EasyChair Preprint

№ 941

Analysis of geospatial data for Urban Informatics applications: the case of Google Place in the city of Milan

Domenico Monaco

April 28, 2019

UNIVERSITA' DEGLI STUDI DI MILANO-BICOCCA

Scuola di Scienze

Dipartimento di Informatica, Sistemistica e Comunicazione

Corso di Laurea Magistrale in Teoria e Tecnologia della Comunicazione

Analisi di dati geospaziali per applicazioni di Urban Informatics: il caso dei Google Place nella città di Milano

Relatore:
Prof. Giuseppe Vizzari

Correlatore:
Dott. Andrea Gorrini

Tesi di Laurea di:
Domenico Monaco
Matricola n. 803245

Anno Accademico 2017 / 2018

Indice

Lista delle figure

Introduzione

La città digitale

Lo sviluppo e la diffusione delle tecnologie hanno portato alla digitalizzazione della vita di tutti i giorni. Ogni nostra azione, come utilizzare i mezzi pubblici, incontrare un conoscente o fare dello sport, è condizionata dai dati digitali e ne produce di nuovi. In altre parole la nostra quotidianità si fonde con la tecnologia, creando una rappresentazione digitale di noi e delle nostre interazioni. [35]

Si parla ormai di una Città Digitale, rappresentazione digitale della città fisica ovvero dei suoi spazi, dei suoi utilizzatori e delle sue dinamiche sotto forma di flussi di informazioni digitali sempre più eterogenei fra loro ed in quantità sempre più elevate. Oggi più che mai la realtà sociale, economica e materiale della città dipendono in modo inevitabile da flussi di informazioni. Dare un senso a questa massa di dati eterogenei, che potrebbero essere utilizzati ben al di là delle singole funzioni, spesso di natura commerciale, per le quali sono stati prodotti, è una delle sfide del futuro.

Interpretare la città come realtà fisica, economica e sociale attraverso la sua rappresentazione digitale vuol dire introdurre nuovi metodi di analisi. Esplorare nuove forme di acquisizione di dati ed introdurre nuove tecnologie e metodologie di analisi a supporto del processo di produzione della conoscenza della città è l'obiettivo di questo lavoro.

L'analisi di dati geospaziali

L'elaborato descrive un caso di studio relativo all'analisi di dati geospaziali per applicazioni di Urban Informatics sul territorio di Milano, quest'ultima realizzata attraverso l'acquisizione di Google Place e la successiva caratterizzazione del territorio utilizzando tecniche di clustering spaziale ed altri metodi di analisi.

L'elaborato si sviluppa in cinque capitoli divisi in tre parti: una prima parte di stato dell'arte, una seconda relativa alla descrizione del progetto

ed infine ultima parte di discussione dei risultati, conclusioni e sviluppi futuri.

Il primo capitolo affronta lo stato dell'arte della Urban Informatics, dei nuovi metodi di analisi delle dinamiche urbane e le nuove informazioni spaziali. Mentre il secondo capitolo affronta in modo dettagliato lo stato dell'arte dei metodi di clustering e più nello specifico del clustering spaziale e gerarchico sui quali è basato l'approccio sviluppato in questo lavoro.

Mentre con i capitoli tre e quattro descrivono nel dettaglio le due fasi principali che compongono l'approccio di analisi geospaziale. Il capitolo tre, dopo un'introduzione alle caratteristiche dei dati offerti da Google Place, prosegue descrivendo in dettaglio l'acquisizione dei dati attraverso un metodo adattivo di raccolta dei dati. Mentre il quarto capitolo descrive inizialmente l'algoritmo di clustering gerarchico-iterativo basato sul DBSCAN e successivamente affronta un'analisi preliminare dei dati.

Infine, l'elaborato si conclude con i capitoli cinque e sei che realizzano rispettivamente l'analisi delle classificazioni ottenute dal processo di clustering e la valutazione del significato di tali risultati. Nello specifico il quinto capitolo effettua un'analisi di natura esplorativa sui dati ottenuti, attraverso una combinazione di metodi di analisi e visualizzazione di dati vettoriali e geospaziali. Mentre il sesto capitolo è composto dalle conclusioni e i possibili sviluppi futuri emersi durante la realizzazione di questo lavoro.

Parte I

Stato dell'arte

Capitolo 1

Urban Informatics

1.1 Urban informatics

Quello che viene definito come Urban Informatics è un campo di ricerca multidisciplinare relativamente recente, che può essere riassunto come il punto di incontro tra le scienze informatiche e le scienze urbane.

Queste ultime, essendo due aree di ricerca ampie e diversificate composte a loro volta da altre discipline, fanno sì che possono coesistere più definizioni dell'Urban Informatics descritte più in dettaglio nelle prossime sezioni.

1.1.1 Nuove tecnologie e Urban Experience

Tra le più citate definizioni di Urban Informatics troviamo quella fornita da Foth et al. in cui viene descritta come una sorta di ponte, non solo tecnologico, ma anche metodologico tra esperienza urbana e nuove tecnologie e, più in generale, tra il livello fisico e quello digitale della città.

Più specificatamente, Foth et al. definisce la Urban Informatics come un'attività composita, caratterizzata dallo studio, il design e l'applicazione dell'esperienza urbana attraverso diversi contesti tecnologici resi possibili grazie alle tecnologie in tempo-reale, ubique ed aumentate che mediano il mondo fisico e digitale.

L'Urban Computing tra pervasività ed ubiquità

Un altro modo di esprimere il punto di contatto tra scienze urbane e scienze informatiche è parlare di Urban Computing, Pervasive Computing e Ubiquitous Computing nel contesto urbano.

Urban Computing Kindberg et al. in [43] descrive l'Urban Computing come il punto di incontro tra la città e l'informatica attraverso l'integrazione di tecnologie di elaborazione, sensori e attuatori negli ambienti urbani e nella vita quotidiana (ad esempio, strade, piazze, pub, negozi, autobus e caffè e qualsiasi altro spazio).

Ubiquitous Computing O'Neill et al. nel 2006 propone, all'interno di un manifesto più articolato, anche l'informatica come parte integrante della progettazione degli ambienti urbani allo scopo di realizzare un sistema città dotato di metodi per l'osservazione, la registrazione, la modellazione e l'analisi della città dal punto di vista fisico, digitale e sociale. [56]

Pervasive Computing Pervasive Computing è un concetto molto simile al precedente che nel lavoro di Kostakos et al. è contestualizzato all'ambiente urbano attraverso il concetto di Urban Pervasive Infrastructure (UPI), che la definisce come un'infrastruttura composta dalle componenti umane, tecnologiche e spaziali che si integrano pervasivamente nel tessuto urbano. [46]

Comprensione delle dinamiche urbane

Il lavoro [36] offre una definizione di Urban Informatics basata su un'indagine condotta su accademici, amministratori pubblici ed amministratori dell'industria tecnologica allo scopo di indagare le applicazioni, i risvolti e gli impatti dell'informatica nella gestione della città.

Nello specifico definisce l'Urban Informatics come l'uso delle tecnologie dell'informazione e della comunicazione per comprendere meglio i bisogni, le sfide e le opportunità urbane in modo da fornire agli amministratori della città gli strumenti per:

- migliorare la fornitura di servizi;
- migliorare i processi di pianificazione;
- aumentare l'impegno pubblico dei cittadini per migliorare la risoluzione dei problemi.

Questo, rispetto ai dati, attraverso due approcci:

- un nuovo uso di dati già esistenti;
- l'uso di nuovi dati in applicazioni esistenti;

1.1.2 Big Data e Smart City

Una definizione più elaborata di Urban Informatics è quella che si sviluppa nell'articolo di Batty [3] che fornisce una visione più ampia e comprensiva della disciplina.

Batty mette in luce l'esistenza di diverse definizioni di Urban Informatics, in alcuni casi legate all'utilizzo dei dispositivi mobili, alle Smart City ed in altre legate ai Big Data.

Secondo questo punto di vista, la Urban Informatics viene definita come composta da due approcci: il primo, più ristretto ed orientato al controllo della città, si focalizza sul modo in cui hardware e software sono integrati nella città allo scopo di renderla più efficiente attraverso l'automazione e l'uso dei dati; il secondo approccio, più ampio e orientato alla gestione dinamica della città, si focalizza sull'uso delle tecnologie dell'informazione e della comunicazione al fine di abilitare servizi che possono essere distribuiti su più domini applicativi realizzando una maggiore partecipazione ed integrazione dei cittadini nella gestione della città. Quest'ultima che si sviluppa attraverso due concetti chiave: Smart City e Big Data.

L'autore conclude dimostrando la nascita di un nuovo campo di ricerca che collega:

- Smart City
- Big Data
- Scienza della città

Metodologie informatiche per l'analisi urbana

Un ulteriore punto di incontro tra scienze urbane e scienze informatiche può essere rappresentato dalle prossime definizioni che descrivono l'Urban Informatics come metodologia caratterizzata da tecnologie ed approcci analitici per la comprensione della città.

Tra i processi utili all'analisi dei fenomeni urbani troviamo:

- processo di acquisizione, integrazione ed analisi
- esplorazione e comprensione urbana

Processo di acquisizione, integrazione ed analisi Una definizione che mette in luce la natura metodologica dell'Urban Informatics rispetto ai Big Data è fornita nel lavoro [77].

In esso, la Urban Informatics, viene descritta come un processo di acquisizione, integrazione ed analisi di grandi quantità di dati eterogenei,

generati da diverse fonti come sensori distribuiti, veicoli, edifici e persone; con lo scopo di affrontare le prossime grandi sfide create dalle moderne città.

Esplorazione e comprensione urbana Una ulteriore definizione di Urban Informatics è fornita nel lavoro [67], essa mette in luce la necessità di metodologie, tecnologie e approcci in grado di estrarre e trasferire conoscenza tra gli attori della città del futuro.

Il lavoro definisce Urban Informatics come lo studio e la comprensione del contesto rubano allo scopo di gestire dinamicamente le risorse, creare e trasferire conoscenza, comprendere modelli e dinamiche urbane, migliorare la partecipazione civica e la pianificazione urbana.

Sempre secondo l'autore la ricerca in merito alla Urban Informatics è guidata sia approcci basati sulla teoria che su scelte empiriche dovute all'incertezza dei Big data.

1.1.3 Verso i Super-organismi Urbani Intelligenti

L'ambiente urbano sta diventando un sistema complesso che Zambonelli nel lavoro [76] chiama super-organismo urbano che, rifacendosi al concetto di super-organismo dell'ambiente animale, è visto come una grande intelligenza composta da più entità che attuano comportamenti come fossero un unico grande organismo. [17][76][5]

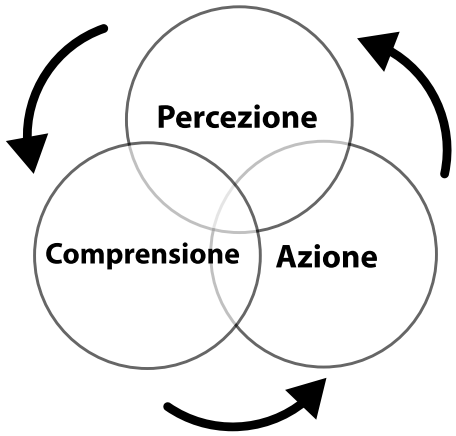


Figura 1.1: Le capacità del super-organismo urbano secondo il lavoro proposto in [76]

Secondo Zambonelli gli ambienti urbani dell'era moderna sono composti da una collettività di entità socio-tecnologiche che cooperano in attività di percezione, comprensione e azione creando una super-organismo nel contesto urbano che consente loro di pianificare, indirizzare le risposte e risolvere i problemi.

Possiamo anche vedere il super-organismo urbano come una rete di agenti intelligenti eterogenei i quali possono interagire con il mondo circostante in modo ibrido, ovvero mediante intelligenze ed abilità sia artificiali che umane.

L'emergere di un super-organismo urbano comporta la condivisione di capacità individuali di rilevamento, comprensione e azione al servizio del gruppo in modo da amplificare le possibili azioni di pianificazione rispetto alle dinamiche urbane.

Tale visione di super-organismo urbano è certamente ottimistica rispetto allo stato attuale delle cose, ma per certi versi lo scenario socio-tecnologico di interazione tra entità intelligenti nel contesto urbano esiste già, ma è chiaramente lontano dall'essere realmente un super-organismo.

Entità urbane ed abilità Sulla base dei concetti proposti da Zambonelli è possibile realizzare una ulteriore definizione di super-organismo urbano, che nella sua definizione iniziale non specifica le entità che ne fanno parte (vedi figura 1.1).

Per definire quali sono le entità socio-tecnologiche che compongono il super-organismo urbano si scopone il concetto in abilità da una parte ed entità dall'altra (figura 1.2), dove:

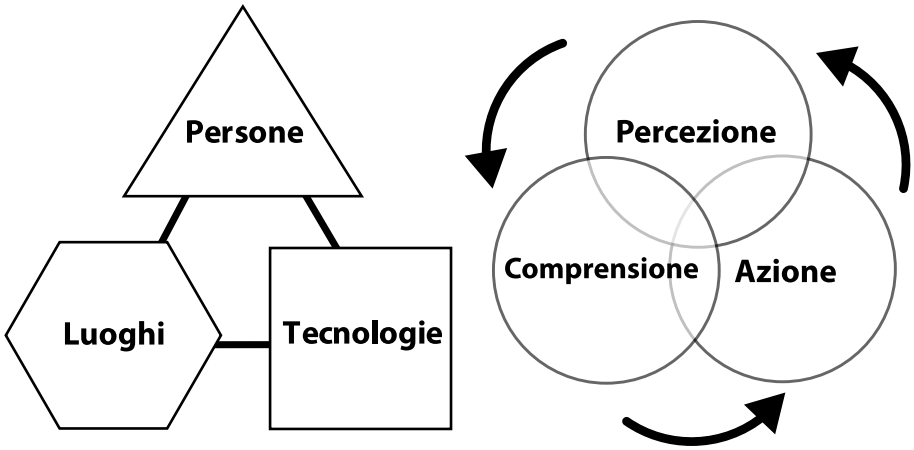
- abilità: si riferisce alle capacità possedute da una generica entità del super-organismo urbano; la quale può essere percezione, azione e comprensione, come nell'idea proposta da Zambonelli;
- entità: si riferisce alle entità socio-tecnologiche che fanno parte del super-organismo urbano, che in un contesto urbano possono essere persone, luoghi e tecnologie;

É così possibile attribuire un certo grado di intelligenza non solo alle persone, ma anche all'ambiente urbano ottenendo un vero super-organismo intelligente composto da entità che cooperano al fine di avere percezione, comprendere ed attuare azioni rispetto al contesto delle entità circostanti, compreso l'ambiente.

