i **core points**: quando un punto soddisfa le condizioni per essere core, tutti i punti nel suo intorno vengono considerati parte dello stesso gruppo. L'algoritmo quindi individua i punti direttamente raggiungibili dai core point, che sono gli unici da cui può iniziare la formazione effettiva di un **cluster**.

02. DEFINIRE LE REACHABILITY DISTANCES

Si calcola la **reachability distance**. Il risultato serve a costruire il **reachability plot**: valori bassi indicano aree dense e ben definite, mentre valori alti segnalano bordi di cluster o punti isolati.

03. SCOPRIRE I BORDER POINTS

L'algoritmo identifica i **border points**. Anche se non rappresentano il centro del cluster, ne fanno comunque parte e ne ampliano la forma, creando una continuità naturale tra diverse zone dei dati. I punti di bordo contribuiscono alla **densità** del cluster e servono a **evitare** che gruppi di punti vengano **separati in modo artificiale**.

04. CLASSIFICARE I NOISE POINTS

Vengono identificati i **noise points**. Nel reachability plot compaiono come **punti isolati** con distanze elevate, segnalando che non appartengono a nessun cluster significativo. In pratica, rappresentano valori **rari** o **poco rilevanti** rispetto alla struttura principale dei dati e vengono **esclusi** dai cluster finali.

05. COSTRUZIONE DEL REACHABILITY PLOT

Viene costruito il **reachability plot**. Analizzando la forma del plot è possibile individuare facilmente i cluster e anche la loro gerarchia, scegliendo una soglia di reachability a cui "tagliare" il grafico. Questa flessibilità rende OPTICS molto utile quando non si conosce in anticipo il numero di cluster presenti nei dati.

OPTICS vs DBSCAN vs K-MEANS

OPTICS

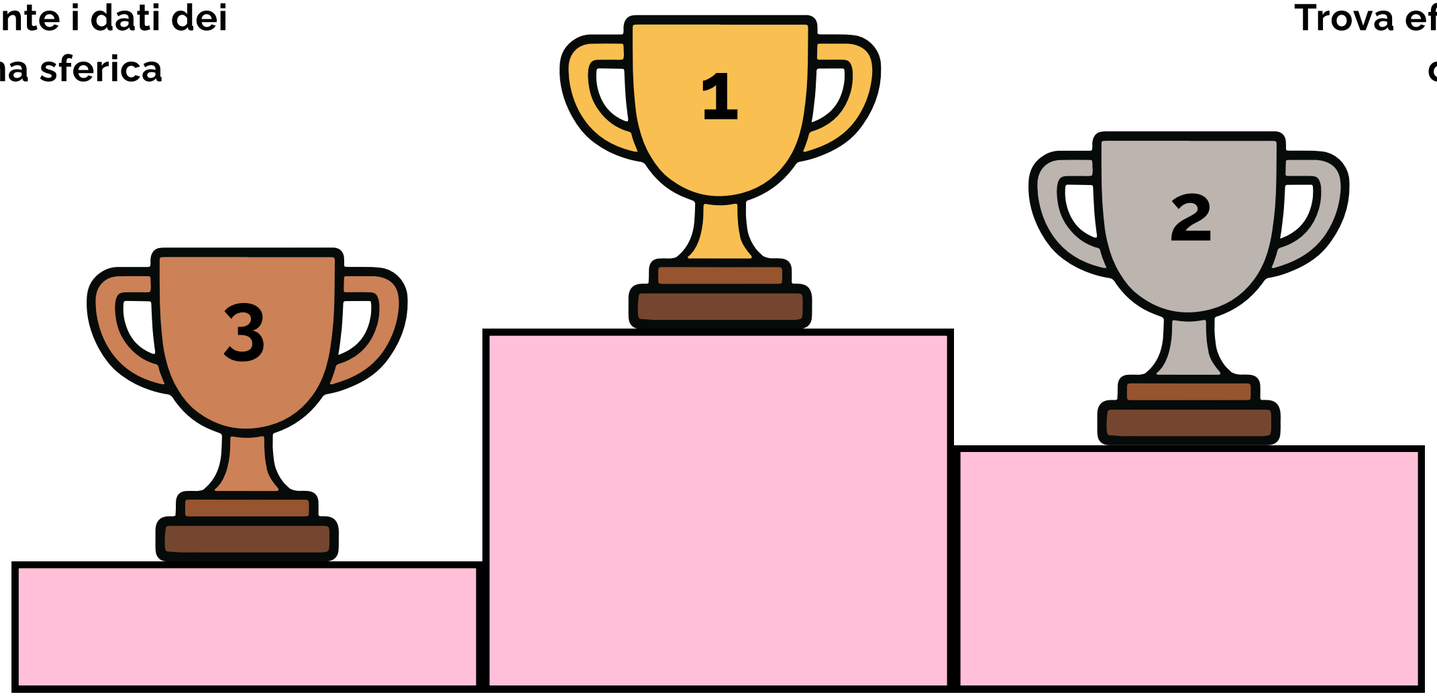
Eccelle nell'identificazione di clusters di densità variabili

