Università degli Studi di Firenze

Scuola di Ingegneria - Dipartimento di Ingegneria dell'Informazione

---

Tesi di Laurea Triennale in Ingegneria Informatica

Analisi comparativa di Metriche per la valutazione di modelli generativi

Candidato
Rocco Tescaro

Relatore
Prof. Paolo Frasconi

Correlatori

Anno Accademico 2024

Indice

Introduzione

Negli ultimi anni, i modelli di reti neurali generative hanno registrato notevoli progressi, trovando applicazione in vari ambiti quali la creazione automatica di immagini, la sintesi musicale, la generazione di testi e molto altro. Algoritmi come le *Generative Adversarial Networks (GANs)* hanno prodotto risultati di grande rilievo, aprendo nuove prospettive sia per la ricerca scientifica che per applicazioni industriali e commerciali.

Con l'avanzamento delle capacità di generazione, è emersa la necessità di sviluppare metodi oggettivi e affidabili per la valutazione della qualità dei modelli generativi, andando oltre la soggettività dell'ispezione visiva umana, che sebbene in ultima analisi rimane comunque il metodo più affidabile e diffuso, risulta intrinsecamente non replicabile, è soggetta a variabilità tra diversi osservatori ed è difficilmente scalabile in contesti che richiedono una valutazione in larga scala.

Per rispondere a questa esigenza, inizialmente sono state proposte metriche scalari, basate quindi su singoli indici, tra cui il Frechet Inception Distance (FID), progettate per stimare la somiglianza statistica tra campioni generati e reali. Tuttavia, tali metriche hanno mostrato limiti nel descrivere con precisione proprietà diverse delle distribuzioni dei dati generati, quali la *fidelity* (la somiglianza tra i campioni generati e quelli reali) e la *diversity* (la varietà tra i campioni generati).

Per superare questi limiti, sono state introdotte metriche più complesse, capaci di fornire una valutazione più articolata della qualità dei modelli. Tra queste metriche, particolare attenzione sarà riservata all'analisi di *Improved Precision and Recall*, *Density and Coverage*, *Probabilistic Precision and Recall* e *Precision and Recall Coverage*. È importante sottolineare la corrispondenza concettuale tra i termini *fidelity* e *precision*, così come tra *diversity* e *recall*. Tuttavia, nel contesto delle metriche di precision e recall, il calcolo si basa specificamente sulla distanza tra campioni generati e reali, nonché tra ciascun campione e i suoi vicini all'interno dello spazio delle caratteristiche.

Nei capitoli che seguiranno verrà approfondita la definizione di tali metriche e le relative versioni più generali e complesse definite in tempi più recenti, vale a dire le *Precision Recall Curves*. Veranno quindi introdotti gli esperimenti condotti e i relativi risultati.

La nostra ricerca si propone di analizzare e confrontare le metriche di valutazione della qualità dei modelli generativi, vale a dire caratteristiche e limiti come la dipendenza dai diversi iperparametri, la sensibilità alle dimensioni del dataset, la resistenza agli *outliers*, la capacità di discriminare dati generati di altà qualità da quelli di bassa qualità e quindi le possibilità di filtrare i risultati ottenuti.

Capitolo 1

Metriche

Le protagoniste della nostra analisi sono, come anticipato nell'introduzione: **Improved Precision and Recall**, **Density and Coverage**, **Probabilistic Precision and Recall** e **Precision and Recall Coverage**. Saranno poi introdotte tecniche più recenti come la **Precision Recall Curve** e altre metriche correlate.

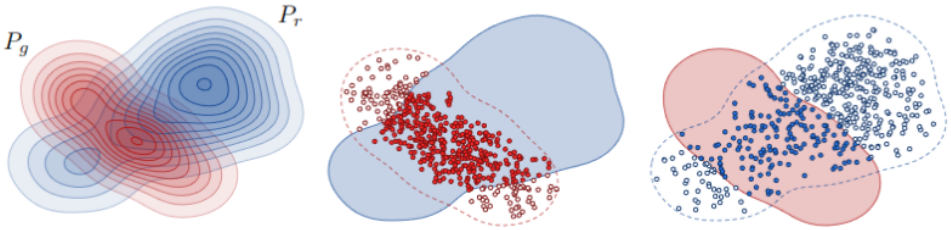
Tutte le metriche citate hanno una serie di caratteristiche in comune, il primo, e forse più ovvio, è che sono presentate in coppia. Questo perchè queste coppie di metriche cercano misurare due aspetti complementari delle reti generative: la **Fidelity** e la **Diversity**.

Il secondo aspetto che hanno in comune, che verrà discusso e approfondito invece nelle sezioni seguenti, è relativo a come queste metriche sono misurate. Ciascuna di esse infatti si basa su un calcolo simile dipendente dalla distanza tra campioni generati e reali e la distanza **interset** vale a dire fra campioni vicini di uno stesso insieme.

È utile definire meglio i concetti appena accennati:

- **Fidelity**: è la somiglianza tra i campioni generati e quelli reali.

- **Diversity**: è la varietà tra i campioni generati.
- **Precision**: misura la proporzione di campioni generati che ricadono nel supporto della distribuzione reale.
- **Recall**: misura la proporzione di campioni reali che ricadono nel supporto della distribuzione generata.



(a) Example distributions (b) Precision (c) Recall

Sebbene *fidelity* e *diversity* siano concetti centrali per la valutazione della qualità dei modelli generativi, risultano difficili da quantificare direttamente a causa della loro natura astratta. Si riduce quindi il problema del calcolo della *fidelity* e *diversity* a quello della *precision* e *recall* (che offrono una formalizzazione più operativa di questi concetti, fornendo una base matematica per una valutazione quantitativa).

Entrambe si basano sul concetto di **supporto** della distribuzione. Dato un numero limitato di campioni, definire un supporto continuo è un problema non banale. Molte delle metriche si preoccupano quindi di stimare il supporto e stabilire un criterio operativo per determinare la somiglianza tra due distribuzioni, il grado di sovrapposizione dei supporti stimati. In questo contesto la stima del supporto è detta **manifold**.

1.1 Improved Precision and Recall

Nonostante il nome *Improved Precision and Recall* possa suggerire un miglioramento rispetto a una versione precedente e meno sofisticata di *precision and recall*, la metrica proposta da Tuomas Kynkäänniemi et al., 2019 [8], precede temporalmente, in realtà, quella che sarà presentata successivamente, ovvero la metrica denominata più semplicemente *Precision and Recall Coverage*. Di fatto, si tratta della forma più semplice di *precision-recall* analizzata in questo lavoro, e rappresenta il punto di partenza per la nostra analisi.

Sia R il dataset di campioni reali (in genere dati usati per l'addestramento della rete) e G il dataset di campioni generati dalla rete neurale, identifichiamo con $\boldsymbol{\Phi_R}$ e $\boldsymbol{\Phi_G}$ (tendenzialmente con $\boldsymbol{\Phi_R} = \boldsymbol{\Phi_G}$) lo spazio delle caratteristiche e Φ_R e Φ_G i vettori di feature estratti da R e G rispettivamente.

È importante sottolineare che, come vedremo anche successivamente nella parte sperimentale, la scelta dell'estrattore di feature è cruciale per la corretta valutazione delle metriche di valutazione dei GAN.

Il manifold stimato a partire da R e G è un insieme di ipersfere, ciascuna con un raggio definito e centrate nei vari punti di Φ_R e Φ_G. Il raggio di queste ipersfere è determinato da un iperparametro `k` che rappresenta il numero dei **nearest neighbors (NN)** da considerare.

Possiamo quindi definire una funzione binaria che determini se un certo punto appartiene al manifold o meno:

$$f_{ipr}(x, \Phi) = \begin{cases} 1 & \text{if} \quad \exists\, y \in \Phi \text{ such that } ||x - y||_2 \leq ||y - NN_k(y, \Phi)||_2 \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

Dove $NN_k(y, \Phi)$ è il `k`-esimo vicino più prossimo di `y` in Φ.

La Precision e la Recall possono essere quindi definite come:

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|} \sum_{g \in \Phi_G} f_{ipr}(g, \Phi_R) \tag{1.2}$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} f_{ipr}(r, \Phi_G) \tag{1.3}$$

Nominalmente la Precision misura la percentuale di punti di Φ_G che ricadono nel manifold di Φ_R e la Recall la percentuale dei punti di Φ_R che ricadono nel manifold di Φ_G. Come tali le due metriche sono quindi normalizzate e assumono valori compresi tra `0` e `1`. Come ripeteremo in seguito la metrica risulta dipendente dall'iperparametro `k` e in particolare pare che per `k = 3` si ottengano i risultati migliori.

1.2 Density and Coverage

Come vedremo nel capitolo seguente l'*Improved Precision Recall* soffre la presenza di *outliers* nei datasets, vale a dire che il manifold stimato da R e G può essere fortemente influenzato da pochi punti molto distanti dalla distribuzione principale e quindi la presenza di molti dati generati in una zona vicina a questi outliers può portare a un errata alta stima della *fidelity-precision*. La *Density and Coverage* è una metrica che cerca di mitigare questo problema (come analizzato da Muhammad Ferjad Naeem et al., 2019 [7]).