Tracce digitali della città

Proseguendo con l'idea di super-organismo urbano siamo obbligati a ripensare al significato dell'analisi di un *fatto* della città, che in questo caso si traduce nell'analisi di flussi di informazioni che si scambiano le entità socio-tecnologiche del contesto urbano.

Non è facile dare una classificazione dei diversi flussi di informazioni del contesto urbano, spesso analizzare un fenomeno nel contesto urbano



(a) Entità (b) Abilità

Figura 1.2: Scomposizione del Super-organismo Urbano in Entità ed Abilità

si traduce in una molteplicità metodologie di raccolta ed analisi delle informazioni che rendono difficoltosa una classificazione precisa.

Ma, partendo dai lavori [77] e [67] alcune possibili tracce digitali della città si possono trovare ad esempio nei sistemi di sensori distribuiti, nei contenuti generati dagli utenti, nei dati amministrativi o in statistiche governative, nei dati del settore privato come le transazioni economiche.

Tali sorgenti possono essere diverse e spesso eterogenee tra loro e provenienti da aree di applicazione diverse, come i dati sul traffico, del settore economico, industriale, energetico, della salute e dell'ambiente.

In questo senso i dati urbani hanno molto in comune con i Big Data che solitamente si riferiscono a grandi quantità di dati eterogenei tra loro di tipo sia strutturato che non strutturato, difficilmente trattabili con tecniche tradizionali di trattamento dati.

1.1.4 Le valutazioni delle dinamiche urbane

Una buona progettazione non è sufficiente a conferire qualità ad un'area urbana, esistono diversi fattori ad influenzare la qualità dei luoghi rispetto ai loro utilizzatori, a tal proposito Jacobs introduce i concetti di vitalità e diversità come misura della qualità di un luogo. [33]

Un'altra misura della qualità di un luogo consiste nello studio della pedonalità, che mira a valutare quegli elementi di progettazione infrastrutturale che garantiscono il comfort, la sicurezza e l'accessibilità delle strutture urbane ad uso pedonale, considerando anche le esigenze specifiche delle persone con mobilità ridotta. La vitalità di un luogo è uno dei fattori che definisce la pedonalità,. [57] [24]

Infatti uno dei tratti fondamentali della vitalità è la presenza umana nelle attività del luogo, dunque pedonalità e presenza di flussi sono aspetti rilevanti nella valutazione, in termini di variazioni temporali, intensità e durata ed eterogeneità dei flussi. [64]

La pedonalità è particolarmente sentita nelle aree fortemente orientate al turismo ed in particolare a quelle aree ad esclusivo accesso pedonale come accade a Venezia ed in alcune zone delle più grandi città europee. [24]

Nello specifico la pedonalità viene valutata attraverso quattro criteri fondamentali:

- utilità;
- comfort;
- sicurezza;
- attrattiva.

Gorrini, nel suo lavoro [24], aggiunge la *leggibilità* delle indicazioni come parametro di valutazione della pedonalità in contesti fortemente turistici.

Tali criteri possono essere misurati attraverso indicatori strutturali relativi agli elementi spaziali dell'area, indicatori comportamentali delle persone relativamente alle caratteristiche spaziali ed indicatori soggettivi di valutazione delle persone nella città.

Le informazioni relative agli indicatori sono solitamente ottenute con metodi tradizionali di raccolta quali: questionari, interviste ed analisi in loco; mentre alcuni lavori più recenti si sono concentrati nella sperimentazione di soluzioni meno costoste e più precise come l'analisi dei degli open data governativi o il data mining di dati provenienti dal Web, applicazioni mobili e social media. [57] [24]

Social media e Walkability Il lavoro di Quercia [57] esplora la possibilità di utilizzare i dati dei social media per identificare automaticamente le strade più sicure e percorribili da un pedone, tale caratteristica è identificata dal termine *camminabilità* (definita dall'autore *walkable street*), misurata con l'assegnazione di punteggi alle strade attraverso l'analisi dei dati ottenuti dai servizi web Flickr e Foursquare.

Una parte dell'analisi è rivolta all'identificazione di parole chiavi legate alla pedonalità analizzando come sia possibile collegare i comportamenti online con caratteristiche urbane e comportamenti offline.

Quercia [57] rileva che gli upload dei contenuti di Flickr effettuati in strade più percorribili differiscono da quelle meno percorribili soprattutto in termini di ora di caricamento e tipi di tag associati.

Infatti, a partire dall'idea che collega un aumento del numero di crimini in strade meno frequentate da pedoni che fungono da sistema naturale di sorveglianza, ha rilevato un aumento del numero di foto caricate su Flickr maggiore nelle ore notturne in strade generalmente più sicure, suggerendo che è possibile profilare la pedonabilità delle strade attraverso gli User Generated Content (UGC) dei social media senza utilizzare invadenti e costosi sondaggi.

Un altro lavoro utile come spunto per l'utilizzo dei dati dei social media per realizzare analisi delle dinamiche urbane è quello di Berzi [4] che propone un'analisi degli UGC per misurare la pedonalità della città di Milano, basandosi sull'utilizzo di oltre 500mila dati ottenuti da Flickr e Foursquare.

Il lavoro si conclude dimostrando come un'analisi dei dati dei social media può fornire informazioni relativamente all'attrattività di un'area della città per abitanti e visitatori al fine di misurare la pedonalità utilizzando un approccio bottom-up in alternativa ai più tradizionali metodo dall'alto verso il basso, offrendo numerosi spunti e domande di ricerca future che meritano ulteriori approfondimenti e ricerche.

Vitalità e nuove metriche urbane Il lavoro di Sulis et al. [66] propone un approccio computazionale al concetto di vitalità e diversità relativamente alle aree urbane facendo uso di nuovi dati digitali.

In particolare si concentra sulla possibilità di costruire nuove metriche di valutazione urbana basate su dati geolocalizzati come quelli provenienti dalle le smart card, dai dispositivi mobili e dai social media, al fine fornire misure quantitative per misurare fenomeni complessi e spesso intangibili come il concetto di vitalità introdotto da Jacobs.

Jacobs collega la vitalità direttamente alla presenza di persone in un luogo e la distribuzione dei flussi nel tempo (intensità, variazione e durata).

Partendo da tale collegamento tra presenza di persone e vitalità l'autore Sulis et al. realizza un'analisi della stessa sul territorio di Londra utilizzando i dati dati delle smart card contactless per l'accesso ai mezzi pubblici, di Twitter e di OpenStreetMap. I primi sono stati utilizzati per analizzare la diversità mentre i secondi, quelli provenienti da Twitter, sono stati utilizzati per calcolare la vitalità ed infine gli ultimi, quelli provenienti da OpenStreetMap, per convalidare e visualizzare le aree più vivaci della città di Londra.

Sulis et al. sostiene che al crescere della diversità si ha un aumento della vitalità, dunque quest'ultima è calcolata misurando il flusso di

persone nel tempo attraverso tre metriche:

- intensità: che rappresenta la quantità di persone presenti in un luogo in un determinato intervallo di tempo, calcolata attraverso il numero totale di persone che entrano ed escono dal luogo;
- variabilità: che rappresenta quanto varia nel tempo la quantità di persone, calcolata attraverso la differenza di flussi tra diversi giorni, utile per comprendere la stabilità dei modelli temporali in relazione a specifiche attività tra i giorni della settimana (i.e. legati al pendolarismo) oppure nel fine settimana (i.e. legati al tempo libero, allo shopping, ecc.);
- consistenza: che rappresenta la variazione oraria dei flussi durante lo stesso giorno, utile per osservare se i luoghi presentano uno schema temporale continuo o al contrario caratterizzato da picchi concentrati. Un numero maggiore di valori anomali indica un modello temporale più irregolare nell'uso dello spazio.

I risultati mostrano che la vitalità risulta più elevata nei luoghi centrali della città con interessanti eccezioni nella periferia, presumibilmente legati a flussi legati al tempo libero e al turismo.

In fine, il lavoro suggerisce che utilizzare nuovi dati digitali geolocalizzati per realizzare nuove metriche di valutazione della città permetterebbe di misurare anche quantitativamente aspetti complessi e difficili da comprendere, senza però dimenticare le limitazioni di questo tipo di dati in termini di rappresentatività.

Infatti, tali metriche così ottenute, non rappresentano una visione globale della città, ma piuttosto un modo per migliorare o validare valutazioni urbane effettuate con metodi tradizionali o con sorgenti di dati diversi.

Comprendere la città attraverso i servizi

La valutazione dei servizi in area urbana è una componente importante sia per la misura della vitalità che della pedonalità, attraverso valutazioni come la presenza, la natura e l'uso che di tali servizi se ne fa.

Dal punto di vista della vitalità un'analisi dei servizi può fornire informazioni al fine di validare e comprendere la correlazione tra i flussi e il tipo di servizi presenti in una determinata area.

Mentre da un punto di vista della pedonalità un'analisi dei basata sui servizi risulta utile per i seguenti criteri:

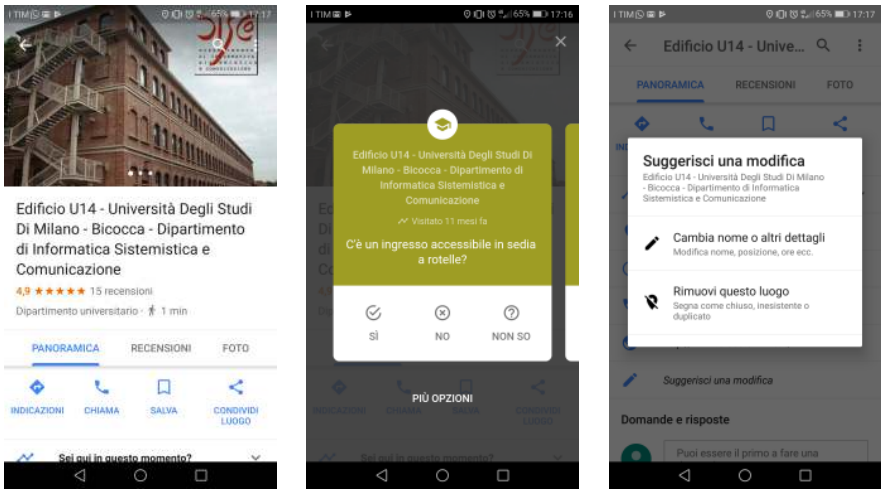
- utilità: la densità di servizi commerciali e presenza di servizi pubblici intorno a distanze pedonali;

- confort: la presenza di aree verdi, giardini o piazze pedonali;
- attrattività: la presenza di aree attrattive, aree e luoghi di interesse ed eventi;

In oltre, un'analisi dei servizi che renderebbero evidenti fenomeni che portano alla graduale scomparsa di servizi di base per i cittadini in favore di servizi per il turismo e la graduale riqualificazione di abitazioni in case vacanze, Bed and Breakfast e più in generale servizi di pernottamento. [24]

Uno degli strumenti basati su tecnologie web che può offrire la base per la valutazione dei servizi nel contesto urbano è certamente Google Map[1] che offre la possibilità a chiunque di inserire la propria attività commerciale o suggerirne di mancanti, rendendolo uno degli strumenti preferiti per chi cerca un determinato luogo, attività o punto di interesse e vuole raggiungerlo.

Google Map è un archivio di servizi nella città generalmente molto accurato per i grandi centri urbani, talvolta molto meno accurato per aree rurali o piccoli centri abitati. Google Map è in continuo aggiornamento, liberamente accessibile e realizzato in gran parte grazie ai contributi liberi e volontari degli utenti che posso inserire la propria attività commerciale, suggerire modifiche a quelle esistente e recensire i luoghi catalogati.



(a) (b) (c)

Figura 1.3: Crowdsourcing delle informazioni relative ad un place presente su Google Map

Il vantaggio di utilizzare Google Map come fonte per l'analisi dei servizi nelle aree urbane può essere quello di non dipendere da dinamiche di aggiornamento tipiche dei sistemi tradizionali

[1] http://maps.google.com

amministrativi ed al contempo avere una maggiore 'accuratezza dei dati tramite geolocalizzazione ed una maggiore ricchezza di informazioni collegate ai singoli place (ie. foto, web link, descrizioni, orari di apertura) senza dimenticare il fatto che spesso sono gli stessi gestori dei servizi a curare le informazioni perché interessati ad avere una elevata qualità delle descrizione e della georefernziazione.

Ciò rende Google Map il principale candidato all'analisi dei servizi delle aree urbane per questo lavoro.

1.2 Dati, informazioni e strumenti abilitanti

Fino a questo punto si è parlato di una *rappresentazione digitale della città* che può essere fotografata dai flussi di informazioni e relazioni tra Entità, che a loro volta attuano processi di percezione, comprensione e azione sulla base delle informazioni a loro disponibili.

Questa rappresentazione digitale è fornita soprattutto dalla diffusione delle tecnologie mobili ed internet e si riversa principalmente nelle piattaforme web quali social media o applicazioni mobili.

Quindi se esiste una rappresentazione digitale della città, in primo luogo la si può trovare nei dati e nelle informazioni dei social media. Non è certo una cosa nuova e diversi lavori hanno sperimentato la possibilità di analizzare la città attraverso i contenuti digitali generati dagli utenti, ma la natura mutevole della città e delle tecnologie rende necessaria la realizzazione di sperimentazioni.

1.2.1 Contenuti digitali urbani

Comunemente quando parliamo di contenuti degli utenti nel web e nei social media ci si riferisce ai cosiddetti UGC che vengono spesso descritti come generati dalla libera volontà degli utenti senza una organizzazione top-down.

Ma tale definizione dei contenuti generati dagli utenti non è realistica in quanto descrive i contenuti digitali online come "totalmente liberi" e specchio di quello che un dato utente vuol comunicare.

In realtà questo non è sempre vero perché a condizionare la produzione di UGC esistono diversi aspetti quali scelte di business dei social media, modalità di interazione o l'integrazione di contenuti degli utenti con altri generati da algoritmi o organizzazioni. Spesso quello che viene definito come UGC è una combinazione di dati generati da essere umani (utenti finali del servizio o personale specializzato) e sistemi automatici.