K-MEANS

Trova efficientemente i dati dei clusters in forma sferica

DBSCAN

Trova efficacemente clusters di densità uniforme



APPLICAZIONE DI OPTICS SU UN INSIEME DI DATI REALI

DATASET: “Penguins”

OBIETTIVI E STRUMENTI

Obiettivi:

L'obiettivo di questa parte pratica è **vedere come OPTICS si comporta** su un dataset reale con poche dimensioni e una possibile struttura a cluster e, allo stesso tempo, **capire se nei dati** dei pinguini **emergono** davvero **gruppi di individui con caratteristiche simili** (ad esempio per dimensioni, massa corporea e sesso).

Strumenti utilizzati:

Per analizzare il comportamento dell'algoritmo OPTICS è stato utilizzato il **dataset "*Penguins*"**, che contiene una serie di misure morfologiche di un ampio numero di pinguini (lunghezza e profondità del becco, lunghezza della pinna, massa corporea) e il loro sesso.

L'algoritmo è stato implementato in **Python**, utilizzando le principali librerie per l'analisi dei dati e il machine learning.

DAL PUNTO DI VISTA IMPLEMENTATIVO...

01. SETTING DELL'AMBIENTE

Per prima cosa sono state importate tutte le librerie necessarie al funzionamento del programma:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt #per i grafici

from sklearn.preprocessing import StandardScaler #per la standardizzazione
delle feature
from sklearn.cluster import OPTICS #algoritmo di clustering
from sklearn.decomposition import PCA #per la riduzione della dimensionali
```

Il dataset è stato caricato da file CSV in un DataFrame *pandas* e ne è stata effettuata una prima ispezione:

```
df = pd.read_csv(file_path)
print(df.head())
print(df.info())
```

L'output mostra le prime righe e alcune informazioni strutturali:

- 344 righe totali
- 5 colonne:
  - culmen_length_mm,culmen_depth_mm, flipper_length_mm, body_mass_g (di tipo *float64*)
  - sex (di tipo *object*)

Viene inoltre evidenziata la presenza di valori mancanti.

IMPLEMENTAZIONE

02. PRE-PROCESSING DEI DATI

Prima di poter applicare l'algoritmo è stato necessario ripulire e trasformare i dati, mediante i seguenti step:

Rimozione dei valori mancanti:

Si eliminano le righe contenenti almeno un valore mancante; questo riduce leggermente il numero di pinguini ma garantisce che l'algoritmo lavori su un dataset completo.

```
df = df.dropna()
```

Come risultato si hanno 335 righe totali

Codifica della variabile categorica *sex*

OPTICS richiede **esclusivamente feature numeriche**. La variabile sex è stata trasformata in una variabile binaria:

```
df = pd.get_dummies(df, columns=["sex"], drop_first=True)
print(df.head())
```

L'opzione *drop_first=True* **evita la collinearità**: invece di creare due colonne (*sex_FEMALE*, *sex_MALE*) ridondanti, viene mantenuta solo *sex_MALE* (0 = femmina, 1= maschio)

IMPLEMENTAZIONE

03. SELEZIONE DELLE FEATURE

Sono state selezionate tutte le feature disponibili per il clustering:

- **culmen_length_mm** -> Lunghezza del becco
- **culmen_depth_mm** -> Profondità del becco
- **flipper_length_mm** -> Lunghezza della pinna
- **body_mass_g** -> Massa corporea
- **sex_MALE** -> Sesso

```
features = [
    "culmen_length_mm",
    "culmen_depth_mm",
    "flipper_length_mm",
    "body_mass_g",
      “sex_male” #flag che indica il sesso
]

X = df[features].values
```

IMPLEMENTAZIONE

04. STANDARDIZZAZIONE DELLE FEATURE

Le variabili presentano **scale molto diverse** (millimetri, grammi, variabile binaria). Poiché OPTICS si basa sulle distanze, è necessario riportarle su una **scala comparabile** tramite **standardizzazione**, per evitare che alcune feature **influenzino** maggiormente l'algoritmo.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X) #applica ad ogni colonna una formula


print("Shape X_scaled:", X_scaled.shape)
print("Prima riga:", X_scaled[0])
```

L'output conferma che:

- ***X_scaled*** ha dimensione (335, 5)
- Ogni riga rappresenta un pinguino in uno spazio a 5 dimensioni standardizzato, ad esempio:

  Prima riga: [-0.89772327 0.77726336 -0.12689335 -0.57223347 0.99108452]

IMPLEMENTAZIONE

05. APPLICAZIONE DI OPTICS ...

L'algoritmo OPTICS è stato applicato a X_scaled con i seguenti parametri:

- ***min_samples*** **= 10** -> numero minimo di punti nel vicinato di un punto perché *questo sia considerato* ***core point***;
- ***xi*** **= 0.05** -> controlla la **sensibilità alle variazioni di densità**: valori più piccoli rendono OPTICS più sensibile a cambiamenti locali e tendono a produrre più cluster;
- ***min_cluster_size*** **= 0.05** -> **dimensione minima** del cluster pari al 5% del numero totale di punti.

```
optics = OPTICS(
      min_samples=10,
      xi=0.05,
      min_cluster_size=0.05
)

optics.fit(X_scaled)
```

IMPLEMENTAZIONE

05. ...APPLICAZIONE DI OPTICS

Dopo il fit vengono estratte le etichette di cluster:

Le etichette hanno il seguente significato:

- -1 -> punti considerati **rumore** (outlier)
- 0, 1, 2 … -> **cluster individuati** dall'algoritmo

```
labels = optics.labels_ #array delle etichette dei cluster
unique, counts = np.unique(labels, return_counts=True)

print("Label uniche:", unique)
print("Distribuzione:", dict(zip(unique, counts)))
```

Nell'esperimento con tutte le feature (incluso *sex_MALE*) OPTICS ha individuato **6 cluster** (0-5) più una quota di **rumore** (-1).

```
Label uniche: [-1  0  1  2  3  4  5]
Distribuzione: {
  -1: 141,
   0: 25,
   1: 33,
   2: 20,
   3: 18,
   4: 75,
   5: 23         }
```

(La somma corrisponde ai 335 pinguini rimasti dopo il preprocessing).

IMPLEMENTAZIONE

06. COSTRUZIONE DEL REACHABILITY PLOT...

Per interpretare il risultato di OPTICS è stato costruito il **reachability plot**, che *rappresenta la reachability distance dei punti nell'ordine in cui vengono visitati dall'algoritmo*.

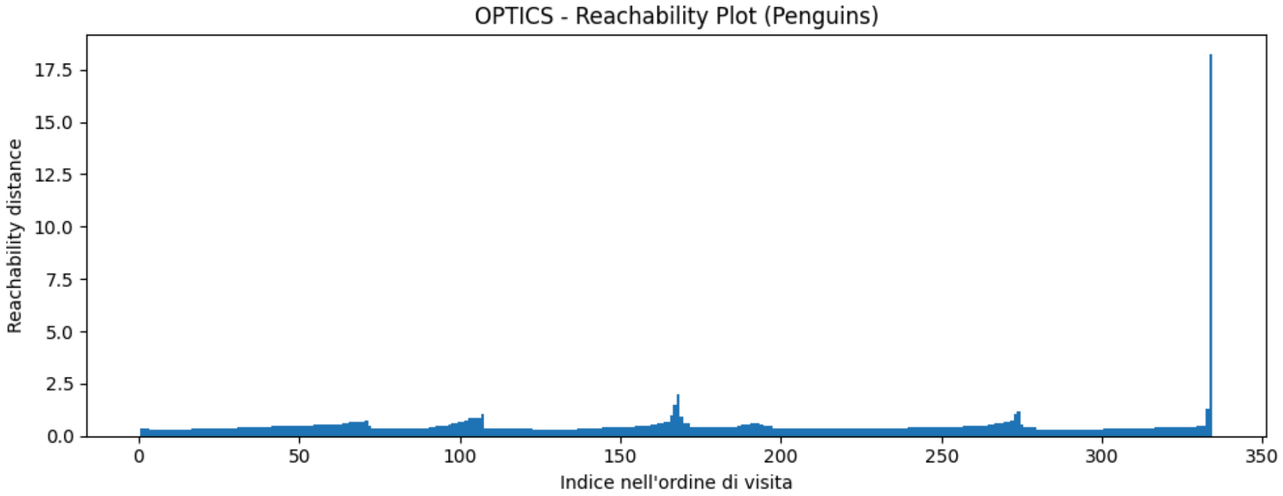
```
ordering = optics.ordering_
reachability = optics.reachability_[ordering]
```

```
#Grafico del reachability plot
plt.figure(figsize=(10, 4))
plt.title("OPTICS - Reachability Plot (Penguins)")
plt.xlabel("Indice nell'ordine di visita")
plt.ylabel("Reachability distance")
plt.bar(np.arange(len(reachability)), reachability, width=1.0)
plt.tight_layout()
plt.show()
```

- ***ordering*** è la permutazione degli indici dei pinguini secondo l'ordine di visita di OPTICS (guidato dalla densità, non dall'ordine originario del dataset)
- ***reachability*** contiene, per ciascun punto in questo ordine, la sua reachability distance.

IMPLEMENTAZIONE

06. ...E ANALISI



Nel grafico risultante si osservano:

- "**vallate**" a bassa reachability distance -> regioni dense dello spazio dei dati, corrispondenti ai cluster;
- **picchi più elevati** -> punti isolati o transizioni tra cluster, spesso associati a rumore.