Invece che contare se le varie ipersfere in Φ_R contengono almeno un punto di Φ_G, la *Density* conta quanti punti di Φ_G sono inclusi in ciascuna ipersfera centrata su Φ_R. Viene quindi scalato il risultato per il numero di ipersfere totali per il valore di `k` (si suppone che ciascuna ipersfera contenga almeno `k` punti). Questo conporta però che la *density* non è una metrica normalizzata

e in certi casi assume valori maggiori di `1`. La formula è la seguente:

$$Density(\Phi_R, \Phi_G) = \frac{1}{k|\Phi_G|} \sum_{g \in \Phi_G} \sum_{r \in \Phi_R} \mathbb{1}_{||r-g||_2 \leq ||g-NN_k(g,\Phi_G)||_2} \tag{1.4}$$

Rispetto alle altre metriche in cui si poteva apprezzare la simmetria fra le formule per la stima di *fidelity* e *diversity*, in questo caso la *Coverage* è una metrica completamente diversa dalla *Density*. Questo perchè rispetto a Φ_G, Φ_R non dovrebbe presentare *outliers*. La Coverage valuta se ogni dato reale $r \in \Phi_R$ è rappresentato da almeno un punto generato $g \in \Phi_G$, costruendo il manifold in Φ_R e verificando se esiste almeno un punto di Φ_G in ciascuna ipersfera.

Anticipando la sezione che seguirà *Precision Recall Curve*, è utile definire una funzione binaria che determini per la *Coverage* se un certo punto appartiene a questa nuova definzione di manifold:

$$f_{cov}(x, \Phi_x, \Phi_y) = \begin{cases} 1 & \text{if} \quad \exists\ y \in \Phi_y \text{ such that } ||x-y||_2 \leq ||x - NN_k(x, \Phi_x)||_2 \\ 0 & \text{otherwise.} \end{cases} \tag{1.5}$$

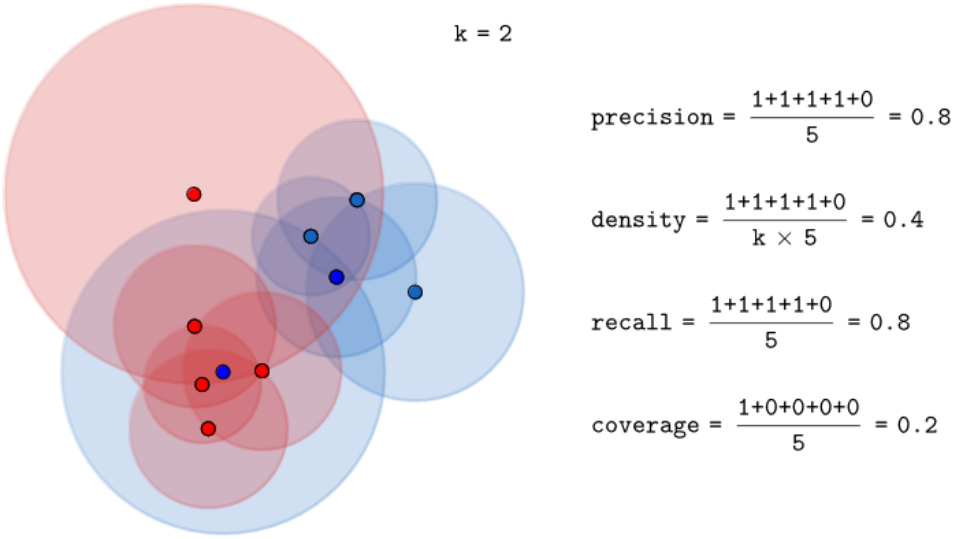
La formula della metrica sarà quindi:

$$Coverage(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} f_{cov}(r, \Phi_R, \Phi_G) \tag{1.6}$$

La formula potrebbe sembrare molto simile a quella dell'Improved Precision ma con un'importante differenza, ci chiediamo se esiste almeno un punto di Φ_G per ciascuna ipersfera costruita su Φ_R e non se per ciascun punto di Φ_G esiste una ipersfera che lo contiene.

Al contrario della Density, la Coverage è una metrica normalizzata e assume valori compresi tra 0 e 1.

Sperimentalmente, si è verificato che un valore di `k = 5` fornisce risultati ottimali, bilanciando la granularità delle ipersfere e la loro rappresentatività del manifold.



1.3 Probabilistic Precision and Recall

Un altro approccio al problema della stima del manifold consiste nel considerare la densità di probabilità delle distribuzioni reali e generate. Le formule utilizzate sono simili a quelle dell'*Improved Precision e Recall*, ma con una differenza cruciale: invece di utilizzare una funzione di scoring binaria (f_{ipr}), si valuta la probabilità che un punto appartenga al manifold stimato. Questa probabilità viene calcolata dividendo il supporto in sottosupporti locali (*subSupports*). Seguendo il lavoro di Dogyun Park et al., 2023 [2], la probabilità di un punto x di appartenere a un sottosupporto centrato su y è definita come:

$$P(x \in subSupp(y)) = \begin{cases} 1 - \frac{||x-y||_2}{\tau} & \text{se } ||x-y||_2 \leq \rho \\ 0 & \text{altrimenti.} \end{cases} \tag{1.7}$$

La probabilità che un punto appartenga al manifold stimato viene quindi calcolata calcolando la probabilità che il dato non appartenga all'intersezione fra tutti i complementari dei sottosupporti locali (supponendo quindi

l'indipendenza degli eventi $x \in subSupp^C(y)\ \forall y$):

$$P(x \in manifold(\Phi_y))) = 1 - \prod_{y \in \Phi_y} (1 - P(x \in subSupp(y))) \tag{1.8}$$

Per la precision, la somma itera x su Φ_G con manifold calcolato su Φ_R, mentre per la recall, si inverte il ruolo dei due dataset:

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|} \sum_{g \in \Phi_G} P(g \in manifold(\Phi_R)), \tag{1.9}$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} P(r \in manifold(\Phi_G)). \tag{1.10}$$

La variabile ρ gioca un ruolo cruciale in questo approccio, rappresentando il raggio delle ipersfere usate per stimare i sottosupporti locali. Per evitare sovrastime, si adotta una strategia simile alla *Kernel Density Estimation* con varianza fissata, definendo ρ come segue:

$$\rho(\Phi) = \alpha \cdot \frac{1}{|\Phi|} \sum_{x \in \Phi} ||x - NN_k(x, \Phi)||_2 \tag{1.11}$$

dove α è un nuovo iperparametro (con valore consigliato $\alpha = 1.2$). Per la precision si adotta quindi $\rho(\Phi_R)$, mentre per la recall si utilizza $\rho(\Phi_G)$.

L'approccio probabilistico è particolarmente utile perché affronta i limiti delle metriche basate sul `k`-Nearest Neighbor. La scelta di una ρ costante pari alla media delle distanze k-NN sopprime l'influenza degli outliers, riducendo la sovrastima del manifold in regioni scarsamente popolate e rende la metrica più robusta alla scelta di `k`, minimizzando la sensibilità della metrica ai cambiamenti del numero di vicini considerati.

Sempre seguendo la letteratura il valore di `k` consigliato è `k = 4`.

1.4 Precision and Recall Coverage

L'ultima metrica che verrà presentata è la *Precision and Recall Coverage*, proposta da Fasil Cheema et al., 2023 [3].

Mantenendo la formalizzazione della sezione precedente si introduce un nuovo iperparametro `k' = Ck`. L'idea è quella di fornire un nuovo elemento per regolare la dimensione del manifold, e in particolare le aree da considerare trascurabilmente piccole e quelle sufficientemente grandi. L'algoritmo proposto prevede per la Precision di costruire un manifold su Φ_G con ipersfere con raggio pari alla distanza dal `k'`-th nearest neighbor e contare il numero di ipersfere che contengono almeno `k` punti di Φ_R, si divide quindi per il numero di ipersfere totali (la dimensione di Φ_G). Per la Recall si procede in modo analogo ma invertendo i ruoli di Φ_R e Φ_G.

È importante notare come la metrica sia fondamentalmente la stessa della Coverage ma non è più sufficiente che esista un unico punto di Φ_R in un ipersfera di Φ_G ma è necessario invece che ce ne siano almeno `k`.

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|} \sum_{g \in \Phi_G} \mathbb{1}_{|\{r \in \Phi_R, \text{s.t.} ||r-g||_2 \leq ||r - NN_{k'}(r, \Phi_R)||_2\}| \geq k} \quad (1.12)$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} \mathbb{1}_{|\{g \in \Phi_G, \text{s.t.} ||g-r||_2 \leq ||g - NN_{k'}(g, \Phi_G)||_2\}| \geq k} \quad (1.13)$$

In questo caso l'aggiunta di un nuovo parametro, nonostante aumenti la complessità combinatoria di possibili valori assegnabili, si verifica sperimentalmente che mantenendo `C=3` la scelta di `k` e conseguentemente di `k'` può essere arbitraria ottenendo comunque buoni risultati.

1.5 Precision Recall Curve

Studi recenti [1, 5, 6] hanno dimostrato che precision e recall possono essere interpretate come combinazioni lineari degli errori di tipo I e II di un classificatore binario ottimo. Questa formulazione consente di costruire un limite superiore per la Precision e la Recall anche utilizzando un classificatore binario sub-ottimale. Inoltre, permette di analizzare il trade-off tra

Precision e Recall tramite una curva, di cui le metriche viste in precedenza rappresentano solo i valori estremi.

I classificatori per la costruzione della curva Precision-Recall si basano sui manifold stimati. Le formule per i classificatori associati alle metriche sono:

$$f_\lambda^{ipr}(x) = \mathbb{1}_{\lambda \cdot |\{r \in \Phi_R, \text{s.t.} ||r-x||_2 \leq ||r-NN_k(r,\Phi_R)||_2\}| \geq |\{g \in \Phi_G, \text{s.t.} ||g-x||_2 \leq ||g-NN_k(g,\Phi_G)||_2\}|} \tag{1.14}$$

$$f_\lambda^{cov}(x) = \mathbb{1}_{\lambda \cdot |\{r \in \Phi_R, \text{s.t.} ||r-x||_2 \leq ||r-NN_k(x,\Phi_G)||_2\}| \geq |\{g \in \Phi_G, \text{s.t.} ||g-x||_2 \leq ||g-NN_k(x,\Phi_R)||_2\}|} \tag{1.15}$$

È interessante notare come le condizioni di appartenenza agli insiemi la cui cardinalità viene confrontata siano analoghe alle condizioni per f_{ipr} e f_{cov} con la differenza che qui andiamo a contare il numero di punti che soddisfano la condizione e non se esiste almeno un punto che la soddisfi.