Prendiamo ad esempio una fotografia pubblicata sul popolare social network Instagram è certamente un contenuto generato da un essere

umano, ma le coordinate Global Positioning System (GPS) allegate ad essa non è detto siano aggiunte dall'utente, potrebbero essere state inserite automaticamente dall'App. Quindi nel suo complesso è UGC o no? In più cattura un qualche dettaglio dell'ambiente, ad esempio un panorama o il cielo azzurro e queste ultime informazioni seppur mediate da un essere umano ed uno strumento tecnologico sono comunque di origine ambientale.

Contestualizzazione spaziale

Parlare di interpretazione dei fenomeni urbani vuol dire catturare dati ed informazioni che parlano della città. Per decidere se un insieme di dati o informazioni sono utili a produrre conoscenza nel contesto urbano è necessario definire alcune caratteristiche di base.

Tali caratteristiche sono la componente *tempo* e la componente *spazio*, che accompagnano una terza componente, quella *informativa*, che può assumere una molteplicità di forme. Lo spazio è senza dubbio la caratteristica più importante, senza la quale sarebbero solo informazioni non contestualizzabili nello spazio urbano.

La componente spazio nei contenuti digitali può assumere diverse forme che può essere visto come un collegamento logico che contestualizza spazialmente l'informazione e può essere di due tipi [30]:

- about: si ha quando il contenuto si riferisce ad un luogo nello spazio;
- in: si ha quando il contenuto è creato e in un luogo nello spazio;

ed ovviamente il caso misto in cui si riferisce ad un luogo ed è creato in un altro luogo.

Stessa cosa accade con il tempo: esistono contenuti digitali che appartengono ad un dato istante, oppure si riferiscono ad un periodo.

Differenza e ambiguità tra "in" e "about" Un esempio utile per comprendere i due tipi di collegamento può essere offerto da un semplice post su un social media che dice: "Domani prenderò per la prima volta un aereo da Malpensa", creato il giorno 8/3/2018 alle 19:31 in Piazza del Duomo, Milano.

Il contenuto digitale in figura 1.4 fa riferimento a Milano Malpensa, ma è creato in Piazza Duomo, Milano. Fa riferimento ad un evento che accadrà il giorno seguente e non è scritto nel momento in cui accade l'evento. Ci dice in oltre che l'utente era in Piazza Duomo ed ha deciso di scrivere quel post, ci dice anche che presumibilmente non era impegnato e che stava programmando un viaggio per il giorno seguente da Malpensa e così via.

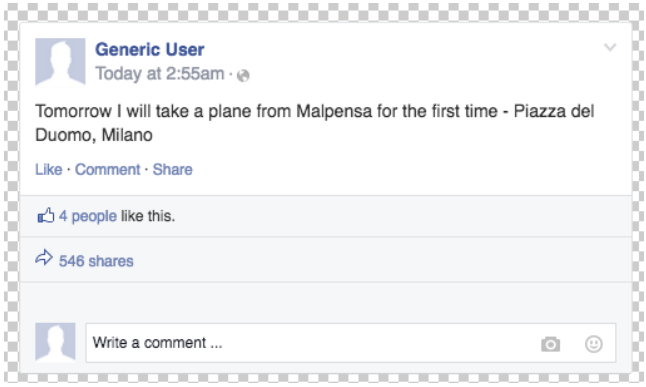


Figura 1.4: Esempio di post Facebook in cui si mostra la differenza della componente spazio "in" ed "about"

Il più delle volte la differenza del tipo di collegamento tra contenuto digitale e il luogo non è così netta come nell'esempio in figura 1.4, ma può essere più articolata e difficile da identificare rendendo la stessa analisi più complessa.

1.2.2 Informazioni geografiche

Le informazioni geografiche sono definite dalla letteratura come qualsiasi fatto, cosa, nome o altre informazioni collegate ad un determinato luogo geografico. Le informazioni geografiche possono essere sia cartacee che digitali e possono essere di diversa tipologia. [37] [38]

In modo più formale, secondo Goodchild, le informazioni geografiche sono una coppia di informazioni (x, z), dove x è una posizione nello spazio-tempo e z è un insieme di proprietà relativa alla determinata posizione. [20]

Le informazioni quando sono collegate ad un luogo geografico si dicono *georeferenziate*, tale tipo di collegamento può avvenire attraverso l'utilizzo di nomi (i.e. Milano, quartiere Brera), indirizzi (i.e. Via della Spiga Milano) o coordinate specifiche di un sistema geografico (i.e. GPS 5°28'22.1"N 9°10'38.3"E). [29]

Volunteered Geographic Information Una categoria particolare di informazioni geografiche sono i cosiddetti Volunteered geographic information (VGI), informazioni geografiche volontarie . Definite come informazioni geografiche che, a differenza delle altre, invece di essere prodotte da esperti di settore sono volontariamente create dagli utenti finali. [21]

La diffusione di tecnologie e piattaforme distribuite ha contribuito

all'aumentare in maniera significativa della quantità di VGI disponibili in diverse forme.

La diffusione di tali contenuti è cresciuta rapidamente negli ultimi anni grazie alla diffusione delle tecnologie internet, perché da un lato sono estremamente economiche per i fornitori mentre per l'utente finale tal volta risultaano più accurate. [48]

In questo contesto i VGI sono anche un sotto-insieme dei dati UGC con la sola differenza che riguardano specificatamente il dominio geografico. [21] [12]

Secondo [12] le VGI differiscono rispetto alle forme di informazione geografiche raccolte convenzionalmente per:

- contenuto delle informazioni;
- tecnologie di acquisizione;
- qualità e affidabilità delle informazioni;
- metodi e tecniche di utilizzo;
- processi sociali che sottostanno alla loro creazione;
- impatti sociali che il loro utilizzo possono avere;

Una piattaforma composta da VGI è OpenStreetMap che contiene una grande quantità di mappe ed informazioni geografiche create e gestite da utenti da tutto il mondo, in modo totalmente libero e gratuito.

Tra le VGI troviamo descrizioni, foto, estensione, nomi e parametri amministrativi nonchè le coordinate di riferimenti di un gran numero di luoghi, accuratamente redatti dagli utenti [21] [12] [26]

UGC geoeferenziati Infine [12] fa emergere un'altra importante categoria di informazioni geografiche volontarie, quelle per così dire *involontarie* generate attraverso comportamenti non direttamente mirati alla creazione di conoscenza geografica, ma che incidentalmente forniscono informazioni su un luogo o un'area.

A questa categoria di informazioni appartengono tutte le UGC georeferenziati, i quali appunto non sono creati per fini geografici, ma si possono riferire ad essi come tutti quei contenuti digitali creati e condivisi dagli utenti nei social media con tag geo-spaziali, coordinate geo-spaziali o riferimenti a luoghi.

Questo tipo di contenuti sono oggi molto popolari anche grazie alle tantissime Web App che promuovono l'inserimento e l'uso di informazioni geolocalizzate [12].

Le persone come Sensori distribuiti La diffusione delle tecnologie mobili e la tendenza a creare contenuti digitali georeferenziati conduce ad un interessante interpretazione degli utenti che possono essere visti come *sensori distribuiti* che interagendo con lo spazio urbano sono in grado di raccogliere, memorizzare e creare informazioni geografiche. [21]

Questioni sociali Una delle criticità della diffusione delle VGI consiste nel fatto possono presentare problemi legati all'accuratezza o al Digital Divide. L'accuratezza può essere compromessa in quanto le informazioni sono create da persone comuni con strumenti non specificatamente pensati a mappare lo spazio. Mentre il Digital Divide si avverte in quelle aree con una diffusione inferiore delle tecnologie mobili e di internet che produce l'effetto di una scarsità sia qualitativa de VGI. Elwood et al.

1.2.3 Sistemi per la gestione delle informazioni geografiche

Le informazioni geografiche nel contesto informatico necessitano di sistemi appositamente realizzati per la loro creazione, memorizzazione, distribuzione e modifica. Ne esistono di diverse tipologie ed offrono supporto a diverse applicazioni attraverso modalità di interazione differenti.

Sistemi GIS Un sistema informativo geografico detto Geographic Information System (GIS) è un'applicazione progettata per eseguire una vasta gamma di operazioni sulle informazioni geografiche. Può includere funzioni per l'inserimento, la memorizzazione, la visualizzazione, l'esportazione e l'analisi delle informazioni geografiche. Tali sistemi possono essere utilizzati in una vasta gamma di applicazioni come ad esempio la gestione delle risorse distribuite di un'azienda, la gestione delle emergenze, l'orientamento sul territorio delle persone ed in moltissime applicazioni scientifiche come l'ecologia o la criminologia. [22]

Web GIS partecipativi Storicamente i sistemi GIS sono applicazioni desktop, ma con la diffusione delle tecnologie web ed internet oggi ne esistono diversi accessibili attraverso la rete, detti diversi Web Geographic Information System (WebGIS).

Tra questi possiamo trovare ad esempio OpenStreetMap che oltre ad essere un importante sistema WebGIS è anche uno dei più grandi contenitori di VGI distribuite per mezzo di interfacce web, il progetto raccoglie e distribuisce liberamente mappe ottenute grazie a informazioni prodotte da utenti volontari: sentieri, strade, edifici,

località e tutto quello che si possa rappresentare sulle mappe digitali. Il suo obiettivo principale è di realizzare un sistema di mappe digitali aperte.

Un altro esempio di WebGIS sono i portali Sistemi Informativi Territoriali (SIT) che le regioni, provincie e Comuni italiani gestiscono per rendere disponibili informazioni relativamente al territorio attraverso mappe ed interfacce online.

Social Media per Informazioni Geografiche

Alle piattaforme appena citate sino ad ora si aggiunge un'altra categoria di WebGIS, questi sono i social media per Informazioni Geografiche quali ad esempio Google Map[2], Bing Map[3], Facebook Places[4], Foursquare [5] e TripAdvisor[6].

Ad esempio Google Map talvolta è definito come un GIS, ma può anche essere più genericamente definito come un social media con un forte orientamento ad informazioni geo-riferite, dunque un social media per informazioni geografiche.

La particolarità di tali sistemi sta nel fatto che non sono veri e propri GIS in quanto non sono prettamente pensati per gestire informazioni geografiche. Non si può nemmeno parlare di meri social media in quanto i loro contenuti cardine sono spesso luoghi e località.

Tali sistemi sono accessibili sia da Browser che App per Smartphone e forniscono informazioni dettagliate sul territorio attraverso mappe cartografiche digitali (vedi Web Map Service (WMS)) e i cosiddetti place.

Place Un place, in questo contesto, è qualsiasi cosa sia presente sulla superfice terrestre; il termine si traduce in luogo, ma può anche essere un monumento, un edificio, un parco o addirittura un'area della città; quindi può essere di natura puntuale o più estesa, ma comunque identificata da una singola coordinata e dei confini.

Tali place vengono corredati di un gran numero di informazioni come foto, descrizioni, orari di apertura e chiusura, numero di telefono e recensioni degli utenti.

Gli autori delle informazioni Tutte le informazioni rese disponibili sono una unione tra VGI, informazioni amministrative o generate da

[2]http://maps.google.com/
[3]https://www.bing.com/maps
[4]http://facebook.com/
[5]https://it.foursquare.com/
[6]https://www.tripadvisor.it/

algoritmi oppure create dalla compagnia che mette a disposizione il servizio.

I social media di questo tipo si differenziano sia dai più classici GIS sia dai WebGIS, offrendo spesso servizi di alta qualità in modo del tutto gratuito, come la geo-localizzazione dei servizi in una città o prevedere il percorso più breve per raggiungere una località. Ma operano sempre in precise modalità di distribuzione e manipolazione guidati dagli obiettivi di business della compagnia.

Questa caratteristica rende le mappe online uno strumento economico ed affidabile per la comprensione della città attraverso informazioni provenienti dagli utenti, ed è proprio su Google Map che intendiamo concentrare la nostra analisi urbana.

1.3 Caratterizzazione del territorio urbano

Così come accade, ad esempio, nei dati merceologici che è necessario raggrupparli in categorie per poterli analizzare, la stessa cosa accade per i dati geografici che necessitano di unità di riferimento per essere utilizzati al meglio, una necessità tipicamente amministrativa, ma utile in generale per organizzare le informazioni. Come vedremo esistono diversi modi di suddividere lo spazio geografico con diversi obiettivi e diversa granularità.

Nelle sezioni successive saranno affrontati principalmente due modalità di suddivisione geografica, la metodologia top-down e bottom-up.

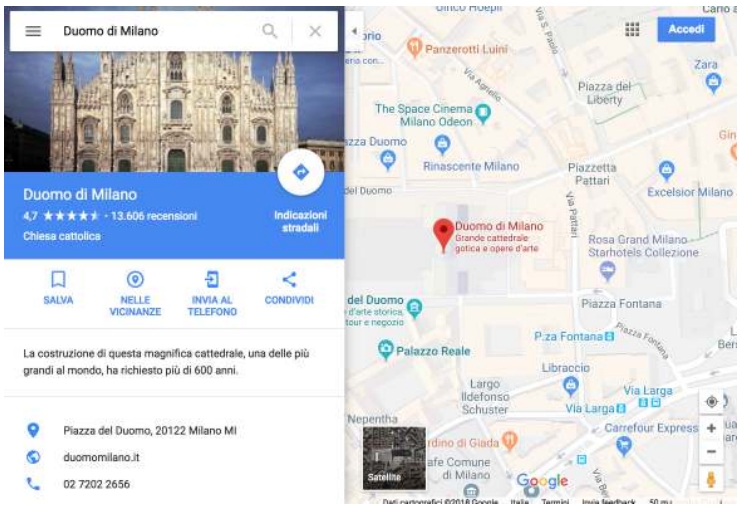


Figura 1.5: Place di Google Map relativo al Duomo di Milano

1.3.1 Unità di riferimento top-down

Nel caso delle unità di riferimento top-down abbiamo unità imposte *dall'alto* ovvero in modo fisso e preciso secondo parametri decisi a priori.