(L'enorme picco all'estremità rappresenta un punto estremamente distante (altissima reachability distance), e viene quindi classificato come rumore).

IMPLEMENTAZIONE

07. VISUALIZZAZIONE DEI CLUSTER TRAMITE PCA 2D...

Per verificare i cluster è stata applicata la PCA (Principal Component Analysis) per ridurre lo spazio da 5 a 2 dimensioni:

```
pca = PCA(n_components=2) #riduzione a 2 dimensioni
X_pca = pca.fit_transform(X_scaled)
print("Varianza spiegata:", pca.explained_variance_ratio_)
```

Le due componenti principali (PC1, PC2) spiegano una frazione consistente della varianza totale. I punti sono stati proiettati nel piano PC1 - PC2 e colorati in base all'etichetta di cluster:

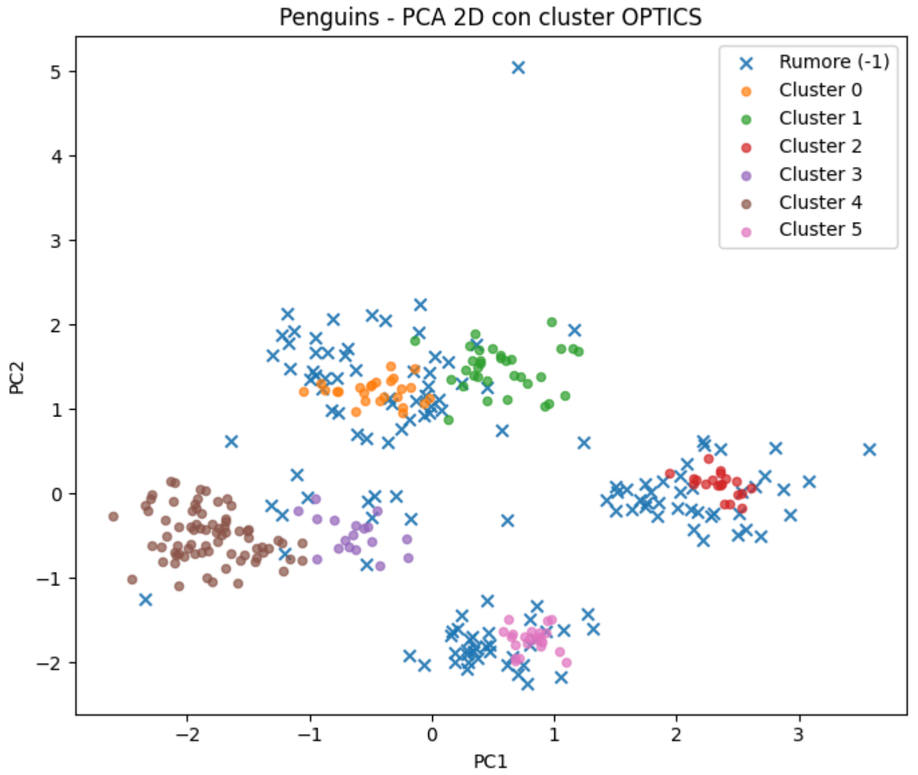
```
for cluster_id in np.unique(labels):
    mask = labels == cluster_id
    if cluster_id == -1:    #rumore
        lab = "Rumore (-1)"
        marker = "x"
        size = 40
        alpha = 0.9
else:
    lab = f"Cluster {cluster_id}"
    marker = "o"
    size = 20
    alpha = 0.7
plt.scatter(
    X_pca[mask, 0],
    X_pca[mask, 1],
    s=size,
    marker=marker,
    alpha=alpha,
    label=lab
)
```

IMPLEMENTAZIONE

07. ...E ANALISI

Nel grafico risultante si osservano:

- I **punti** appartenenti allo stesso cluster tendono a formare gruppi compatti e ragionevolmente separati;
- Il **rumore** (label = -1) appare più disperso, spesso ai margini delle regioni occupate dai cluster.



RISULTATI

Per sintetizzare il risultato è stato creato un DataFrame esteso con l'assegnazione di cluster:

```
df_clusters = df.copy()
df_clusters["cluster"] = labels

print("\nDistribuzione dei pinguini per cluster:")
print(df_clusters["cluster"].value_counts().sort_index())

print("\nMedie per cluster:")
print(df_clusters.groupby("cluster")[features].mean())
```

Distribuzione dei pinguini per cluster:

cluster	n_pinguini
-1	141
0	25
1	33
2	20
3	18
4	75
5	23

Medie di valori per cluster:

cluster	culmen_length	culmen_depth	flipper_length	body_mass	sex_MALE
-1	44.773759	16.804255	237.801418	4557.446809	0.64539
0	40.744000	18.708000	191.240000	3931.000000	1.00000
1	51.066667	19.209091	199.606061	3912.878788	1.00000
2	49.775000	15.960000	223.500000	5642.500000	1.00000
3	46.488889	17.483333	191.944444	3608.333333	0.00000
4	37.565333	17.549333	187.720000	3357.666667	0.00000
5	46.500000	14.378261	213.347826	4802.173913	0.00000