La formula per f_λ^{ipr} può essere interpretata come una forma di Kernel Density Estimation (KDE) con bandwidth variabile.

$$f_\lambda^{ipr}(x) = \mathbb{1}_{\frac{\hat{p}(x)}{\hat{q}(x)} \geq \frac{1}{\lambda}}, \tag{1.16}$$

dove $\hat{p}(x)$ e $\hat{q}(x)$ rappresentano le densità stimate dai manifold reali e generati rispettivamente. In questa formulazione, il parametro λ agisce come un fattore di soglia che modula la decisione del classificatore:

- $\lambda \to 0$: Include quasi tutti i punti, abbassando la precisione e massimizzando la recall.

- $\lambda \to \infty$: Include solo i punti con densità generata molto bassa, massimizzando la precisione ma riducendo la recall.

Dal momento che questi classificatori vanno a cotruire un limite superiore, è possibile prendere il minimo fra le curve ottenute con i diversi classificatori per ottenere una stima più precisa.

1.6 Altre metriche e funzioni correlate

Per aggiungere un'appendice alla sezione precedente, sono state brevemente sfruttate altre metriche/classificatori accennati nel medesimi paper e in particolare nell'articolo di Benjamin Sykes et al, 2024 [1]. Tra questi risulta il knn classifier e il parzen classifier.

$$f_\lambda^{knn}(x) = \mathbb{1}_{\lambda|\{r\in\Phi_R,\text{s.t.}||r-x||_2\leq||r-NN_k(x,\Phi_U)||_2\}|\geq|\{g\in\Phi_G,\text{s.t.}||g-x||_2\leq||g-NN_k(x,\Phi_U)||_2\}|} \tag{1.17}$$

$$f_\lambda^{parzen}(x) = \mathbb{1}_{\lambda|\{r\in\Phi_R,\text{s.t.}||r-x||_2\leq\rho(\Phi_G)\}|\geq|\{g\in\Phi_G,\text{s.t.}||g-x||_2\leq\rho(\Phi_R)\}|} \tag{1.18}$$

con $\Phi_U = \Phi_G \cup \Phi_R$ e $\rho(\Phi)$ definito come nella sezione relativa a *Probabilistic Precision and Recall* ($\alpha = 1$), vale a dire che il raggio delle ipersfere è fissato alla media delle distanze `k`-NN.

f_λ^{knn} è fondamentalmente analogo al f_λ^{cov} classifier ma costruisce l'ipersfera su entrambi i dataset. Il f_λ^{parzen} classifier invece nutre delle somiglianze con il f_λ^{ipr} classifier ma costruisce l'ipersfera con un raggio fisso, è quindi per transitività simile ad una KDE con bandwidth costante.

Sebbene non sia propriamente una metrica di valutazione per i GAN, la Kernel Density Estimation è un metodo di stima della densità di probabilità di un dataset, ed è stata utilizzata nei nostri esperimenti per osservare alcune caratteristiche dei dataset reali e generati. In particolare sono state sfruttate l'approssimazione di **Silverman** e nelle fasi di testing anche la stima di **Scott**.

La stima della densità tramite Kernel Density Estimation (KDE) con un kernel gaussiano può essere espressa come:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{1.19}$$

dove $K(x)$ è, nel nostro caso, il kernel gaussiano definito come:

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \tag{1.20}$$

h è il parametro di bandwidth e n rappresenta il numero di punti nel datases.

$$h_{\text{Scott}} = n^{-\frac{1}{d+4}} \cdot \sigma \tag{1.21}$$

$$h_{\text{Silverman}} = \left(\frac{4}{3n}\right)^{\frac{1}{5}} \cdot \sigma \tag{1.22}$$

dove σ è la deviazione standard del dataset e d è la dimensione dei dati.

Un ultima funzione che è stata utilizzata per valutare la qualità dei singoli dati è il **realism score** proposto da [8] e definito come:

$$\text{realism score}(g) = \max_{r \in \Phi_R} \{ \frac{||r - NN_k(r)||_2}{||g - r||_2} \} \tag{1.23}$$

Un punto che ha realism score ≥ 1 è molto simile ai dati reali, viceversa un punto con realism score in [0,1) è molto diverso dai dati reali.

Per quanto riguarda l'applicazione dei classificatori per la costruzione della Precision Recall Curve, i dataset sono stati divisi in training e testing set con split specificati nella parte sperimentale. Mentre per gli estrattori di caratteristiche sono state utilizzate metriche **informate** e **non informate** del dominio dei dati (per queste chiariremo meglio in seguito).

Capitolo 2

Esperimenti

Per la valutazione delle metriche sono stati condotti diversi esperimenti che possono essere suddivisi in due macro-categorie dipendentemente dal tipo di dataset utilizzato:

- **Toy Dataset**: dataset generati artificialmente per testare il comportamento delle metriche in condizioni controllate.
- **Real World Dataset**: dataset reali per testare il comportamento delle metriche in condizioni reali.

Le sezioni seguenti descrivono come sono state implementate le metriche, gli esperimenti così come le distribuzioni di dati utilizzate. La ragione per cui sono stati condotti esperimenti su dateset generati artificilmente (ovvero di matrice matematica, non derivanti dalla realtà o generati da reti neurali) è che in questo modo è stato possibile osservare il comportamento delle metriche in condizioni controllate, ideali e per poter confrontare i risultati ottenuti con quelli attesi presenti nella letteratura. Fanno infatti parte dei test su 'toy dataset', i test di corretta implementazione ovvero un'analisi comparativa delle diverse implementazioni in codice delle metriche. Tali test hanno il fine

di verificare che le metriche restituiscano valori corretti validando gli altri esperimenti.

I dataset reali sono stati utilizzati per testare il comportamento delle metriche in condizioni reali, ovvero per verificare se le metriche si comportano come ci si aspetta in situazioni non ben definite, con distribuzioni di dati non note, ben distanti da quelle ideali. Questo infatti, come vedremo, può sollevare criticità, ad esempio dovute alla assenza di certe categorie di dati o alla presenza di dati non ben distribuiti. Un altro elemento dei dataset reali è che spesso questi non sono di carattere prettamente numerico, nei dataset analizzati ad esempio, abbiamo immagini di farfalle e partiture musicali. Questo comporta la necessità di trasformare i dati per poterli utilizzare, e il processo di **estrazioni di caratteristiche** può equivalere ad una perdita di dati significativi o, al contrario, ad una sovrabbondanza di dati non significativi.

2.1 Toy Dataset

Come anticipato nell'introduzione di questo capitolo, i Toy Dataset sono dataset generati artificialmente tramite funzioni matematiche che producono dati in modo casuale ma controllato. Per questi esperimenti, sono state utilizzate principalmente due tipo di distribuzioni di dati numerici con dimensione variabile: la distribuzione uniforme e la distribuzione normale. In una specifica categoria di esperimenti, sono stati aggiunti anche outliers, ossia dati che si discostano significativamente dalla distribuzione principale, con l'obiettivo di valutare la robustezza delle metriche in presenza di anomalie.

Gli esperimenti condotti si ispirano a quelli presenti in letteratura e hanno diversi obiettivi, tra cui:

- Testare l'influenza degli iperparametri delle metriche sul loro comportamento.
- Valutare la risposta delle metriche alla presenza di outliers.
- Studiare l'impatto della dimensione del dataset sui risultati.
- Confrontare le implementazioni delle metriche esistenti, sia in termini di valori scalari che tramite curve di precision-recall (PR-curve).

Per ciascun esperimento sono stati prodotti grafici che illustrano i risultati ottenuti. L'unica eccezione è rappresentata dai test di corretta implementazione delle metriche scalari, dove la validazione è stata condotta principalmente tramite confronti numerici. La scelta delle tipologie di grafici è stata guidata dall'obiettivo di facilitare il confronto con i risultati riportati in letteratura.

Data la complessità computazionale di alcuni esperimenti, i risultati intermedi e finali sono stati salvati in file `.npy`, permettendo analisi approfondite e riproducibilità senza dover ripetere calcoli onerosi. Questo approccio non solo consente una gestione efficiente dei dati, ma permette anche un'analisi successiva più flessibile, ad esempio per esplorare ulteriori correlazioni o per verificare ipotesi aggiuntive.

2.1.1 Parametro k e dimensione del dataset

Come abbiamo visto, tutte le metriche analizzate si basano sulla distanza dei dati rispetto ai loro vicini. Uno dei parametri più determinanti è l'ordine `k` del vicino più prossimo. Secondo la letteratura, i valori ottimali di `k` variano in base alla metrica analizzata: per l'**improved precision recall** `k = 3`, per la **probabilistic precision recall** `k = 4`, per la **precision recall coverage**

`k`=$\sqrt{|\Phi|}$ indicando con $|\Phi|$ la dimensione del dataset reale e/o generato (o `k = 3` se `C = 3`) , mentre per **density** e **coverage** `k = 5`. Questi valori riflettono un compromesso ottimale tra stabilità della metrica e sensibilità alla densità locale. Ci si aspetta che l'aumento della dimensione del dataset porti a un incremento dei valori delle metriche, poiché una maggiore quantità di dati aumenta la densità dei punti, migliorando la rappresentatività delle distribuzioni e riducendo l'effetto del rumore.