Area urbana Quando si trattano problemi urbani si parla generalmente di Area Urbana, e questo lavoro non fa eccezione che tratta principalmente l'Area Urbana di Milano, quindi la prima definizione che tentiamo di estrapolare è quale sia l'Area Urbana a cui dobbiamo fare riferimento, lo farò traducendo il testo della definizione resa disponibile dal National Geographic:

> *Un'Area Urbana è la regione che circonda una città. [...] Le aree urbane sono aree molto sviluppate, caratterizzate da una alta densità di strutture artificiali come edifici commerciali, strade, ponti e ferrovie. Il termine Area Urbana può riferirsi sia alla città [...] sia alle periferie. Un'area urbana comprende la città stessa, così come le aree circostanti. Molte aree urbane sono chiamate aree metropolitane,* extra-urbana *(greater in inglese), come Greater New York o Greater London, e quindi in italiano Area Metropolitana di Milano. Quando due o più aree metropolitane crescono fino a quando non si uniscono, il risultato è conosciuto come Megalopoli.* [7]

Sempre seguendo la definizione del Nationl Geographic, se parliamo di Aree Urbane si può parlare anche del suo opposto, le aree rurali, spesso definite *la campagna*, sono caratterizzate da una bassa densità di popolazione e grandi quantità di terra non sviluppata. Di solito, la differenza tra un'area rurale e un'area urbana appare chiara. Un'area intermedia tra Area rurale e Area Urbana è l'Area Industriale.

Quartiere Un altro termine comune per indicare le aree della città è quartiere e può essere definito come:

> *def. Quartiere: zona circoscritta di una città, con particolari caratteristiche storiche, topografiche o urbanistiche: quartiere residenziale; un vecchio quartiere popolare | quartieri bassi, la zona più popolare della città | quartieri alti, la zona più elegante | quartiere satellite, agglomerato urbano contiguo a una grande città, autonomo quanto a servizi ma non amministrativamente.* [8]

[7] https://www.nationalgeographic.org/encyclopedia/urban-area/
[8] https://www.garzantilinguistica.it/ricerca/?q=quartiere

Dunque un Quartiere non ha valenza amministrativa, ma piuttosto ha valore nel linguaggio comune, per indicare particolari aree storicamente agglomerate e non sempre facilmente circoscritte.

Distretto Diversa è la definizione di Distretto:

> *def. Distretto: suddivisione del territorio a fini amministrativi o giurisdizionali: distretto postale, scolastico, di corte d'appello* [9]

Diversamente dal quartiere, il distretto, ha valenza amministrativa e rappresenta una suddivisione del territorio con confini ed estensione precise.

Municipio Solitamente la definizione di Municipio è legato al Comune in quanto unità amministrativa che ingloba l'intero territorio di una città, sempre secondo il Garzanti:

> *def. Municipio: il comune, l'amministrazione comunale; la sede di questa amministrazione: essere impiegato al municipio; sposarsi in municipio* [10]

Ma il Comune di Milano (allo stesso modo come Roma, Genova, Torino e Napoli) utilizza tale termine per indicare un'ulteriore suddivisione dell'Area Urbana per fini amministrativi, talvolta chiamata Circoscrizione, che secondo la definizione del sito web del Comune viene definito secondo la descrizione seguente:

> *Ai Municipi vengono assegnate [...] risorse finanziarie per consentire un effettivo perseguimento dell'efficacia ed efficienza nell'erogazione dei servizi. L'entità delle risorse finanziarie spettanti a ciascun Municipio è determinata in base a criteri di riparto oggettivi che tengano conto dei parametri socio economici, delle caratteristiche territoriali, dei Municipi medesimi. Le competenze assegnate, dettagliate nel Regolamento dei Municipi approvato, sono diverse, tra le quali si segnalano quelle relative alla gestione dei servizi, quelle di proposta e consultive nei confronti degli Organi del Comune e inoltre promuovono l'informazione e la partecipazione dei cittadini in ordine all'attività del Municipio medesimo. Gli ambiti di intervento del Municipio, indicati nello Statuto comunale, sono i seguenti:servizi alla persona, educativi, culturali e sportivi, gestione e*

[9] https://www.garzantilinguistica.it/ricerca/?q=distretto%202
[10] https://www.garzantilinguistica.it/ricerca/?q=municipio

> *manutenzione del patrimonio comunale assegnato,edilizia privata, verde pubblico ed arredo urbano, sicurezza urbana e viabilità di quartiere, attività commerciali ed artigianali, rapporti con i cittadini in materia di entrate e lotta alla evasione. [...]* [11]

Per quanto riguarda il caso di Milano, i Municipi sono 9 e sono aree ben definite della città alle quali vengono assegnate particolari competenze di gestione del territorio attraverso delibere, fondi, responsabilità e competenze; attuate attraverso giunte locali e comunità.

NIL - Nuclei d'Identità Locale I NIL, nel contesto del comune di Milano, sono aree con un riconoscimento ed una funzione amministrativa paragonabili ai quartieri; nello specifico il Comune di Milano li definisce come:

> *I NIL - Nuclei d'Identità Locale rappresentano aree definibili come quartieri di Milano, in cui è possibile riconoscere quartieri storici e di progetto, con caratteristiche differenti gli uni dagli altri. Vengono introdotti dal PGT (Piano di Governo del Territorio) come un insieme di ambiti, connessi tra loro da infrastrutture e servizi per la mobilità, il verde. Sono sistemi di vitalità urbana: concentrazioni di attività commerciali locali, giardini, luoghi di aggregazione, servizi; ma sono anche 88 nuclei di identità locale da potenziare e progettare ed attraverso cui organizzare piccoli e grandi servizi (Piano dei Servizi).* [12]

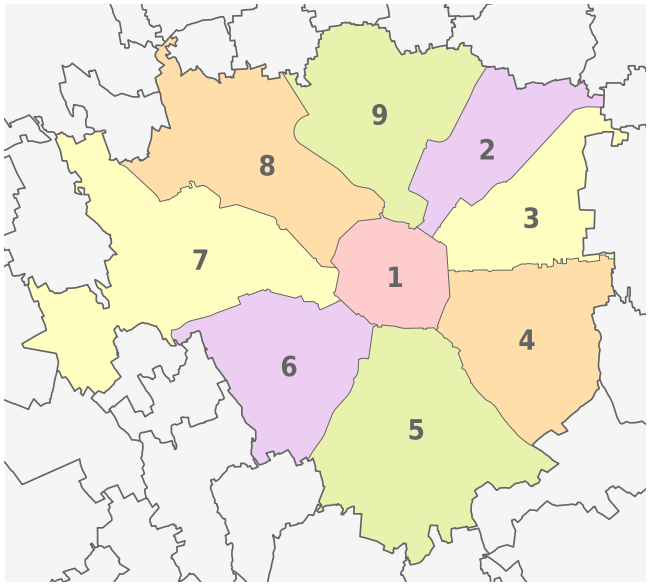
I NIL sono 88 e rappresentano un'ulteriore suddivisione territoriale del Comune di Milano e corrispondono alle unità minime di programmazione previste all'interno del PGT. [58]

1.3.2 Unità di riferimento bottom-up

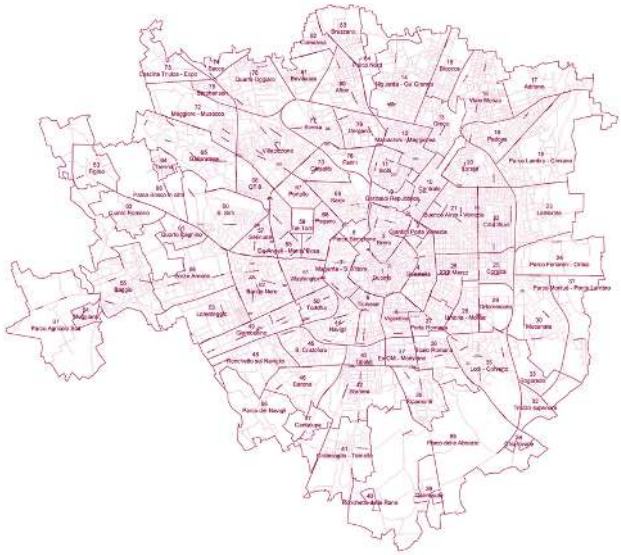
Un modo alternativo di raggruppare i dati è quello di far emergere classificazioni direttamente dalle caratteristiche dei dati stessi. Tale tipo di tecnica di suddivisione è detta bottom-up e nel contesto urbano permette realizzare unità di riferimento che emergono naturalmente dal contesto urbano. Tali raggruppamenti bottom-up possono cambiare in base al contesto, nel tempo e in base all'area urbana che si analizza; rappresentando una suddivisione alternativa del territorio più vicina al contesto di analisi.

[11] `http://www.comune.milano.it/wps/portal/ist/it/amministrazione/governo/Municipi/Municipi_in_dettaglio`

[12] `http://dati.comune.milano.it/dataset/ds61_infogeo_nil_localizzazione_`



(a)



(b)

Figura 1.6: 1.6a I Municipi di Milano [Wikipedia]; 1.6b Gli 88 NIL del comune di Milano definiti dal PGT [58];

Aree d'Interesse Il lavoro Hu et al. [32] realizza un'analisi delle aree urbane attraverso dati UGC geo-riferiti estraendo ed analizzando le così dette Area of interest (AOI).

Definisce genericamente le AOI come regioni all'interno di un ambiente urbano che attirano l'attenzione delle persone e propone un framework per l'estrazione attraverso foto geo-riferite presenti nei social media.

Sostenendo che l'identificazione di tali aree può rivelare informazioni utili agli urbanisti, analisti e tutti gli interessati della città al fine di pianificare nuove attività o estendere l'infrastruttura esistente.

Tali aree possono contenere ad esempio punti di riferimento della città, monumenti, centri commerciali o punti con vista panoramica sulla città. Il concetto di AOI è diverso dall'area urbanizzata in quanto oltre ad essere urbanizzato è soprattutto interessante per le persone.

Di conseguenza un quartiere che è senza dubbio un'area urbanizzata potrebbe non essere un AOI. Lo studio in oltre mette in luce come gli AOI sono soggettivi e spesso vaghi e raramente corrispondono ai confini amministrativi, collegando così le AOI ai *luoghi vaghi* (Vague Place). [54] [32] [39]

Dunque secondo Hu et al. gli AOI esistono nella percezione delle persone e sono definiti dai loro comportamenti, ma tale percezione è stata catturata con difficoltà fino alla diffusione dei social media.

Tali social media memorizzano le interazioni tra gli utenti e con il loro ambiente circostante, offrono quindi la possibilità di individuare aree urbane interessanti, il lavoro di Hu et al. propone un'insieme di metodi per l'estrazione e la comprensione degli AOI urbani basandosi sull'estrazione e l'analisi di UGC geolocalizzati.

L'analisi è stata effettuata raccogliendo le foto caricate su Flickr per sei diverse città di sei paesi diversi. Le foto raccolte vanno dal 2004 al 2014, dalle quali sono stati identificati gli AOI utilizzando l'algoritmo di Clustering DBSCAN, mentre la comprensione degli AOI avviene estraendo tag testuali associati alle immagini.

Quartieri vaghi Secondo [54] le persone tipicamente pensano e si esprimono in merito al mondo circostante attraverso termini e concetti vaghi. Spesso fanno riferimento a categorie che non hanno confini semantici netti e precisi, non fa eccezione il linguaggio naturale relativo allo spazio ed i luoghi, introducendo i Vague Spatial Concept che si dividono in due categorie: regioni e relazioni spaziali. Tra le relazioni possiamo trovare espressioni come: vicino, intorno, ad est. Mentre alcuni esempi di regioni vaghe relativamente ad aree urbane sono: la città bassa, riviera, valle ecc.

Seppur tendiamo ad identificare luoghi e spazi urbani entro determinati confini questi sono spesso relativi al contesto, soggettivi o dipendenti dalla cultura. In questo progetto di analisi e caratterizzazione degli ambienti urbani attraverso gli UGC è importante considerare la natura spesso vaga ed imprecisa dei luoghi e spazi urbani nella mente delle persone.

Un interessante lavoro in questo senso è quello proposto da Brindley et al. [6] che sfrutta una tendenza tipicamente legata al Regno Unito dove

i quartieri non sono aree amministrative ben definite, ma aree definite in modo informale che le persone usano per identificare un'area in modo colloquiale. Ottenendo così quartieri dai confini non ben definiti e che spesso si sovrappongono.

Tali nomi informali dei quartieri sono spesso inseriti negli indirizzi, appunto in modo informale. É proprio attraverso questa tendenza che il lavoro di Brindley et al. collega i quartieri informali ai dati UGC presenti online, in particolare sono stati estratti ed analizzati gli indirizzi presenti nei place di Bing.

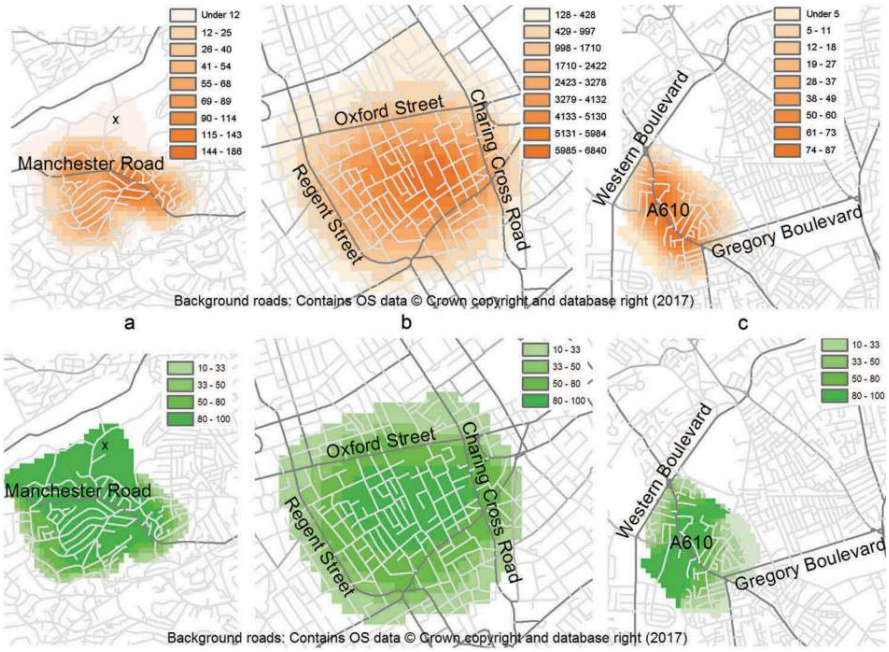


Figura 1.7: Esempio di quartieri vaghi ottenuti attraverso il metodo KDE [6]

L'autore propone una metodologia (bottom-up) per l'estrazione dei quartieri informali attraverso l'estrazione di dati UGC dal web utilizzando l'analisi KDE, senza che quartieri vengano definiti in precedenza, ma estratti e fatti emergere direttamente dai dati UGC.