ANALISI DEI RISULTATI

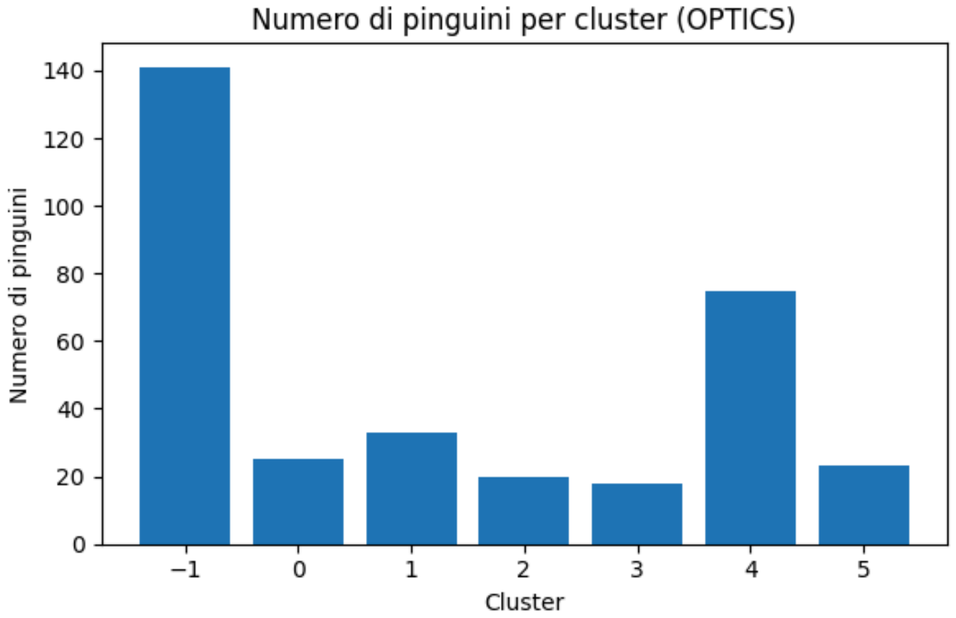
DISTRIBUZIONE DEI PINGUINI PER CLUSTER:

Questa distribuzione è stata anche rappresentata con un **istogramma** del numero di pinguini per cluster, da cui si vede subito che:

- il **cluster 4** è il più ampio (75 individui)
- il **rumore** (-1) con 141 punti è molto numeroso
- i **cluster 0, 1, 2, 3** e **5** hanno quantità più contenute.

L'istogramma rende visivamente chiaro che:

- OPTICS individua **pochi cluster "grandi"** (in particolare il 4) e alcuni cluster più piccoli, che rappresentano sottogruppi morfologicamente più specifici
- Una parte importante del dataset viene comunque etichettata come **rumore** , ossia punti che non rientrano in regioni abbastanza dense.



ANALISI DEI RISULTATI

MEDIE DI VALORI PER CLUSTER:

Per rendere più leggibili gli scostamenti rispetto alla media globale, è stata costruita una **heatmap** delle medie sul dataset standardizzato.
Questa mostra, per ogni cluster e feature, quanto il valore medio è sopra o sotto la soglia globale (0).

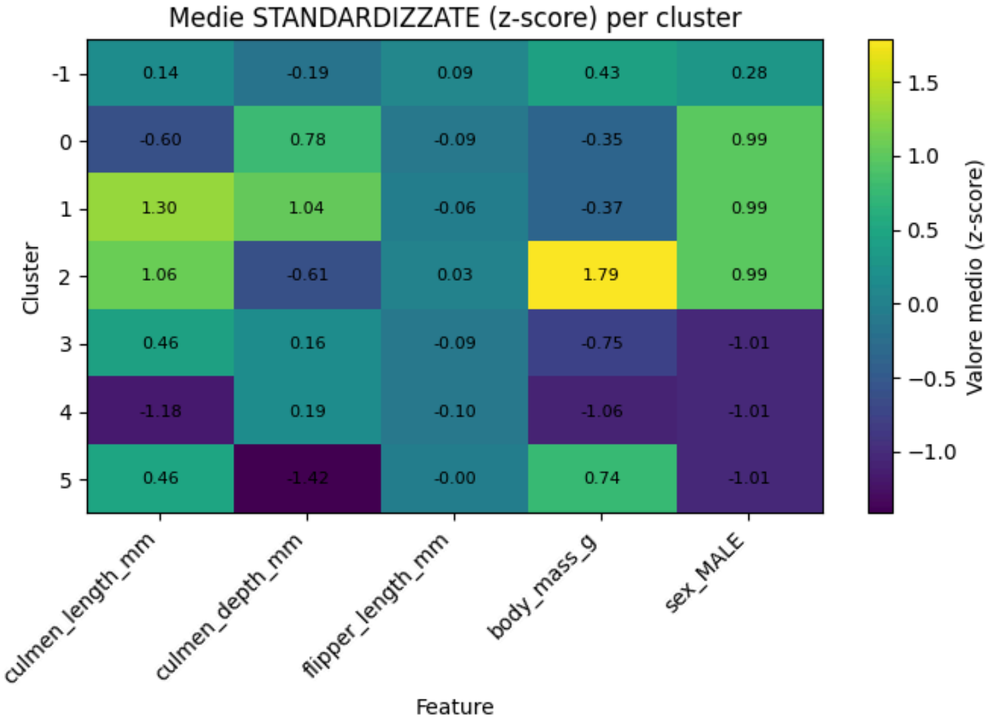
Si osserva che, per i cluster di maschi:

- il **cluster 2** ha un forte valore positivo su *body_mass_g* -> pinguini molto pesanti;
- il **cluster 1** spicca per lunghezza e profondità del becco sopra la media;
- il **cluster 0** ha becco meno lungo ma più profondo.

Per quanto riguarda i cluster di femmine:

- il **cluster 4** ha valori negativi su quasi tutte le feature -> femmine piccole e leggere;
- il **cluster 5** mostra massa sopra la media ma profondità del becco sotto la media -> femmine più pesanti con becco meno profondo;
- il **cluster 3** è vicino alla media su gran parte delle feature

Il **rumore** (-1) ha valori moderati, senza scostamenti estremi



*Per leggere correttamente il grafico è importante considerare **l'intensità** del colore, che misura quanto un valore si discosta dalla media.*

ESPERIMENTO AGGIUNTIVO

RIMOZIONE DELLA FEATURE sex_MALE

Per valutare l'impatto della scelta delle feature, è stato condotto un secondo esperimento eliminando la variabile *sex_MALE* e utilizzando soltanto le quattro feature morfologiche:

- **culmen_length_mm**
- **culmen_depth_mm**
- **flipper_length_mm**
- **body_mass_g**

*I parametri di OPTICS sono stati mantenuti invariati (**min_samples** = 10, **xi** = 0.05, **min_cluster_size** = 0.05) per rendere il confronto significativo.*

Applicando OPTICS, abbiamo come risultato:

Distribuzione di pinguini per cluster:

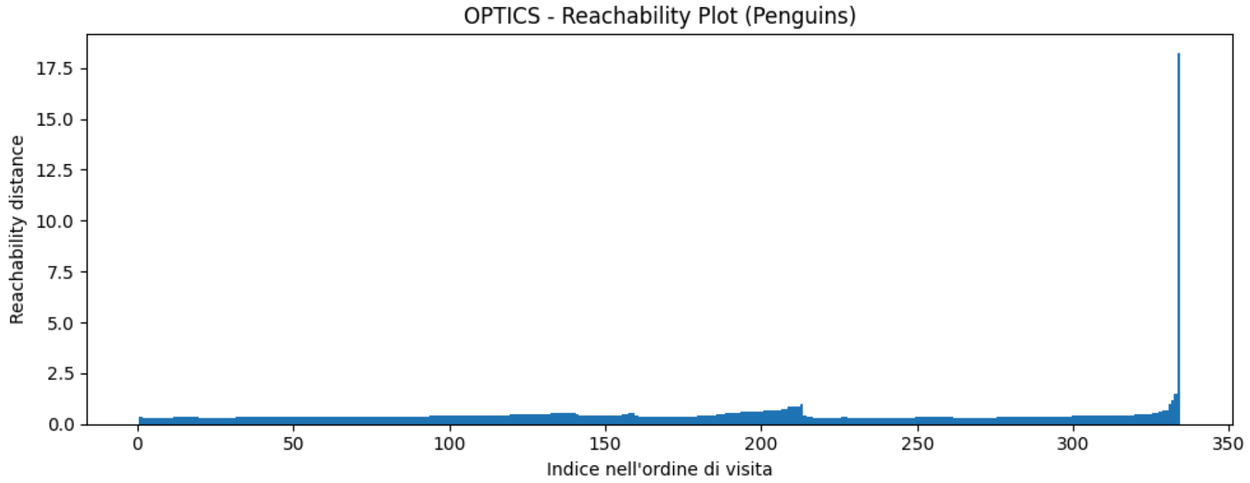
cluster	n_pinguini
-1	268
0	17
1	30
2	20

Medie per cluster:

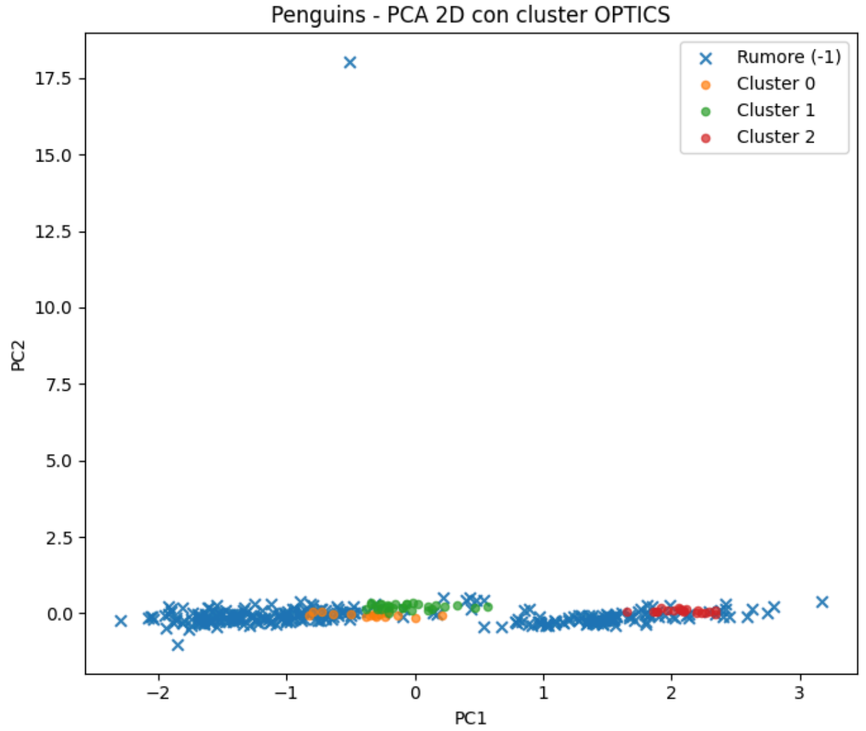
cluster	culmen_length	culmen_depth	flipper_length	body_mass
-1	42.639925	17.025373	216.873134	4183.955224
0	46.500000	17.600000	191.000000	3588.235294
1	50.753333	19.020000	199.000000	3830.833333
2	49.775000	15.960000	223.500000	5642.500000

GRAFICAMENTE...

RIMOZIONE DELLA FEATURE sex_MALE



Il nuovo reachability plot presenta **poche vallate strette** e lunghi tratti con valori di reachability elevati, segno che OPTICS **non individua più regioni estese e stabili di alta densità**.



La PCA 2D mostra **cluster meno compatti** e **più sovrapposti**.

ANALISI E INTERPRETAZIONE

RIMOZIONE DELLA FEATURE sex_MALE

L'analisi delle medie evidenzia che:

- Il **cluster 2** raggruppa pinguini con pinne molto lunghe e massa corporea elevata, interpretabili come **individui di taglia** particolarmente **grande**.
- Il **cluster 1** mostra valori alti sia di lunghezza sia di profondità del becco, indicando **pinguini con becco più sviluppato** rispetto alla media.
- Il **cluster 0,** al contrario, è caratterizzato da pinne più corte e massa inferiore, compatibili con **esemplari** complessivamente **più piccoli**.

Il fatto che la **maggior parte dei punti resti comunque classificata come rumore** suggerisce però che, in assenza della feature sul sesso, queste **differenze morfologiche non siano sufficienti a definire regioni di densità ben separate** nello spazio delle feature.

L'interpretazione è che, **rimuovendo il sesso**, lo **spazio delle feature diventa più "continuo":** maschi e femmine con misure simili finiscono vicini. A parità di parametri, OPTICS non considera più la maggior parte dei punti come appartenenti a cluster sufficientemente densi e li classifica come rumore.

DISCUSSIONE DEI RISULTATI

Importanza delle feature

L'aggiunta della sola variabile *sex_MALE* fa passare da una situazione in cui quasi tutto il dataset è considerato rumore a una in cui emergono diversi cluster ben formati. Il **sesso** risulta quindi una **feature fortemente informativa** → *I maschi sono quindi di natura più grandi delle femmine, e includendo il sesso è possibile individuare correttamente le differenze morfologiche considerando anche questa caratteristica.*

Spazio delle feature vs parametri dell'algoritmo

I parametri di OPTICS sono gli stessi in entrambi i casi; il **cambiamento** nel numero di cluster e nella quantità di rumore **dipende** dunque esclusivamente **da quali feature vengono selezionate**.

Feature poco informative = più rumore, feature più informative =cluster più puliti

Nel caso morfologico puro (senza quindi considerare il sesso), le differenze tra individui sono distribuite in modo graduale, senza stacchi netti, causando un'elevata quota di outlier. Introducendo una feature che separa gruppi relativamente omogenei (come il sesso) aumenta la densità locale e l'algoritmo riesce a identificare cluster più chiari e interpretabili.