L'analisi è stata condotta su dataset generati una sola volta, mantenuti costanti e identici per la valutazione di tutte le metriche, questo per garantire coerenza nei risultati. Sono state utilizzate, come anticipato precedentemente, due diverse distribuzioni: uniforme e normale. I test sono stati effettuati per valori di `k` variabili da 1 a 10 vale a dire $[1, 2, 3, \ldots, 8, 9, 10]$ e per dimensioni del dataset crescenti esponenzialmente, da 500 a 16000 punti ($[500, 1000, 2000, 4000, 8000, 16000]$). Ogni dato è rappresentato come un vettore in $\mathbb{R}^{64}$.

Per presentare i risultati, sono state utilizzate delle **heatmap**, che mostrano i valori delle metriche in funzione di `k` e della dimensione del dataset. In queste rappresentazioni, il rosso indica valori prossimi a `1.`, mentre il verde valori vicini a `0.`. Sebbene le heatmap non offrano precisione numerica immediata, forniscono una visione d'insieme sulle tendenze generali delle metriche e facilitano il confronto con gli esperimenti presenti in letteratura [7]. Nel paper di riferimento, sono state prodotte heatmap solo per le metriche di **improved precision recall** e **density and coverage**, ma per completezza sono state prodotte anche per le altre metriche presentate nel capitolo precedente.

2.1.2 Dimensione del dataset e dimensione dei dati

In questo caso, l'analisi condotta non ha un riscontro diretto nella letteratura esistente, ovvero non presenta antecedenti (quantomeno per gli articoli presi in analisi). L'obiettivo è determinare come la **dimensione del dataset** possa influenzare la densità dei dati della distribuzione (a dimensione dei dati fissata) e, conseguentemente, il valore delle metriche. I risultati di questa analisi costituiscono una parte fondamentale per le analisi su dati reali, dove la scelta del numero di caratteristiche da considerare può risultare determinante.

In questo esperimento, abbiamo considerato una distribuzione normale ($\mathcal{N}(0, I)$) dei dati con **dimensione del dataset variabile** da 50 a 1600, con valori $[50, 100, 200, 400, 800, 1600]$, e **dimensione dei dati** da 2 a 64, con valori $[2, 4, 8, 16, 32, 64]$. A differenza degli esperimenti precedenti, rappresentati tramite heatmap, qui l'assenza di un riscontro nella letteratura ci ha permesso di utilizzare **grafici a linee bidimensionali** per rappresentare i risultati (data la ridotta dimensionalità, almeno per le dimensioni di interesse, di uno degli iperparametri da regolare, ovvero la dimensione dei dati). Il valore dell'**iperparametro** k è stato scelto in accordo con quanto suggerito nei vari articoli, per garantire la massima efficacia della metrica. Le misurazioni sono state ripetute 25 volte e successivamente mediate. Anche in questo caso, il calcolo è stato effettuato in parallelo.

2.1.3 Outliers

Una delle proprietà più rilevanti da esaminare nelle diverse metriche è la loro **resistenza agli outliers**. In linea con la letteratura, abbiamo analizzato come i valori delle metriche cambiano in presenza di dataset con distribuzione

normale, **senza outliers** e con **outliers** inseriti sia nei dati reali sia in quelli generati.

L'esperimento si è svolto considerando una distribuzione reale fissa $X \sim \mathcal{N}(0, I)$ e una distribuzione generata $Y \sim \mathcal{N}(\mu, I)$, con uno shift della media μ variabile in $[-1, 1]$ (con **step** di `0.05`). In aggiunta, sono stati esaminati due scenari di outliers, in cui un campione estremo a $x = +1$ è stato aggiunto ai dati reali o a quelli generati. Lo spazio di lavoro è stato definito in $\mathbb{R}^{64}$, con vettori reali centrati sull'origine e campioni generati con media variabile lungo la direzione del vettore unitario. Come per gli altri esperimenti condotti, le operazioni sono state svolte in **parallelo**.

Oltre alle metriche di **precision-recall** e **density-coverage** (come nel paper), i test sono stati condotti anche sulle metriche di **probabilistic precision-recall** e **improved precision-recall**. Per ciascuna metrica sono stati utilizzati i valori degli iperparametri suggeriti dai rispettivi articoli.

In assenza di outliers, ci si attende che i valori delle metriche diminuiscano gradualmente man mano che μ si allontana da zero, indicando correttamente la divergenza tra le due distribuzioni.

2.1.4 Comparazione con implementazioni esistenti

Non tutti i papers analizzati presentavano un'implementazione delle metriche in codice. Sono stati svolti dei test confrontando su dataset identici le diverse implementazioni delle metriche presenti in letteratura. Non sono stati possibili confronti diretti per quanto riguarda la **precision-recall coverage**, in quanto non sono state trovate implementazioni in codice, mentre per la **improved precision-recall** sono state confrontate due diverse implementazioni. Sono state scelte tre diverse distribuzioni di dati: distribuzione uniforme, distribuzione normale e distribuzione normale con media in

$3/\sqrt{dim}$. Ciascuna distribuzione è stata generata con dimensione del dataset pari a 10000 e dimensione dei dati pari a 64. I risultati sono stati riportati su un file `log.txt` per poter essere confrontati in un secondo momento. Allo scopo di velocizzare la computazione e ridurre il tempo di esecuzione, i test comparativi per la **probabilistic precision-recall** e la **density-coverage** sono stati eseguiti con ordine $k = 3$, nonostante i valori ottimali suggeriti dalla letteratura siano diversi. Questo hai infatti permesso di calcolare le distanze intraset una sola volta, evitando di ripeterle per ogni valore di k e dato che non eravamo interessati a valutare l'efficacia delle metriche, quanto confrontare le diverse implementazioni. TODO Aggiungere bibliografia per le implementazioni in codice delle metriche.

2.1.5 Riproduzione delle pr-curve

Anche per la riproduzione delle pr-curve sono stati utilizzati dataset generati artificialmente. L'articolo di riferimento per questo esperimento [**?**], non presentava un'implementazione in codice delle metriche, ma solo i risultati ottenuti. Abbiamo quindi replicato le pr-curve per i quattro classificatori presentati nel paper, vale a dire il classificatore **ipr**, **coverage**, **knn** e **parzen**, per due differenti distribuzioni di dati, in particolare distribuzioni normali con media in 0 per i dati reali e $1/\sqrt{dim}$ e $3/\sqrt{dim}$ per i dati generati ($dim = 64$). I classificatori hanno operato su un dataset di 20000 elementi, con 10000 elementi per ciascuna classe. Per gli esperimenti condotti i dati sono stati divisi in training e test set sia operando uno split del 50% (ovvero 5000 punti effettivi per classe) sia senza split. Sono stati inoltre scelti due valori di k ovvero $k = 4$ e $k = \sqrt{n}$ (dove n è il numero di punti nel training set). Osservando i risultati del paper ci si aspetta che delle quattro pr-curve generate, la coverage-curve sia la più estrema, ovvero quella che produce un

risultato migliore (più vicino al classificatore ottimale). Un'altra proprietà attesa è la simmetria delle curve rispetto alla diagonale, questo è dovuto al tipo di distribuzione dei dati utilizzata e al fatto che i training set fossero bilanciati. Dei test preliminari hanno poi mostrato fondamentale la scelta del range di valori e degli step per quanto riguarda la variabile λ (ovvero il parametro che regola la trade-off tra precision e recall). Come consigliato da [6] , il range di valori è stato generato dalla formula $\tan(\pi/2 * i/(g+1))$ con $i \in [1, g]$ e $g = 1001$ il numero di valori generati. Questa trasformazione consente di esplorare diverse scale di λ con una densità variabile: i valori crescono rapidamente da 0 a 1, variano lentamente vicino a $\pi/2$, e infine aumentano rapidamente verso l'infinito. Questa caratteristica rende la funzione adatta per analizzare con precisione le transizioni critiche della PR-curve in regioni chiave, bilanciando una copertura fine e una rapida esplorazione delle estremità. In fase sperimentale sono state utilizzate altre funzioni per coprire il range di valori di λ, ma la funzione sopra descritta è risultata la più adatta per l'analisi delle curve, e quella che ha prodotto i risultati più simili a quelli presenti in letteratura.

2.2 Real World Dataset

Come anticipato nell'introduzione di questo capitolo, oltre agli esperimenti condotti in ambienti controllati, regolati e basati su dati sintetici, è fondamentale analizzare il comportamento delle metriche in condizioni reali, ovvero su dataset rappresentativi di problemi pratici. Questa fase di sperimentazione consente di testare l'applicabilità delle metriche in contesti che vanno oltre l'ambito strettamente numerico e teorico, avvicinandosi alle condizioni operative in cui tali strumenti dovrebbero operare. In particolare, l'o-

biettivo finale delle metriche studiate è proprio quello di fornire un supporto concreto nell'analisi della qualità dei dati generati, facilitando l'integrazione delle reti generative in applicazioni pratiche.

Gli esperimenti sui dati reali sono stati condotti su due dataset distinti: un set di immagini raffiguranti farfalle e una collezione di partiture musicali di Alessandro Scarlatti, compositore rappresentativo della musica barocca. Questi dataset presentano specificità intrinseche che richiedono l'estrazione di caratteristiche rilevanti dal dominio dei dati (in particolare per le immagini utilizzare i `raw data` sarebbe improponibile data la loro dimensione). Per le immagini delle farfalle si è scelto di lavorare con feature basilari, come istogrammi di colore e saturazione, per valutare l'abilità delle metriche nel rilevare differenze qualitative senza fare ricorso a rappresentazioni complesse o specifiche del dominio. Nel caso delle partiture musicali, invece, le caratteristiche estratte sono state più mirate e informate dal dominio della musica barocca seguendo quanto descritto nella letteratura [4]. Sono state utilizzate, ad esempio, informazioni di carattere ritmico e tonali. Questo approccio permette di valutare le metriche su dati complessi con maggiore precisione.