Dimostrando la fattibilità di mappare i quartieri informali nel Regno Unito combinando modelli linguistici con i dati raccolti attraverso l'estrazione di indirizzi postali inseriti dagli utenti sul Web senza previa conoscenza dei nomi dei vicini stessi.

1.3.3 Proposta di lavoro

Sulla base di questo contesto è possibile delineare una proposta di lavoro relativa ad un'analisi di dati geolocalizzati nella città di Milano, la quale considera le persone come sensori distribuiti nella città che, per

mezzo delle nuove tecnologie mobili ed internet, forniscono un'immagine digitale della città attraverso flussi di informazioni. Questi ultimi, in parte, identificabili nei contenuti generati dagli utenti (UGC) geolocalizzati come quelli forniti dal servizio di mappe di Google [13].

Dunque, partendo dai contenuti geolocalizzati generati dagli utenti liberamente accessibili attraverso le Web API di Google, si vuole realizzare un processo bottom-up di suddivisione ed analisi del territorio di Milano, che permetta di far emergere le informazioni dai dati stessi piuttosto che imporli dall'alto.

Tale processo deve dunque essere composto dalle componenti di:

- acquisizione dei place attraverso Web API;
- suddivisione in aree della città;
- analisi e caratatterizzazione delle aree identificate.

Figura 1.8: Proposta generale di analisi (acquisizione, suddivisione e caratterizzazione) di dati geolocalizzati generati dagli utenti nella città di Milano attraverso un processo di tipo bottom-up

Quello che rende tale processo di analisi diverso dagli altri è il fatto di non imporre dall'alto né una suddivisione dello spazio urbano, né delle caratteristiche da ricercare nella città, ma lascia che i dati forniti dalle persone che interagiscono nel contesto urbano forniscano tali informazioni.

[13] Google Maps, `https://maps.google.com`

Il lavoro presenta numerose sfide già a partire dall'acquisizione dei contenuti generati dagli utenti che necessita di una strategia in grado di acquisire la totalità dei contenuti in modo autonomo, replicabile e con un consumo di risorse (tempo e memoria) minimo.

Mentre la parte del progetto che necessita di maggiore lavoro è certamente quello relativo alla suddivisione bottom-up del territorio di Milano a partire da contenuti geolocalizzati. Uno dei lavori precedenti a questo realizzato da Berzi [4] propone di utilizzare l'algoritmo DBSCAN per realizzare raggruppamenti di punti geolocalizzati.

Il lavoro [4], seppur mostrando con successo le qualità, evidenzia le criticità dell'algoritmo DBSCAN nell'identificazione di cluster tra aree della città con elevate differenze di densità dei punti, obbligando l'autore a realizzare diversi clustering con parametri differenti e scegliere manualmente, per ogni area, quella che secondo il sentire comune la rappresenta meglio. Manca dunque di un processo automatico di identificazione che permetta di ovviare al problema delle densità differenti.

É a partire da questa criticità che verrà affrontato il prossimo capitolo che mira ad esplorare lo stato dell'arte dei metodi di clustering allo scopo di fornire le basi per un punto di vista differente della città di Milano attraverso un processo di analisi di dati geospaziali di tipo bottom-up approfittando nel contempo della possibilità di acquisire dati differenti dal lavoro di Berzi [4].

Capitolo 2

Analisi dei dati

2.1 Data Clustering

Il Clustering può essere inteso come sinonimo di apprendimento non supervisionato. In genere è utilizzato per scoprire classi (detti anche cluster) di appartenenza all'interno di dataset. Tale tipo di apprendimeno si dice non supervisionato poiché le classi apprese non sono etichettate a priori, dunque il significato semantico delle classi non è deducibile direttamente dal processo di Clustering. [28]

Il Clustering è una componente essenziale per il Data Mining ed il Machine Learning settori di studio. Esistono diverse definizioni di Clustering, ma in generale è un processo automatico non supervisionato che permette il raggruppamento (o partizionamento) degli item di un dataset attraverso la similarità delle caratteristiche, misurata attraverso apposite formule dette *di similarità*. Con il temine Clustering talvolta ci si riferisce anche allo specifico raggrupamento ottenuto da un processo di classificazione non spervisionata. Da ora in poi ci si riferirà al processo con i termini *Processo di clustering*, *metodo di Clustering*, *Algoritmo di Clustering*, mentre all'insieme delle classi ottenute semplicemente con Clustering. [28] [19] [15] [11]

Esiste una differenza di terminologia anche quando si parla dello specifico proccesso di creazione dei cluster: in alcuni casi si dice che gli item sono raggruppati, in altri invece si dice che il dataset è partizionato. Il termine da utilizzare dipende dal tipo di tecnica usata per creare i gruppi, ma in entrambi i casi l'output di un Processo di clustering è un insieme di gruppi di item appartenent al dataset iniziale. [28]

Seppur sussistano diverse definizioni di Processo di clustering esiste invece una definizione generalmente condivisa di cosa sia un *buon Clustering* ovvero una buona riorganizzazione dei dati come risultato del

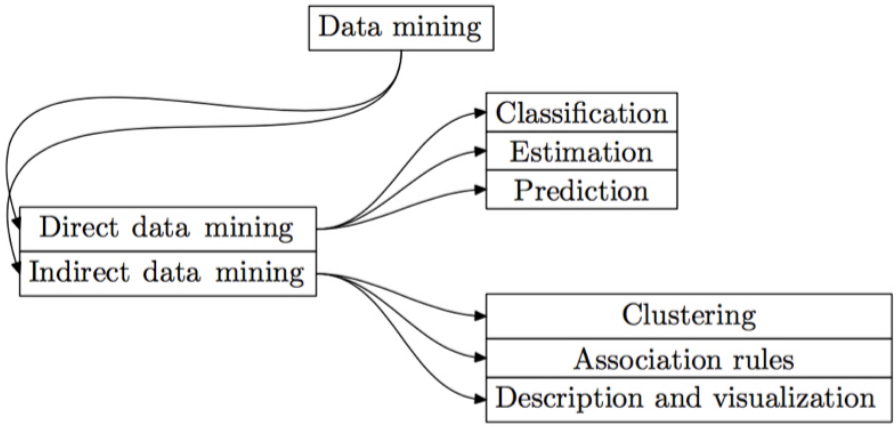


Figura 2.1: Classificazione dei metodi di Data Mining [19]

Processo di clustering ed è rappresentata dalla definizione seguente:

"The objects are clustered or grouped based on the principile of maximizing the inter-class similarity and minimizing intra-class similarity" [15]

Ovvero un *buon Clustering* si ha quando gli oggetti sono raggruppati in modo da massimizzare la similarità tra elementi di uno stesso gruppo e allo stesso tempo massimizzando la dissimilarità tra gruppi diversi.

Un modo alternativo per definire un buon Clustering è quello fornito da Mirkin che generalizza un -buon- cluster come *un set di entità coese per una data ragione, tale da esserlo maggiormente tra quelle di uno stesso cluster e meno con quelle di un cluster esterno.* [53]

"The concept of cluster typically refers to a set of entities that is cohesive in such a way that entities within are more similar to each other than to the outer entities " [53]

Una definizione ancor più generale e semplice è quella che vede il Processo di clustering come un'attività che, a partire da item non etichettati, raggruppa gli item simili in un stesso cluster, mentre raggruppa quelli dissimili in cluster diversi.

" In data clustering, we are given unlabeled data and are to put similar samples in one pile, called a cluster, and the dissimilar samples should be in different clusters." [49]

C'è da dire che comunque la natura dei dati ed il tipo di tecnica utilizzata ha un impatto significativo sulla definizione di cosa sia un Processo di clustering e quindi un buon Clustering. [16]

2.1.1 Applicazioni

Il Processo di clustering può essere visto come un processo esplorativo o un di pre-elaborazione dei dati utile per diverse attività di Machine

Learning e Data Mining: tra cui segmentazione delle immagini, recupero delle informazioni, riconoscimento dei pattern, classificazione dei pattern, analisi delle reti. [49], [19].

Il Processo di clustering può essere utilizzato anche per il rilevamento di valori anomali, infatti spesso tali valori anomali possono essere più interessanti dei casi comuni. [28]

Il Processo di clustering è utilizzato in molte applicazioni pratiche come la business-intelligence, l'image pattern recognition o l'image segmentation, nelle ricerche web, nella biologia e nella sicurezza. [28]

Caratterizzare La caratterizzazione dei dati è vista anche come un processo di *riepilogo delle caratteristiche generali* (in inglese Summarization of the general characteristics or features) ed è una componente fondamentle del trattamento dei dati. [28]

Classificare La classificazione, nel linguaggio comune, consiste in parole che ci aiutano a caratterizzare, riconoscere e discutere di eventi, oggetti, persone e concetti.

Ad esempio, i sostantivi sono essenzialmente etichette usate per descrivere una classe di oggetti con caratteristiche comuni: come gli animali che possono essere a loro volta gatti, cani, cavalli ecc. In questo contesto denominare, etichettare e classificare sono essenzialmente sinonimi che in inglese diventano naming, labeling e classification.

La classificazione risulta importante per l'apprendimento oltre per la caratterizzazione anche per l'opera di semplificazione, ma non sempre le classi sono conosciute a priori e quali caratteristiche si identificano in specifiche classi. [28]

Raggruppare Il Processo di clustering non è una tecnica di classificazione in senso stretto, ma permette l'estrazione di classi attraverso un processo comunque detto di etichettatura o classificazione dei dati con classi astratte non conosciute a priori. Per questo motivo è detta Classificazione non supervisionata. [16] [34] [49] [65]

"The classes are not known a priori but have to be discovered" [23]

In quest'ottica il Processo di clustering è una tecnica che permette di dividere in gruppi i set di dati allo scopo di riepilogare o migliorarne la comprensione associando agli aggregati etichette (categorie, classi) non sono conosciute ne imposte a priori che emergono dai dati stessi. [34] [65]

2.1.2 Definizioni

Di seguito sono dettagliate alcune definizioni di base relativamente dataset, item e Clustering.

Dataset, item e attributi I *dataset* sono gli oggetti di input su cui opera un generico Processo di clustering e sono intesi come collezioni di dati detti item, a loro volta dotati di diversi *attributi* o caratteristiche. Gli *item* sono anche detti più genericamente records, tuple o data-point.

Da un un punto di vista matematico un dataset è un un'insieme D composto da n item dotati di d attributi, quindi $D = \{x_1, x_2, ..., x_n\}$, dove un generico $i-esimo$ item è definito come un vettore di dimensione d, $x_i = (x_{i1}, x_{i1}, ..., x_{id})^T$.

Il numero d di attributi è detto dimensionalità del dataset [19].

I Cluster Il termine Cluster si riferisce ad un singolo gruppo di item che prende parte al Clustering, non ulteriormente divisibile.

Non esiste una definizione univoca per il termine Cluster che dipende fortemente dall'algoritmo e dal tipo di dati trattati [9]. Infatti spesso i termini *cluster, gruppi e classi* sono utilizati in maniera intuitiva senza che esista una definizione univoca e condivisa. [16]

Da un punto di vista formale una semplice definizione di Cluster è la seguente:

Dato un dataset D, $D = \{x_1, x_2, ..., x_n\}$ e $C(D)$ ed un algoritmo di Clustering, applicando quest'ultimo a D si ottiene una particolare riorganizzazione del dataset tale che:

$C(D) = \{C_1, ..., C_j\}$ dove $\forall j = [1 \rightarrow |D|]$ si ha che $C_j \subset D$. Tali C_j sono detti Cluster.

In base al tipo di algoritmo di Clustering possiamo avere che un generico elemento x_i può appartenere in maniera esclusiva ad un solo Cluster C_j oppure a più Cluster contemporaneamente. [19]

Un dataset si differenzia da un cluster per il solo fatto che primo è l'input di un Processo di clustering mentre il secondo è parte dell'output.

2.1.3 I metodi

In letteratura sono presentate diverse tenchice di Clustering e spesso possono sovrapposrsi tra loro, per questo moitvo non è possibile fonire un unica tassonimia che li caratterizzi nella loro completezza. [28].

L'obiettivo principale di un processo di Clustering è assegnare elementi con proprietà simili ad uno stesso cluster.

Il primo modo di classificare i metodi di Clustering è quello di suddividerli in base al tipo di riorganizzazione del dataset che sono in grado di realizzare (figura 2.2) , in tal senso i metodi di Clustering sono suddivisi in due categorie principali, l'Hard Clustering ed il Fuzzy Clustering (o Clustering morbido).

Nel caso dell'Hard Clustering, un item appartiene ad un solo Cluster, mentre nel caso del Fuzzy Clustering può appartenere a più di un Cluster. [19].

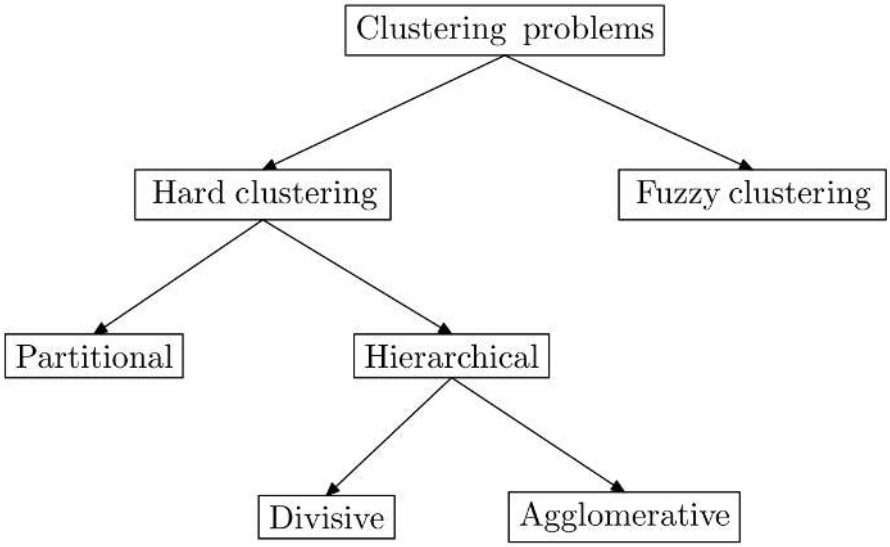


Figura 2.2: Classificazione dei metodi di clustering [19]

Tra le tipologie più studiate di Clustering troviamo il Clustering Partitivo ed il Clustering Gerarchico. A sua volta il Clustering Gerarchico si può suddividere in Agglomerativo (bottom-up) e Divisivo (top-down). [19]

Un'altro modo di categorizzare gli algoritmi di Clustering è quello proposto nel lavoro [49] che li suddivide in base al tipo di approccio utilizzato per riorganizzare il dataset, dividendoli in:

- Modelli Probabilistici
- Modelli Generativi
  - Basati sulla densità
  - Basati su griglia
  - Basati sulla distanza
    - Piatto
    - Gerarchico
      - Agglomerativo

- Divisivo

Una classificazione troppo rigida non rappresenta in modo adeguato il panorama dei medoti di Clustering perchè molti di essi utilizzando diverse tecniche contemporaneamente rendendo difficile posizionare un algoritmo in una categoria ben precisa.