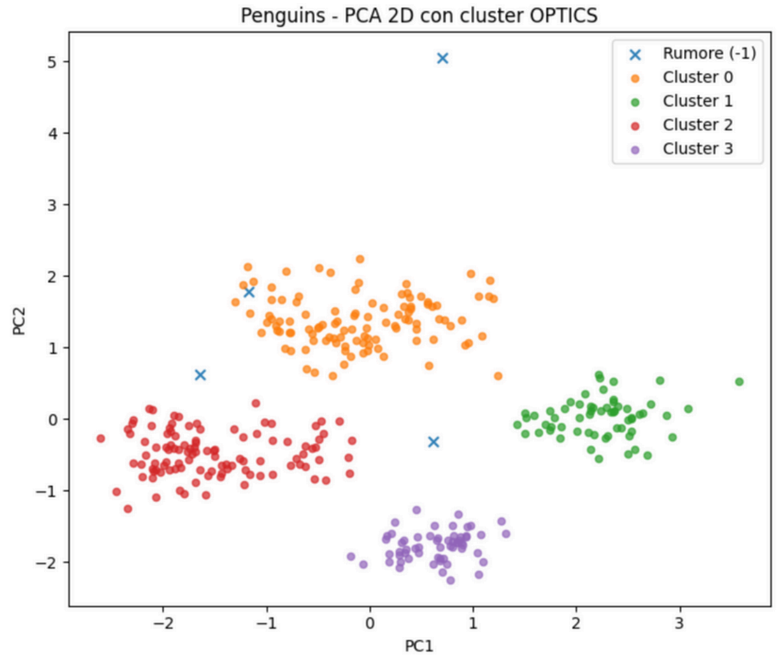
In sintesi, l'esperimento sul dataset Penguins mostra che OPTICS non "inventa" cluster dove la struttura non è supportata dai dati: se lo spazio delle feature non contiene regioni ben separate, l'algoritmo restituisce prevalentemente rumore. Quando invece le feature catturano le differenze tra sottogruppi, il reachability plot e la proiezione PCA rivelano una struttura a cluster complessa ma coerente

ALTRO ESPERIMENTO

ANALISI DI SENSIBILITÀ RISPETTO AL PARAMETRO xi

Per valutare l'effetto dei parametri di OPTICS, è stato eseguito un ulteriore esperimento variando il parametro xi, *che controlla quanto l'algoritmo è sensibile a variazioni locali di densità*.

L'esperimento è stato effettuato impostando ***xi*** al valore di 0.15, ciò che ne è risultato è stato:



Distribuzione di pinguini per cluster:

cluster	n_pinguini
-1	4
0	107
1	60
2	105
3	59

Medie per cluster:

cluster	culmen_length	culmen_depth	flipper_length	body_mass	sex_MALE
-1	43.550000	19.950000	1310.000000	4062.500000	0.5
0	43.883178	19.094393	194.990654	4018.224299	1.0
1	49.558333	15.691667	221.583333	5488.750000	1.0
2	40.054286	17.580000	189.104762	3411.904762	0.0
3	45.545763	14.262712	212.779661	4683.050847	0.0

ANALISI

ANALISI DI SENSIBILITÀ RISPETTO AL PARAMETRO xi

Aumentando ***xi*** (riducendo quindi la sensibilità) si osserva che:

- Il **numero di cluster diminuisce** (da 6 cluster + rumore a 4 cluster principali + rumore),
- Il numero di **punti etichettati come rumore crolla** (da 141 a 4 osservazioni),
- I **cluster** risultano **più estesi e compatti** nello spazio delle feature.

Nel corrispondente grafico PCA 2D i quattro cluster formano **insiemi ben separati ->** i **pochi outlier** rimangono **isolati**.
Le medie di *sex_MALE* per cluster mostrano ancora una **forte separazione** tra gruppi di soli **maschi** (valore medio 1) e gruppi di sole **femmine** (valore medio 0), indicando che la struttura legata al sesso rimane visibile anche con una scelta di xi molto meno sensibile. Questo esperimento conferma che:

- OPTICS può passare da una descrizione più fine (molti cluster + più rumore) a una più grossolana (pochi cluster grandi + quasi nessun rumore) semplicemente variando xi
- La scelta di xi non crea cluster “dal nulla”, ma **decide a che livello di dettaglio vogliamo strutturare la densità** già presente nei dati, quindi la scelta delle feature rimane fondamentale per il corretto funzionamento dell’algoritmo.

OSSERVAZIONE FINALE

RISPETTO ALLA MODIFICA DEI PARAMETRI

Sono stati effettuati diversi esperimenti variando tutti i parametri dell'algoritmo, prima con modifiche di piccola entità e poi con cambiamenti più marcati rispetto ai valori iniziali. In tutti i casi, la quantità di punti etichettati come rumore è rimasta **elevata**, a meno di ridurre la sensibilità del metodo fino a valori tali da rendere i dati quasi uniformi.

Questo indica che **il problema non dipende tanto dalla scelta dei parametri,** quanto dal fatto che i punti del dataset sono effettivamente molto dispersi nello spazio delle feature.

GRAZIE PER L'ATTENZIONE