Un ulteriore strumento di analisi utilizzato in questo contesto è la **Kernel Density Estimation** (**KDE**), che si è dimostrata particolarmente utile per ottenere una stima non parametrica della distribuzione dei dati. La **KDE** permette di visualizzare come i dati siano distribuiti nel loro spazio delle caratteristiche, fornendo così un quadro più completo delle relazioni tra i campioni reali e quelli generati. Questa informazione è cruciale per interpretare meglio il comportamento delle **metriche**, soprattutto quando si cerca di identificare regioni di alta o bassa densità che potrebbero indicare rispettivamente dati generati di alta qualità o outlier.

Infine, un obiettivo centrale di questa fase sperimentale è quello di veri-

ficare l'efficacia delle metriche nel discriminare dati generati di alta qualità da dati generati di bassa qualità, fungendo così da filtro ultimo per le reti generative. In questa ottica, le metriche potrebbero operare come strumento di selezione, scartando i dati che non soddisfano determinati standard di qualità e potenzialmente indicando i campioni da rigenerare.

2.2.1 Butterflies

Gli esperimenti condotti sul dataset di immagini di farfalle si sono basati su un'analisi semplice ma efficace delle caratteristiche visive, sfruttando estrattori di caratteristiche basati su istogrammi. In particolare, per ogni immagine sono stati calcolati sei tipi di istogrammi:

- **hue histogram**
- **saturation histogram**
- **value histogram**
- **grayscale histogram** (diverso dal value histogram, in questo caso abbiamo una combinazione lineare dei valori RGB)
- **rgb histogram**
- **hsv histogram**

Ogni istogramma è stato generato utilizzando 256 **bin** (per un totale di 256*3 bin per le rappresentazioni **RGB** e **HSV**), con l'obiettivo di catturare le distribuzioni dei valori cromatici e di intensità nelle immagini.

Per la rilevazione dei **falsi positivi**, è stato utilizzato un classificatore analogo a quello impiegato negli esperimenti sulle **IPR-curve** (in questo caso però sono stati considerati tutti e soli quei dati generati che ricadevano

nel manifold dei dati reali, quindi senza una variazione del parametro λ). In questo caso, le differenze tra istogrammi sono state misurate utilizzando sia la norma l_1 che la norma l_2 come funzioni di distanza. Il classificatore ha operato su un **k-nearest neighbors** (**k-NN**) con $k = 3$ e $k = \sqrt{n}$, dove n è il numero di punti nel training set. Per valutare le prestazioni del classificatore, gli esperimenti sono stati condotti utilizzando diversi schemi di divisione dei dati: uno split 80-20 per il **training** e il **test**, e un approccio senza divisione, in cui tutti i dati venivano considerati come parte di un unico set per la classificazione.

Il dataset di farfalle usato per l'allenamento della rete generativa e quindi comprendente dati reali era composto da 1000 immagini, mentre il dataset generato era composto da 895 immagini. Data la natura delle metriche utilizzate (alcune di esse richiedevano che la dimensione dei due dataset fosse uguale) è stato impiegato per la valutazione la dimensione minima fra le due dimensioni dei dataset considerati, ovvero 895 immagini. Una rapida presa visione dei dati ha mostrato che le immagini presenti nel dataset di farfalle reali contenevano dati di qualità molto variabile, con alcune immagini contenenti impurità o artefatti di vario genere. Non sono state operate tuttavia operazioni di pulizia dei dati, in quanto l'obiettivo era valutare l'efficacia delle metriche nel discriminare tra dati reali e generati, indipendentemente dalla qualità dei dati stessi.

In questo caso l'applicazione della KDE per le distanze inter e intra set è stata fondamentale per la comprensione dei risultati ottenuti, in quanto ha permesso di visualizzare le distribuzioni dei dati in uno spazio a dimensione ridotta, fornendo un quadro chiaro delle relazioni tra i campioni reali e quelli generati.

2.2.2 Scarlatti

Al contrario del dataset di farfalle, oer il dataset di partiture musicali è stato adottato un approccio più mirato e informato dal dominio dei dati. Per l'estrazione delle caratteristiche infatti sono state utilizzate informazioni di carattere ritmico e tonale, seguendo quanto descritto nella letteratura [4]. Il paper indicato, in particolare, presentava un implementazione delle metriche per l'estrazione di caratteristiche musicali basato su una rappresentazione delle partiture musicali in forma di **piano-roll**. Il piano-roll è una rappresentazione grafica delle note musicali in cui l'asse delle ascisse rappresenta il tempo e l'asse delle ordinate rappresenta le note. Ogni cella della matrice corrisponde a una nota suonata in un determinato istante di tempo. Presenta tuttavia delle limitazioni, in quanto in condizioni particolari può esserci perdita di informazione. Invece che utilizzare il piano-roll, sono state estratte le caratteristiche direttamente dai file MIDI, quindi da una lista di note con informazioni sul tempo e sulla durata, reimplementando le metriche del paper ma con questa diversa rappresentazione. Le caratteristiche estratte si dividono in due categorie: **caratteristiche ritmiche** e **caratteristiche tonali**. Le prime includono informazioni sul tempo e sulla durata delle note, mentre le seconde riguardano la tonalità delle note e le relazioni armoniche tra di esse. Tra le diverse metriche sono poi presenti: metriche scalari, metriche vettoriali e persino metriche matriciali.

Con 'measure' si intende una singola battuta di musica, ovvero un'unità di tempo musicale, ad esempio se il tempo è 4/4, una battuta è composta da 4 semiminime (o 8 crome, o 16 semicrome, ecc.). La maggior parte dei dati musicali di musica barocca è scritta in 4/4 con 8 battute, per quei dati in cui il tempo non era specificato è stato considerato 4/4, e in caso di battute mancanti sono state aggiunte battute vuote.

Caratteristiche tonali:

- **number of pitches per measure** - il numero di toni diversi presenti in una singola battuta (vettore di dimensione 8)
- **pitch class histogram** - l'istogramma delle classi di toni presenti in una singola battuta (vettore di dimensione 12)
- **pitch class histogram per measure** - l'istogramma delle classi di toni presenti in una singola battuta (vettore di dimensione 12x8 = 96)
- **pitch class transition matrix** - la matrice di transizione delle classi di toni (matrice 12x12=144)
- **pitch range** - la differenza tra il tono più alto e il tono più basso (scalare)
- **average pitch shift** - la media delle variazioni di tono tra note consecutive (scalare)

Caratteristiche ritmiche:

- **number of notes per measure** - il numero di note presenti in una singola battuta (vettore di dimensione 8)
- **note length histogram** - l'istogramma delle durata delle note presenti in una singola battuta (vettore di dimensione 24)
- **note length histogram per measure** - l'istogramma delle durata delle note presenti in una singola battuta (vettore di dimensione 24x8 = 192)
- **note length transition matrix** - la matrice di transizione delle durate delle note (matrice 24x24=576)

- **average IOI** - l'intervallo inter-onset medio, ovvero la media delle variazioni di durate di note consecutive (scalare)

Mentre per le immagini di farfalle avevamo un unico modello generativo per la generazione dei dati, per le partiture musicali sono stati utilizzati più modelli generativi, o meglio un unico modello a cinque diverse epoche di training. Per la valutazione delle metriche sono stati utilizzati i dati generati da ciascuna epoca di training, confrontandoli con i dati reali. Ci si aspetta che le metriche siano sensibili alle differenze tra i dati generati a diverse epoche di training, riflettendo le variazioni qualitative dei campioni generati.

Il primo test eseguito è stato un sanity check per verificare che la kde calcolata sulle inter e intra set distance fosse corretta. Per fare ciò sono state prese come distribuzioni di dati il training e test set per le varie epoche di training del modello generativo, l'obiettivo era verificare che la distribuzione dei dati, essendo essi in questo caso reali per entrambi i set, fosse pressoché identica. Sono state poi operati gli stessi esperimenti fatti con il set di immagini di farfalle, ovvero la valutazione delle metriche su dati reali e generati. In questa istanza sono state calcolate e confrontate le kde per le distanze interset per i diversi modelli, così come l'**overlaped area** tra le distribuzioni. Infine sono stati individuati i falsi positivi con un approccio **leave-one-out**, ovvero per ogni campione generato si è calcolato se in caso di sua rimozione la kde calcolata sulle distanze interset lo avrebbe considerato come falso positivo.

Dato che i primi dati e ultimi dati del dataset generato erano di bassa qualità, del sample di 1000 dati sono stati rimossi i primi 150 e gli ultimi 150 dati, le metriche hanno quindi operato su un dataset di 700 dati, sia per i dati reali che per quelli generati.