2.2 Analisi di dati spaziali

L'analisi dei dati spaziali comprende tecniche di Clustering Spaziale, *Regionalization* e *Point Pattern Analysis*. Tutte tecniche che hanno come scopo comune l'estrazione o il supporto all'estrazione di nuova conoscenza da database spaziali. [52]

2.2.1 Dati spaziali

I dati spaziali sono una particolare categoria di dati dotati di relazioni e vincoli spaziali, un esempio possono essere i punti o gli oggetti presenti sulla superfice terrestre.

Essi rappresentano una sfida per il Data Mining e quindi anche per i processi di Clustering, che in questo caso specifico prendono il nome di Clustering spaziale.

Molti dati oggi sono georeferenziati, cioè dotati di attributi spaziali, ottenuti anche grazie ai servizi online basati sulla localizzazione. Dunque i dati georeferenziati sono diventati risorse di fondamentale importanza nella moderna società dell'informazione.

Il trattamento dei dati spaziali acquisisce così un'importanza strategica data la natura unica degli stessi. [71]

Gli oggetti spaziali, sono dotati sia di attributi spaziali che non-spaziali. Gli attributi spaziali di un oggetto includono spesso informazioni relative a posizione (es longitudine, latitudine, elevazione) oppure altri tipi di coordinate, alla forma o all'estensione. In base al tipo di dato possono contenere altri tipi di coordinate -spaziali- e descrivere diversi tipi di oggetti spaziali quali: punti, linee e poligoni o oggetti estesi. [71] [62]

Date le caratteristiche uniche dei dati spaziali, essi possono avere criticità specifiche quali precisione e accuratezza, dinamicità, sparsità (es. distribuzioe non omogenea), rumore o ridondanza [71].

Quando si parla di dati spaziali è facile pensare ad informazioni geo-localizzate, ma non sono le sole infatti in generale ci riferisce a qualsiasi informazioni con relazioni spaziali, oggetti puntuali o oggetti spazialmente estesi in uno spazio 2D, 3D o a più dimensioni come nel caso dei dati satellitari o del campo medico. [61] [25]

2.2.2 Clustering spaziale

Il Clustering Spaziale è la branca dei metodi di Clustering che tratta i dati spaziali, essa si sovrappone ed eredita in parte le delle tecniche classiche di Clustering, ma con alcune differenze dovute al campo di applicazione specifico.

La categorizzazione più citata in letteratura del Clustering Spaziale è quella che ricalca quella dei metodi di Hard Clustering, suddividendoli in:

- Partitivo
- Gerarchico
  - Agglomerativo
  - Divisivo

Un'altra classificazione degli algoritmi di Clustering spaziale che non sostituisce la precedente, ma la integra enfatizzando caratteristiche più specifiche dei dati spaziali è la seguente:

- Basati sulla distanza [49]
- Basati sulle griglie [28] [16]
- Basati sulla densità [28] [16]
- Basati su proprietà locali [45][10]
- Basati sulla distribuzione [45]

Anche nello Clustering spaziale si incontrano spesso sovrapposizioni e interoperabilità tra categorie dovuto all'utilizzo congiunto di tecniche diverse (figura 2.3) secondo cui gli algoritmi di Clustering si possono raggruppare in tre macro gruppi: Partitioning, Hierarchical e Locality. [45]

Classificazione che personalmente trovo molto interessante perché da un lato conserva la tassonomia classica e largamente condivisa degli algoritmi gerarchici e partitivi, ma allo stesso tempo pone enfasi ad una una nuova categoria relativa alle *proprietà locali*.

Per gli scopi del presente lavoro ci concentreremo sopratutto su gli algoritmi basati sulla densità e gerarchici.

In fine ho trovato interessante considerare nel contesto del Clustering spaziale due lavori che da un punto di vista spaziale considerano importanti sia gli ostacoli che altri tipi di vincoli nel Processo di clustering:

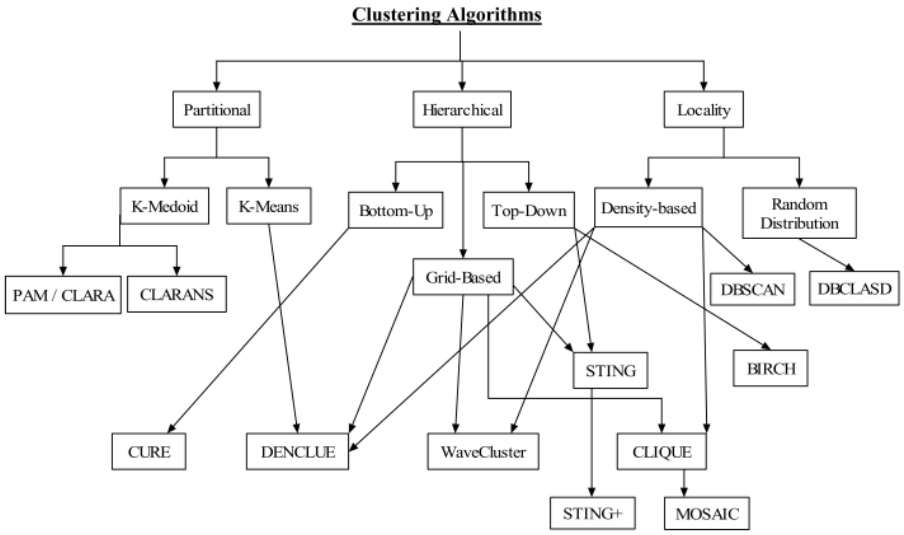


Figura 2.3: Classificazine degli algoritmi di Clustering alternativa, con enfasi alle prorpietà Locali [45]

- Spatial Clustering in the presence of obstacles [69]
- Clustering spatial data when facing physical constraints [75]

2.3 Clustering basato su densità

Secondo la categorizzazione della figura 2.3 gli algoritmi di Clustering basati sulla densità fanno parte della categoria *Locality-Based* dove gli item sono raggruppati secondo relazioni locali, dove in questa coso un particolare tipo di relazione locale è la *densità*.

2.3.1 Metodi basati sulla densità

Gli algoritmi basati sulla densità si caratterizzano per la loro capacità di idenfiticare cluster di forma arbitraria, definiti come regioni ad alta densità di punti distribuiti arbitrariamente, ben separati da regioni a bassa densità (figura 2.4). Tali algoritmi sono in grado di gestire in maniera efficiente il rumore e la presenza di punti anomali e non è necessario conoscere a priori il numero di cluster da identificare abbassando così il grado di conoscenza del dominio, necessaria per utilizzare efficacemente uno di tali medoti. [19] [28] [45] [47] [14]

Un ulteriore caratteristica è data dal fatto che i cluster identificati possono essere sia convessi che concavi, cosa che non riescono a fare altrettanto bene gli algoritmi basati su centroidi che sono più adatti invece a identificare cluster di forma tendenzialmente circolare (convessi). [28]

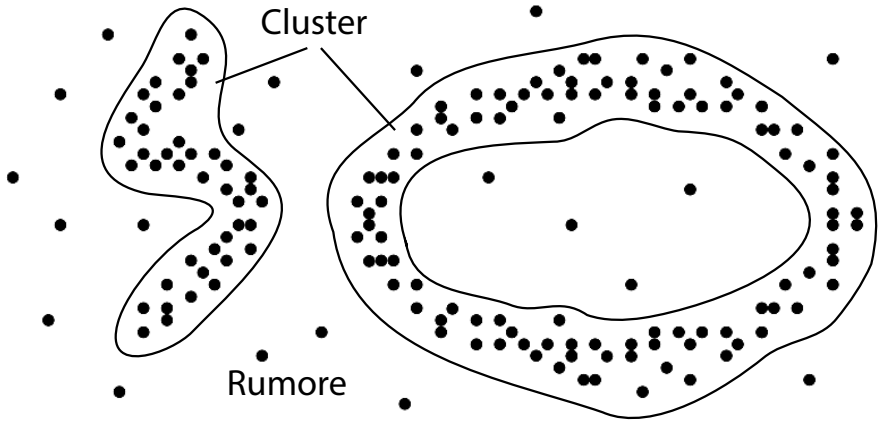


Figura 2.4: cluster di punti ad alta densità e rumore di punti a bassa densità [28]

Una delle criticità, invece, risiede nella nella scelta dei parametri di configurazion dell'algoritmo, come può essere ad esempio la soglia di densità. [19]

I metodi basati sulla densità non conosiderano i metodi Fuzzy, ma solo i metodi di Hard clustering. Essi si suddividono in Partitivi o Gerarchici e la maggior parte degli approcci basati sulla densità sono stati sviluppati specificatamente per il clustering spaziale, essendo la *densità* una proprietà tipicamente spaziale. [19] [28]

Alcuni tra gli algoritmi di Clustering basati sulla densità sono [49] [28] [19]:

- DBSCAN [14]
- DBCLASD
- OPTICS [2]
- DENCLUE [31]
- BRIDGE [19]
- CUBE

Tra di essi il più popolare è il DBSCAN [14] che secondo la maggior parte della letteratura è considertato il primo algoritmo basato sulla densità ad essere stato introdotto. [42]

Mentre secondo il lavoro [60] emerge che algortmi di Clustering basati sulla densità sono stati già esplorati da Wishert nel 1969. [72]

Il DBSCAN può vantare numerose alternative e varianti rappresentando uno degli approcci basato sulla densità più rilevanti nell'ambito del clustering spaziale. [52]

2.3.2 Definizioni di base

La *densità* è una proprietà tipicamente spaziale ed è alla base degli algoritmi basati sulla densità, i quali dividono lo spazio in cluster definiti come aree ad alta densità e ruomore come aree a bassa densità. [14]

La densità, da un punto di vista puramente spaziale, è intesa come il rapporto tra lo spazio ed il numero di elementi presenti nello spazio considerato.

Definizione di Densità Nel contesto degli algoritmi basati sulla densità, essa è intesa come il rapporto tra l'area circoscritta in una circonferenza di raggio r e di centro in p_0, ed il numero di punti $p_1, ..., p_n$ presenti al suo interno.

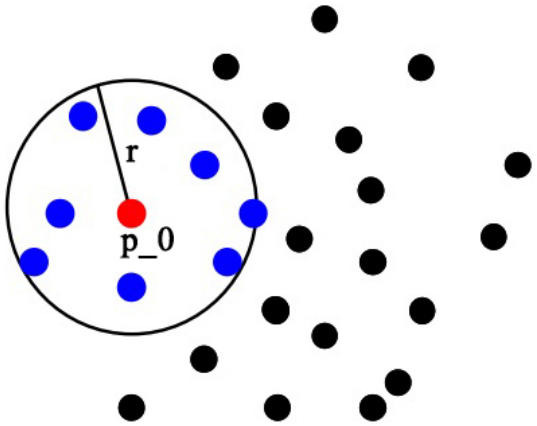


Figura 2.5: Densità di punti rispetto alla circonferenza di raggio r e di centro in p_0 [68]

Il raggio r ed il numero n di punti $p_1, ..., p_n$ rappresentano i parametri di densità per il Clustering basato sulla densità, tali parametri sono identificati in letteratura sotto le voci *eps* (raggio r), $MinPts$ (numero di punti n), dove tipicamente $MinPts \in \mathbb{N}_{>0}$ e $Eps \in \mathbb{R}_{>0}$. [14] [13] [19] [28] [68]

Soglia di Densità I parametri *eps* e $MinPts$ sono detti anche Soglia di Densità ed a partire da essi è possibile definire i concetti di base degli algoritmi basati sulla densità attraverso i concetti illustrati di seguito.

Eps-Neighborhood: Tradotto in *vicinato* rispetto a *eps* del punto p_i, ovvero tutti quei punti che sono distanti meno del valore *eps* rispetto al punto considerato.

Dato D un dataset di punti, $D = \{p_0, ..., p_k\}$. Preso un punto $p_i \in D$, il vicinato N_{eps} di p_i rispetto a eps è definito come: $N_{eps}(p_i) = \{p_j \in D,\ p_i \neq p_j \mid \forall j,\ distance(p_j, p_i) < eps\}$

Core-Point: Detto anche *punto centrale*, ovvero quel punto tale che il numero di *vicini* supera il valore $MinPts$. Preso un punto $p_i \in D$, si dice che p_i è un Core-Point se $|N_{eps}(p_i)| \geq MinPts$

Directly Density-Reachable: Si dice che q Directly Density-Reachable da p_i se valgono le seguenti proprietà:

- $q \in N_{eps}(p_i)$
- $|N_{eps}(p_i)| \geq Min_{eps}$

Dato un punto q di tipo *directly density-reachable* da p_i, q può essere ancora *Core-Point* o *border point*.

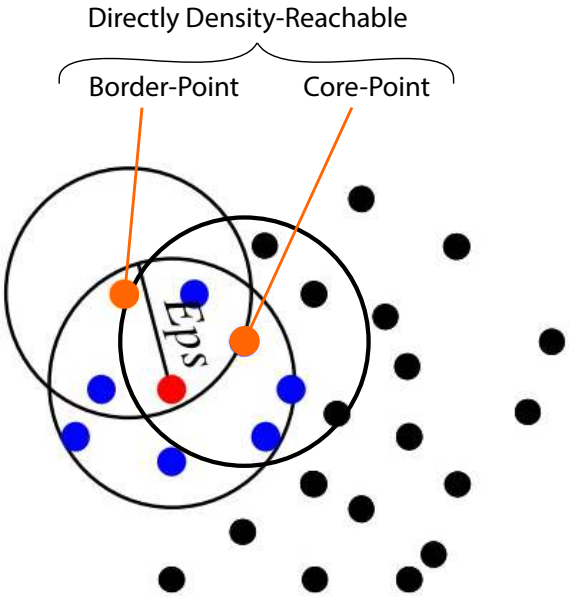


Figura 2.6: Directly Density-Reachable: Core-Point e Border-Point rispetto al punto p_i ed eps [68]

Border-point: Si dice q è border-point di p_i se $|N_{eps}(p_i)| < MinPts$

Density-Reachable: Un punto q, si dice Density-Reachable da un punto p se $\exists$ una sequenza di punti $p_1 \ldots, p_i, x_{i+1} \ldots, p_n$, con $p = p_1$ e $p_n = q$ dove $i \geq 1$ e $n \geq 2$, si ha che $\forall i \leq n$ p_i tale che p_{i+1} è Directly density-reachable da $p_i \in D$ [68]

Density-connected: Due punti p, q si dicono Density-Connected se $\exists$ un punto z se sia p che q sono Density-Reachable da z.