Capitolo 3

Conclusioni

3.1 Risultati degli esperimenti sui toy-dataset

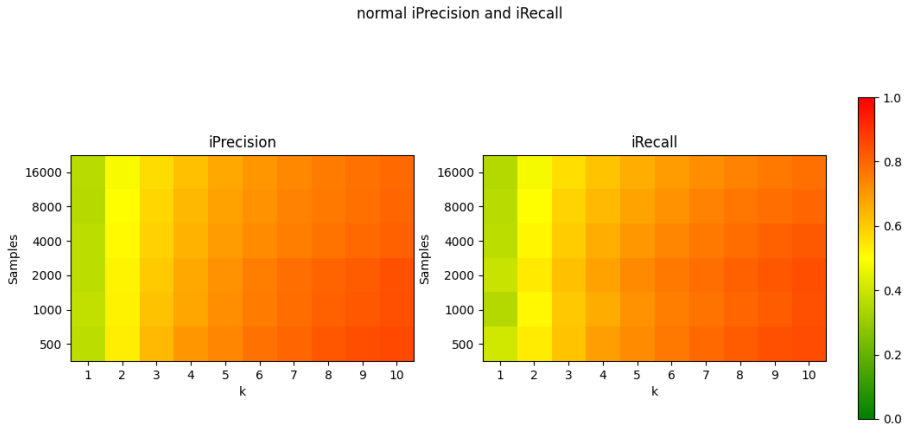
3.1.1 Parametro k e dimensione del dataset

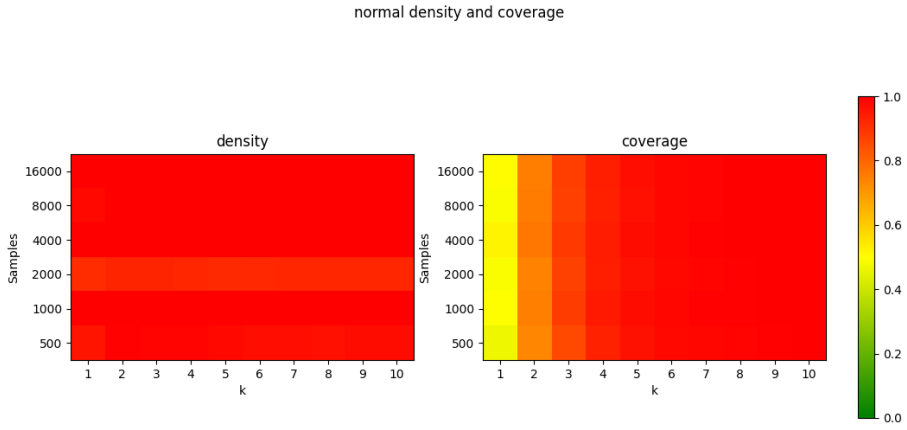
3.1.2 Dimensione del dataset e dimensione dei dati

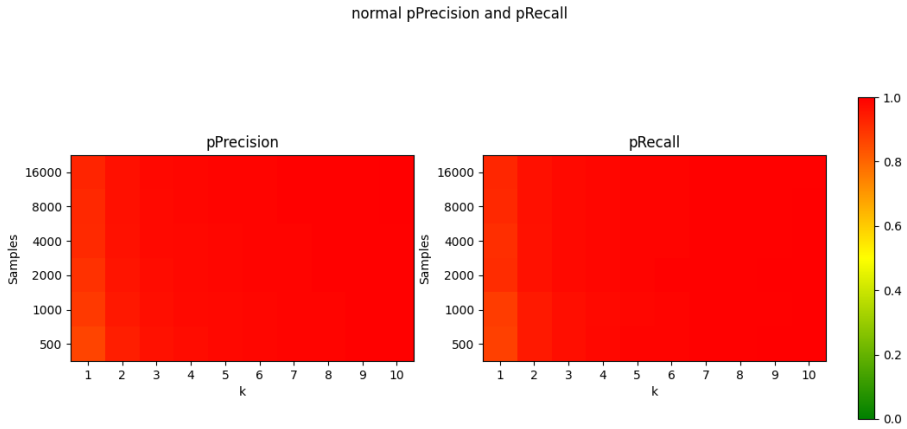
3.1.3 Outliers

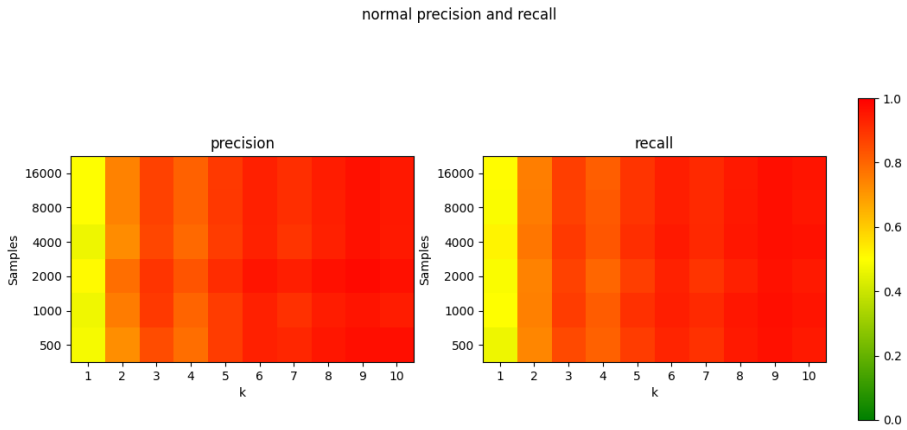
3.1.4 Comparazione con implementazioni esistenti e riproduzione delle pr-curves

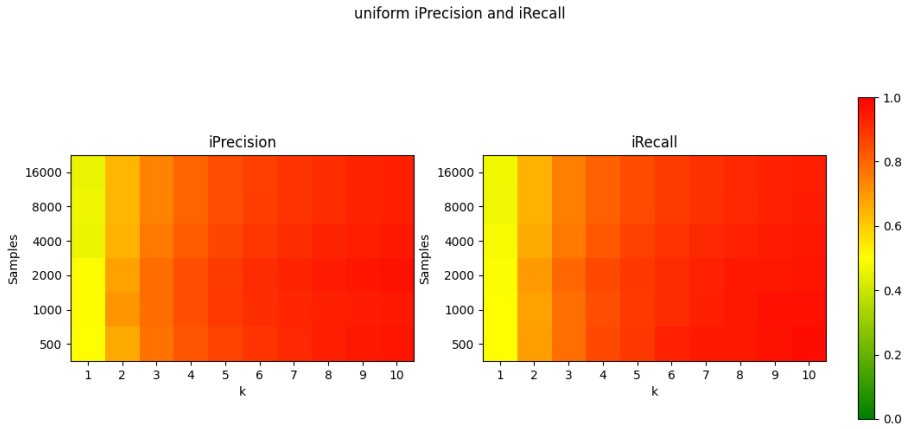
3.2 Risultati degli esperimenti su dataset reali

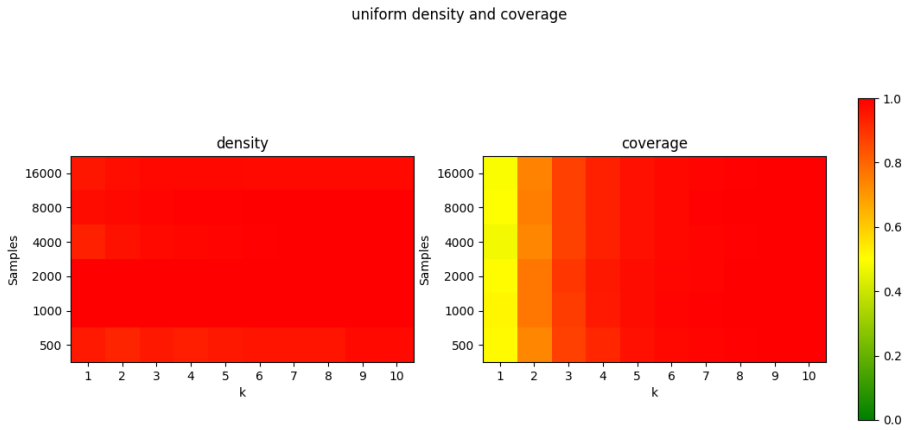

normal iPrecision and iRecall
iPrecision
iRecall
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

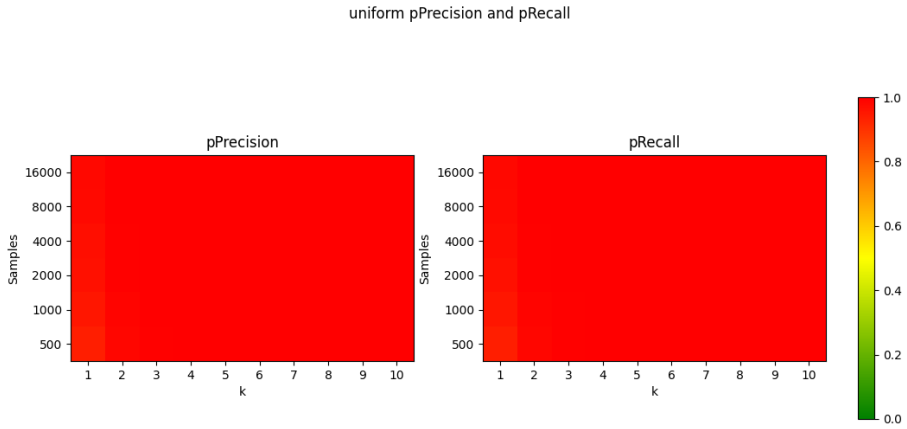

normal density and coverage
density
coverage
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

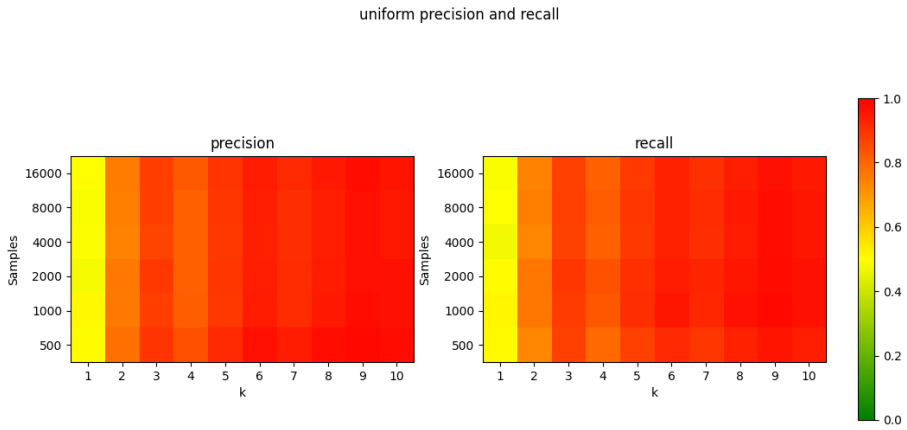

normal pPrecision and pRecall
pPrecision
pRecall
Samples
k
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
1.0
0.8
0.6
0.4
0.2
0.0