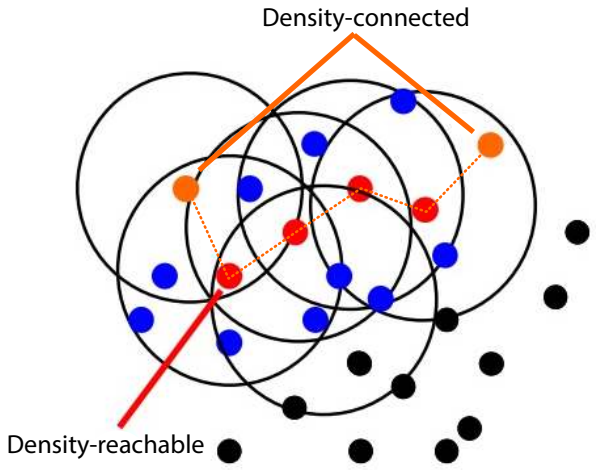


Figura 2.7: Proprietà di Density-Connected e Density-Reachable dei punti [68]

Cluster: Dato D un dataset di punti. Il Cluster C_j, rispetto ai valori Eps e $MinPts$, è un sottoinsieme non vuoto di D che rispetta le seguenti proprietà:

1. Maximality: $\forall\ p, q\ :\ se\ p \in C$ e q è densamente-raggiungibile (*Density-Reachable*) da p, rispetto ai valori Eps e $MinPts$, allora $q \in C$

2. Connectivity: $\forall\ p, q \in C$ p è densamente-connesso (*Density-Connected*) a q, rispetto ai valori Eps e $MinPts$.

Rumore: Dati $C_1, \ldots, C_k$ Cluster validi, si definiscono rumore quei punti che non fanno parte di nessun Cluster, ovvero:

$$NOISE_{Eps,MinPts} = \{p_i : p_i \in D,\ \forall\ k\ \notin C_k\}$$

Distanza tra Clusters Un modo per calcolare la distanza tra Cluster è quella di considerare la distanza più breve tra i punti facenti parte dei due Cluster

Ovvero, dati due Cluster C_1, C_2 la distanza tra Cluster è:

$distanza_{cluster}(C_1, C_2) = min(distanza_{punti}(x_i, y_j))$ tale che $x_i \in C_1$ e $x_j \in C_2$. [19]

2.3.3 DBSCAN

Il *DBSCAN* è l'algoritmo di Clustering basato sulla densità che raggruppa i punti dello spazio in Cluster di forma arbitraria ad alta densità separati da aree a bassa densità. [14]

Il DBSCAN è considerato il primo algoritmo legato al concetto di densità, attraverso il quale è stato presentato anche il concetto di densità per il Clustering spaziale. [52] [14]

L'algoritmo è composto da due componenti principali:

- `DBSCAN()`: che si occupa di scorrere l'intera lista di punti per identificarne i *Core-Point*
- `ExpandCluster()`: richiamata dalla prima, si occupa di espandere i Core-Point attraverso la ricerca di punti che godano delle proprietà *Density-Reachable* e *Density-Connected*

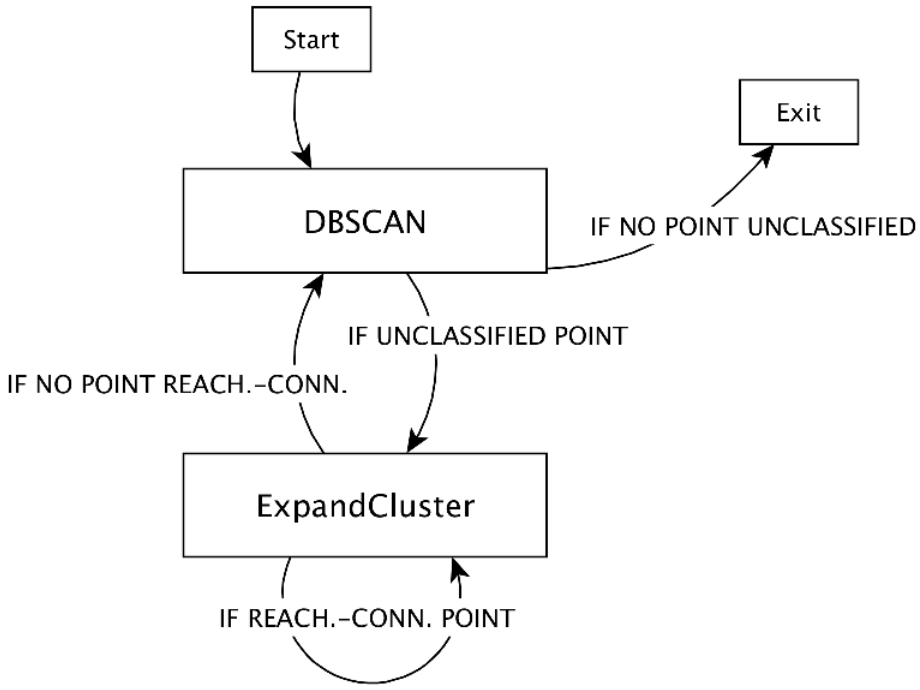


Figura 2.8: Componenti di massima dell'algoritmo DBSCAN

Ricerca dei Core-Point

La prima componente, nella sua versione base, scorre l'intero dataset di punti cercando quelli che non sono stati classificati ne come rumore ne come facenti parte di un Cluster, nel caso non siano stati ancora

classificati (`UNCLASSIFIED`) vengono passati in input alla seconda componente denominata `ExpandCluster()`.

L'i-esimo punto non ancora classificato è un potenziale Core-Point oppure sarà rumore, nel caso rispettasse la proprietà dei Core-Point la componente `ExpandCluster()` continuerà ad espandere il fronte del Cluster a partire dal Core-Point finché è in grado di identificare aree densamente popolate. In caso contrario la componente `ExpandCluster()` terminerà e verrà passato nuovamente il controllo alla componente principale `DBSCAN()`.

```
BDSCAN(Dataset, Eps, MinPts): ClusterId ← nextId(NOISE);
for i ← 1 to Dataset.size do
    Point ← Dataset.get(i);
    if Point.ClId == UNCLASSIFIED then
        if ExpandCluster(Dataset, Point, ClusterId, Eps, MinPts) then
            ClusterId ← nextId(ClusterId);
        end
    end
end
```

Algoritmo 1: Ricerca dei Core-Point dell'algoritmo DBSCAN [14]

Estensione dei cluster

La seconda componente del DBSCAN è il cuore dell'algoritmo ovvero quella componente che costruisce il Cluster a partire da un Core-Point, oppure nel caso il non fosse un Core-Point viene classificato come rumore.

```
ExpandCluster(Dataset, Point, ClId, Eps, MinPts):
seeds ← Dataset.regionQuery(Point, Eps);
if seeds.size < MinPts then
    Point.CiId ← NOISE;
    return False;
else
    seeds.changeAllClIds(ClId);
    seeds.delete(Point);
    while seeds isNot Empty do
        currentP ← seeds.first();
        result ← Dataset.regionQuery(currentP, Eps);
        if result.size ≥ MinPts then
            for i ← 1 to result.size do
                resultP ← result.get(i);
                if resultP.ClId IN {UNCLASSIFIED, NOISE} then
                    if resultP.CiId = UNCLASSIFIED then
                        seeds.append(resultP);
                    end
                    resultP.changeAllClIds(CiId);
                end
            end
        end
        seeds.delete(currentP);
    end
end
```

Algoritmo 2: Estensione del Cluster a partire da un Core-Point [14]

Complessità L'algoritmo di Clustering *DBSCAN* nella versione ha una complessità che dipende principalmente dalla funzione *Dataset.regionQuery(Point, Eps)* che effettua la ricerca dei *vicini* rispetto a *Point*.

Il caso peggiore si ha quando *tutti i punti del dataset sono segnati come rumore*, in tal caso il l'algoritmo è tenuto ha eseguire *n-volte* la funzione *Dataset.regionQuery(Point, Eps)* (algoritmo 2) che a sua volta, senza alcuna ottimizzazione, dovrà inevitabilmente calcolarsi la distanza da *Point* per ogni punto del dataset, per una complessità totale del *DBSCAN* pari a $O(n^2)$.

Gli stessi autori dell'algoritmo originale [14] propongono una soluzione per ottimizzare il costo computazionale dell'algoritmo attraverso la rappresentazione di *Dataset* con un *R-Tree*. Soluzione che nel caso peggiore porta la funzione *Dataset.regionQuery(Point, Eps)* ad avere una complessità di $O(\log n)$, per un totale per il DBSCAN di $O(n \log n)$. [14]

Ma quest'ultima soluzione è praticabile solo per dataset non troppo grandi, dunque per dataset molto grandi, l'algoritmo ha una complessità di $O(n^2)$. [14] [13] [49]

Limiti e vantaggi

La bontà degli algoritmi di Clustering generalmente si misura in termini di robustezza, efficienza (tempo e memoria), accuratezza e quantità di conoscenza pregressa del dominio necessaria per una buona applicazione. [74]

Limiti Diverse criticità del DBSCAN emergono dalla letteratura ed esse rappresentano le principali motivazioni per le varianti proposte in letteratura, come si può vedere dal confronto illustrato nella figura 2.9 relativamente ad alcune varianti del DBSCAN rispetto alla miglioria apportata. [42] [1]

Le limitazioni più citate sono:

- La complessità è pari a $O(n^2)$ senza ottimizzazioni e per dataset di grandi dimensioni; [14]
- L'efficienza limitata nell'identificare Cluster i dataset con densità molto diversificata; [51]
- Il consumo di memoria elevato; [14]
- L'intrinseca difficoltà nello scegliere la combinazione ottimale dei parametri di input Eps e $MinPts$; [63]

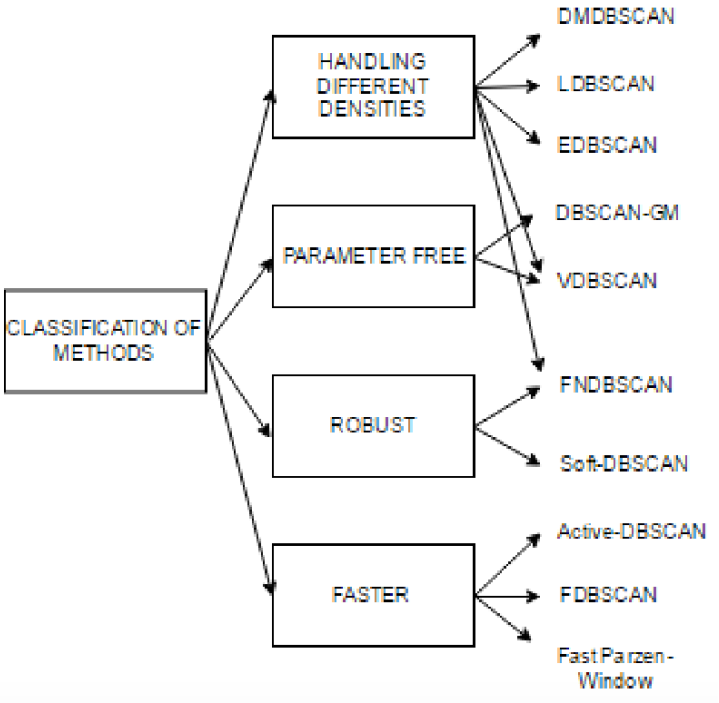


Figura 2.9: Confronto fra varianti del DBSCAN[74]

Vantaggi: Il DBSCAN è l'algoritmo più popolare tra gli approcci di Clustering spaziale, questo è dovuto principalmente per le sue potenzilità, ad esempio più volte è stato detto che uno dei maggiori vantaggi dell'utilizzo del DBSCAN è senza dubbio la sua capacità di identificare Cluster di forma arbitraria mentre altri vantaggi possono essere: [44] [1]:

- Il numero dei Cluster da identificare non è necessario conoscerlo in anticipo;
- La capacità di identificare in modo efficiente rumore e valori anomali;

2.3.4 OPTICS

L'algoritmo OPTICS non è un algoritmo di Clustering in senso stretto, ma è una tecnica di ordinamento dei Cluster basata sulla densità, tale tecnica è utilizzata in diversi approcci per il Clustering. [2]

OPTICS non produce praticamente un Clustering, ma crea un ordinamento del dataset (figura 2.10) che enfatizza la struttura dei Cluster basati sulla densità. Questo ordinamento dei Cluster contiene informazioni corrispondenti ad un ampio intervallo di impostazioni dei parametri rappresentando una base versatile per l'analisi Cluster

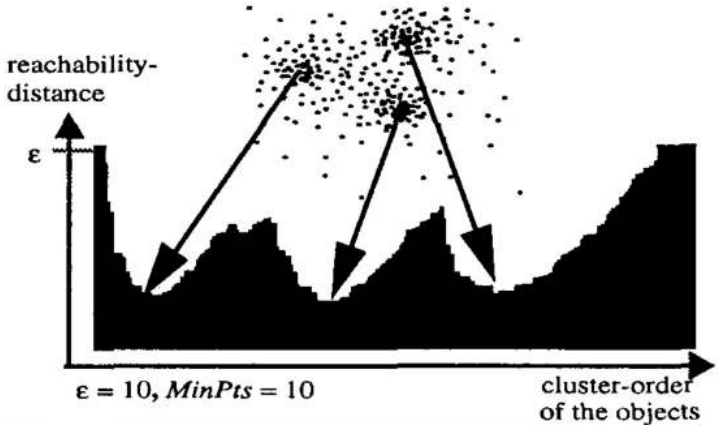


Figura 2.10: Ordinamento dei punti per densità realizzato dall'algoritmo OPTICS [2]

automatica.