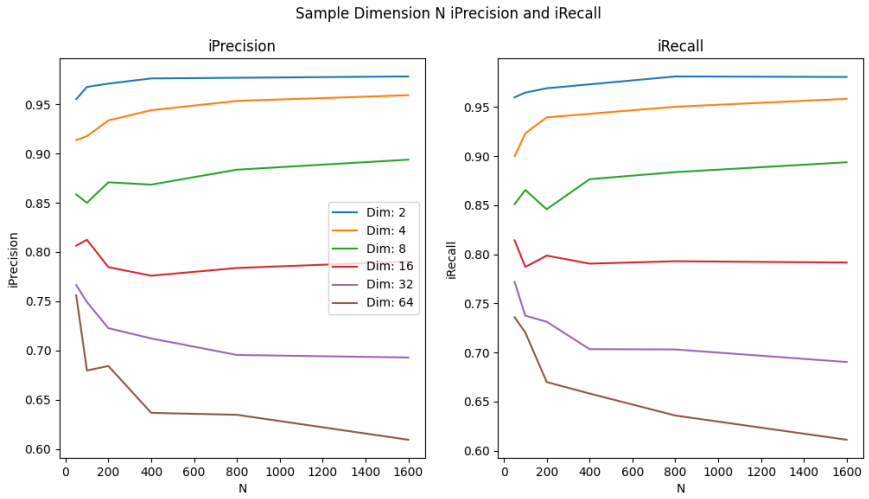

normal precision and recall
precision
recall
Samples
k
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
1.0
0.8
0.6
0.4
0.2
0.0

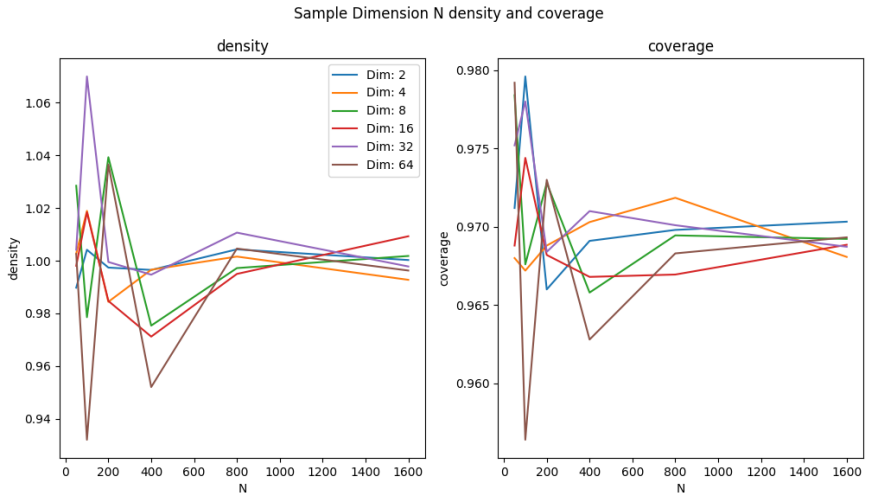

uniform iPrecision and iRecall
iPrecision
iRecall
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

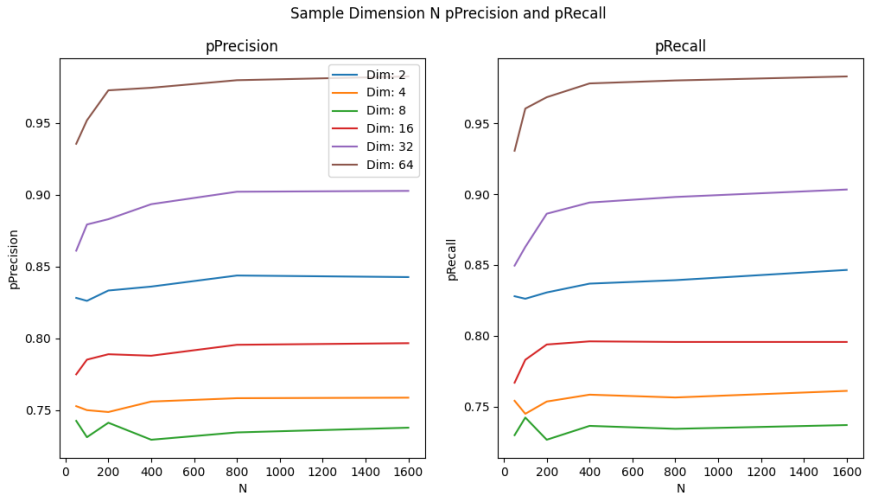

uniform density and coverage
density
coverage
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

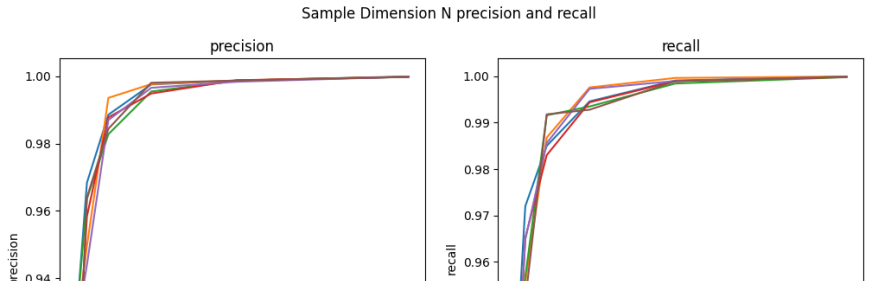

uniform pPrecision and pRecall
pPrecision
pRecall
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

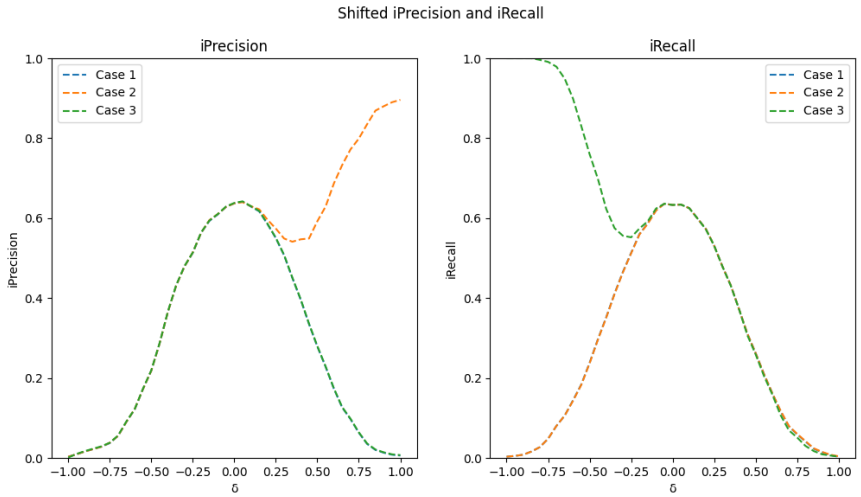

uniform precision and recall
precision
recall
Samples
16000
8000
4000
2000
1000
500
1 2 3 4 5 6 7 8 9 10
k
1.0
0.8
0.6
0.4
0.2
0.0

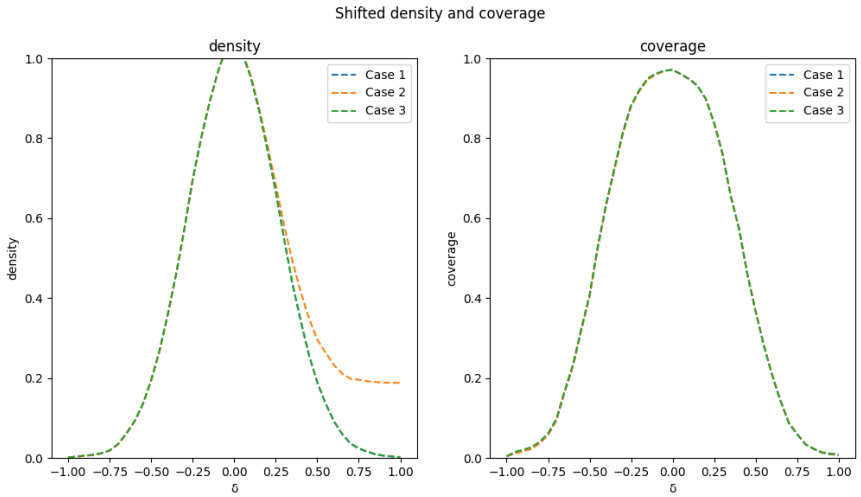

Sample Dimension N iPrecision and iRecall
iPrecision
iRecall
iPrecision
iRecall
N
N
Dim: 2
Dim: 4
Dim: 8
Dim: 16
Dim: 32
Dim: 64

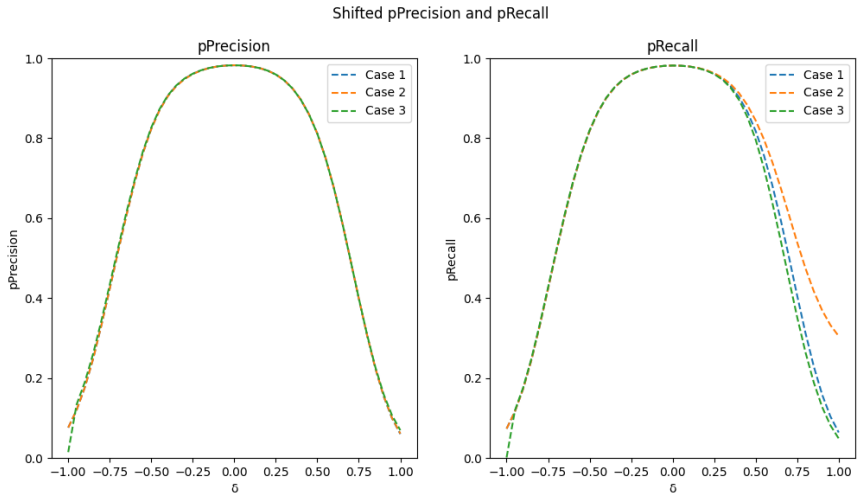

Sample Dimension N density and coverage
density
coverage
density
coverage
N
N
Dim: 2
Dim: 4
Dim: 8
Dim: 16
Dim: 32
Dim: 64

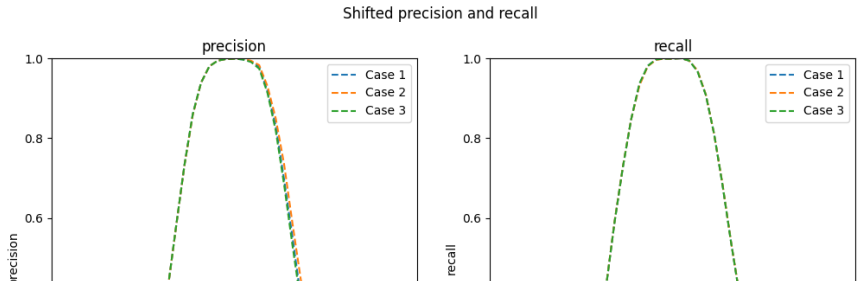

Sample Dimension N pPrecision and pRecall
pPrecision
pRecall
pPrecision
pRecall
N
N
Dim: 2
Dim: 4
Dim: 8
Dim: 16
Dim: 32
Dim: 64

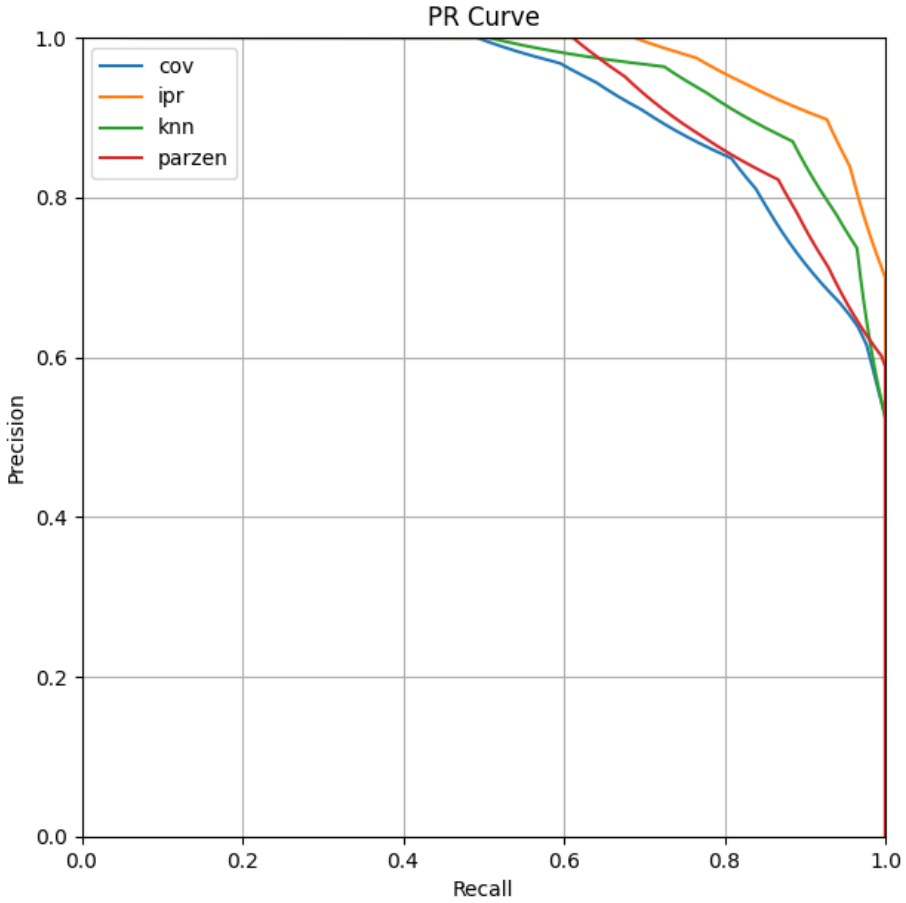

Sample Dimension N precision and recall
precision
recall

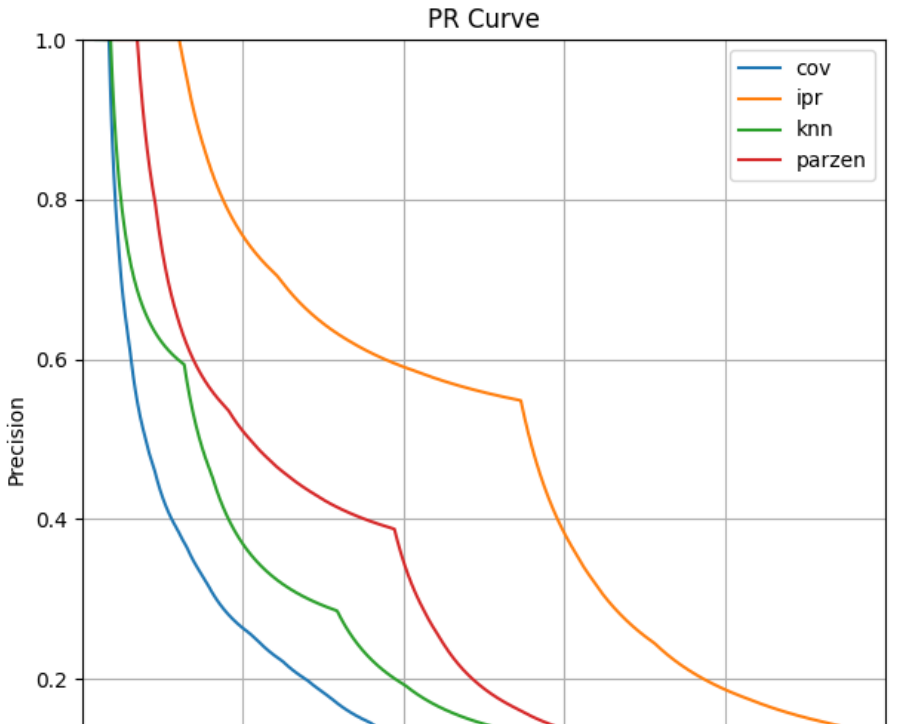

Shifted iPrecision and iRecall
iPrecision
iRecall
Case 1
Case 2
Case 3
iPrecision
iRecall
δ


Shifted density and coverage
density
coverage
Case 1
Case 2
Case 3
density
coverage
δ


Shifted pPrecision and pRecall
pPrecision
pRecall
Case 1
Case 2
Case 3
pPrecision
pRecall
δ


Shifted precision and recall
precision
recall
Case 1
Case 2
Case 3
δ


PR Curve
cov
ipr
knn
parzen
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


PR Curve
cov
ipr
knn
parzen
Precision
1.0
0.8
0.6
0.4
0.2

Capitolo 4

Note Implementative

Per la generazione di dataset con distribuzione uniforme è stata utilizzata la funzione `numpy.random.uniform`, mentre per la generazione di dataset con distribuzione normale è stata utilizzata la funzione `numpy.random.multivariate_normal` di numpy. La prima oltre alla dimensione dei sample e al numero prende come parametri anche il range di valori che i dati possono assumere, mentre la seconda prende come parametri lo shift (che di default è 0). Come media avremo quindi `shift*numpy.ones(dim)` e come covarianza la matrice identità (`numpy.eye(dim)`). Sono state poi adottate una serie di funzioni per facilitare il debugging attraverso la visualizzazione dei dati. In particolare una funzione che mostri dati di due dimensioni con il corrispondente manifold e fuzioni come il **realism score** che permette di valutare la verosimiglianza dei singoli dati generati.

Questi grafici sono stati prodotti utilizzando la libreria `matplotlib` di python.

Bibliografia

[1] Julien Rabin Benjamin Sykes, Loic Simon. Unifying and extending precision recall metrics for assessing generative models. 2024.

[2] Suhyun Kim Dogyun Park. Probabilistic precision and recall towards reliable evaluation of generative models. 2023.

[3] Ruth Urner Fasil Cheema. Precision recall cover: A method for assessing generative models. 206(6571-6594), 2023.

[4] Alexander Lerch Li-Chia Yang1. On the evaluation of generative models in music. 32:4773–4784, 2020.

[5] Julien Rabin Loïc Simon, Ryan Webster. Revisiting precision and recall definition for generative model evaluation.

[6] Mario Lucic Olivier Bousquet Sylvain Gelly Mehdi S. M. Sajjadi, Olivier Bachem. Assessing generative models via precision and recall. 2018.

[7] Youngjung Uh Yunjey Choi Jaejun Yoo Muhammad Ferjad Naeem, Seong Joon Oh. Reliable fidelity and diversity metrics for generative models. 2019.

[8] Samuli Laine Jaakko Lehtinen Timo Aila Tuomas Kynkäänniemi, Tero Karras. Improved precision and recall metric for assessing generative models. 2019.