OPTICS è proposto come tecnica esplorativa di pre-elaborazione per un processo di Clustering vero e proprio, cercando di andare in soccorso ai principali principali degli algoritmi di Clustering, ovvero che:

- quasi tutti gli algoritmi di Clustering richiedono parametri di input che nella maggior parte dei casi sono difficili da determinare
- tali parametri spesso hanno un'influenza significativa sul risultato del processo di Clustering;
- in molti casi non esiste un unico parametro globale in grado di descrivere in modo corretto e significativo il dataset;

2.3.5 DENCLUE

DENsity-based CLUstEring (DENCLUE) è un algoritmo di Clustering basato sul metodo del Kernel Density Estimation (KDE) applicabile a dataset di grandi dimensioni. [31]

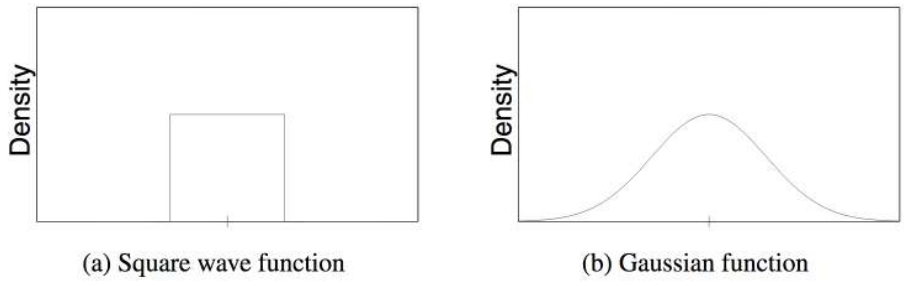


Figura 2.11: Funzioni kernel quadra e gaussiana [31]

Il KDE è un metodo di stima di densità che si basa sull'idea che l'influenza di ogni punto può essere modellata formalmente usando una funzione matematica, chiamata in questo caso funzione kernel.

La funzione del kernel può essere vista come una funzione che descrive l'influenza di un punto rispetto ai suoi vicini. Esempi di funzioni kernel possono essere le funzioni paraboliche, quadre o gaussiane (figura 2.11).

Tali funzioni kernel vengono applicate a ciascun punto allo scopo di calcolare la stima della densità complessiva dello spazio che può essere calcolata come la somme delle influenze di tutti i punti (figura 2.12), i Cluster possono essere determinati matematicamente identificando i cosiddetti *attrattori densità* ovvero i massimi locali della funzione di densità complessiva.

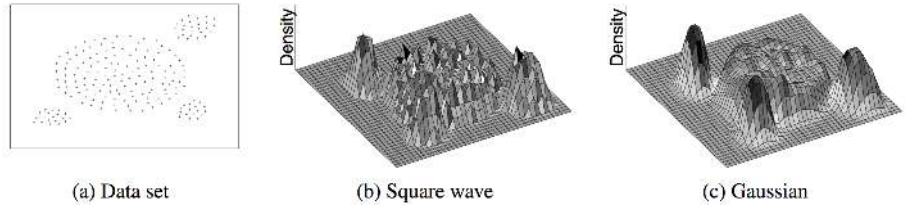


Figura 2.12: Influenza dei punti rispetto all'utilizzo di funzioni kernel quadre o gaussiane [31]

2.4 Clustering Gerarchico

Una delle categorizzazioni più popolari degli algoritmi di Clustering è quella che considera la struttura in cui sono organizzati i Cluster identificati. In questo senso si suddividono in Clustering Partitivo (anche chiamato Flat Clustering) e Clustering Gerarchico.

Nel caso del Clustering Partitivo si dice che identificano Cluster esclusivi ovvero si ottiene un unica ripartizione del dataset in Cluster ognuno dei quali ben separato dall'altro, mentre nel caso del Clustering Gerarchico si ottiene una riorganizzazione del dataset in una struttura gerarchica.

Definizione Un algoritmo di Clustering Gerarchico organizza gli oggetti in una struttura gerarchica, dividendo o raggruppando il set di dati in una sequenza di partizioni annidate, struttura che viene rappresentata per mezzo un albero tipicamente binario attraverso diverse notazioni, dove ogni livello è una particolare riorganizzazione del dataset (figura 2.13).

Tali algoritmi costruiscono la gerarchia attraverso due tecniche principali che danno il nome a due ulteriori categorie: Algoritmi di

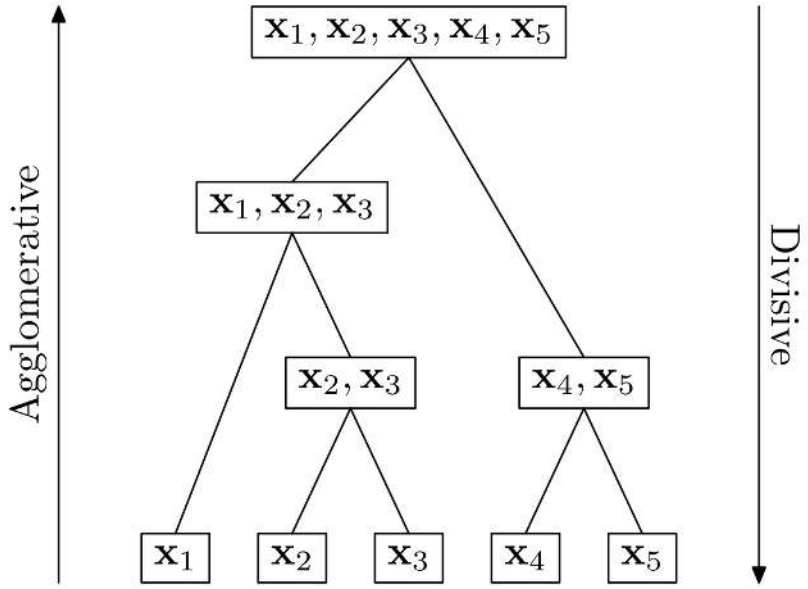


Figura 2.13: Confronto tra Clustering Gerarchico Agglomerativo e Divisivo [19]

Clustering Gerarchici di tipo Divisivo oppure di tipo Agglomeratico. [19]

Irrevocabilità Indipendentemente se l'algoritmo di Clustering Gerarchico è di tipo di Divisivo o Agglomerativo, essi sono caratterizzati dal fatto di essere metodi *irrevocabili*, ovvero, una volta intrapresa una specifica divisione o raggruppamento, essa è irrevocabile a meno di non ripetere il procedimento.

Così che quando un algoritmo agglomerativo unisce due oggetti, essi non possono essere successivamente separati ed ovviamente quando invece un algoritmo divisivo applica una divisione, essa non può essere annullata. [19] [41]

Come citato da Kaufman and Rousseeuw: *"Un metodo gerarchico soffre del fatto che non può mai riparare ciò che è stato fatto nei passaggi precedenti. Questa rigidità dei metodi gerarchici è la loro chiave di successo e insuccesso allo stesso tempo" [41]*

Il metodo di divisione o raggruppamento è quindi di fondamentale importanza, perché una volta che un gruppo di oggetti viene unito o diviso, il processo nel passaggio successivo dipenderà dai Cluster identificati dal passaggio precedente. Quindi se il processo non è adeguato si possono avere organizzazioni del dataset di bassa qualità. [28]

In generale le tecniche gerarchiche non sono intese come in concorrenza rispetto ai metodi di Clustering Partitivo poiché non perseguono lo stesso obiettivo, descrivendo i dati in un modo completamente diverso. Sono quindi visti più come tecniche alternative che in competizione. [41]

2.4.1 Metodo Agglomerativo e Divisivo

Clustering Gerarchico Agglomerativo I metodi agglomerativi costruiscono la struttura ad albero considerando inizialmente i singoli item del dataset come singoli Cluster, i quali vengono successivamente uniti iterativamente a formare Cluster sempre più grandi (figura 2.13), fino a quando tutti gli item non si trovano in un singolo Cluster finale oppure fin quando determinate condizioni non sono soddisfatte. Tale approccio di raggruppamente a partire dagli item singoli è detto di tipo bottom-up perchè effettua la clusterizzazione dal basso. [19] [28])

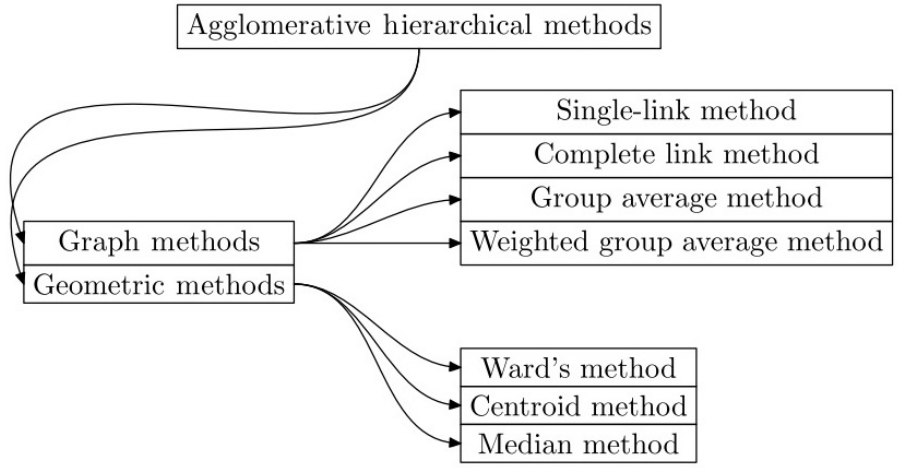


Figura 2.14: Metodi di Clustering Gerarchico Agglomerativo [19]

Esiste una ulteriore differenziazione degli algoritmi di Clustering Agglomeriativi che dipende dal metodo applicato durante la cosiddetta *fase fusione dei Cluster* che rappresenta il campo di maggiore interesse nella letteratura per questo tipo di algoritmi.

Una buona classificazione degli algoritmi di Clustering Gerarchico Agglomerativi è fornito da Murtagh nella figura 2.14.

Clustering Gerarchico Divisivo I metodi divisivi costruiscono invece la struttura ad albero considerando inizialmente tutti gli *item* del dataset come facenti parte di un unico cluster, il quale diverrà la *radice* del'albero di cluster (figura 2.13).

Il Cluster iniziale sarà diviso iterativamente sino a formare Cluster sempre più piccoli, fino a quando tutti gli item non si troveranno in Cluster separati, oppure quando non saranno soddisfatte determinate condizioni di terminazione come la coesione tra gli item dei cluster, similitudine degli item o numero di Cluster identificati in un dato livello dell'albero.

Tale approccio di Clustering per dvisione a partire dalla totalità degli item, è detto top-down perché effettua il Clustering dall'alto. [19] [28]

2.4.2 Metodi Divisivi Monothetic e Polithetic

I metodi di Clustering Gerarchico Divisivi sono principalmente di due tipi: *monothetic* e *polithetic*. [16]

I metodi Monothetic usano un singolo parametro (o descrittore) ad ogni passo del partizionamento, considerato come il migliore per un dato livello; mentre i metodi Polithetic usano diversi parametri (o descrittori) simultaneamente per effettuare la divisione dei Cluster. [50]

2.4.3 Clustering Gerarchico tradizionale e non tradizionale

Tradizionalmente un algoritmo di Clustering Gerarchico è caratterizzazione da un algoritmo ripetitivo che a ogni step divide aggiungendo un diminuendo di un Cluster in base al tecnica utilizzata (diviso o agglomerativo). Questo in letteratura è descritto come il metodo tradizionale di intendere il Clustering Gerarchico basandosi sul concetto di distanza tra Cluster, caratteristiche che portano alla tipica rappresentazione in albero binario dei cluster. [28] [16]

2.5 Distanze, Similarità e Dissimilarità

Di fondamentale importanza nell'analisi di Clustering è l'osservazione e la quantificazione di quanto due oggetti sono *vicini* (o distanti) tra loro, sia che essi sono dei Cluster sia che degli item. La letteratura offre diverse definizioni e formule in grado di quantificare la vicinanza di due oggetti, ma in generale ci si riferisce ad esse come formule (anche coefficiente o metrica) di prossimità, similarità, dissimilarità o più in generale con distanze.

Esse si suddividono essenzialmente nel modo di quantificare tali osservazioni ed il tipo e la natura degli oggetti osservabili tramite esse. [16] [19].

Definizione di base Due oggetti sono vicini (o simili) quando la loro distanza (o diversità) è piccola, o la loro somiglianza è elevata. [9]

La distanza, in senso lato, è un concetto che è alla base dell'esperienze umane di tutti i giorni e generalmente riguarda il grado di vicinanza tra due oggetti fisici misurata ad esempio in lunghezze, intervalli di tempo o spazio, indici di differenze. [9]

Se risulta immediato e radicato nel senso comune il concetto di distanza tra due città o intervallo di tempo, risulta invece più complicato e meno immediato la somiglianza (o distanza) di oggetti complessi, per questi ultimi è più comune utilizzare i termini similarità o dissimilarità.

Esistono diverse tecniche di misura della distanza in base alla natura degli oggetti da a analizzare, ad esempio possiamo avere oggetti puntuali o caratterizzate da un estensione.

2.5.1 Distanza puntuale tra oggetti

Di base consideriamo la somiglianza o la distanza tra due oggetti in termini puntuali, di seguito alcune definizioni. [9]

Distanza Dato un insieme X, la funzione $d : X \times X \to \mathbb{R}$ è chiamata distanza (o dissimilarità) su X se, $\forall x, y \in X$ si hanno le seguenti proprietà:

- Non negatività: $d(x, y) \geq 0$
- Simmetria: $d(x, y) = d(y, x)$
- Riflessione: $d(x, x) = 0$

Similarità Un coefficiente di similarità indica la forza della relazione tra due oggetti e ne esistono di diversi. Più due oggetti dati si assomigliano, maggiore è il coefficiente di similarità, che è generalmente compleso tra 0 *e* 1. [16] [19]

Sia X un insieme ed x, y due oggetti complessi rale che $x = (x_1, x_2, ..., x_d)$ e $y = (y_1, y_2, ..., y_d)$. Una funzione $s : X \times X \to \mathbb{R}$ è chiamata somiglianza su X o anche coefficiente di somiglianza tra x e y su X, una funzione $f(y, x) = s(x, y) = s(x_1, x_2, ..., x_d, y_1, y_2, ..., y_d)$ [16] dotata delle seguenti proprietà:

- $s(x, y) \geq 0$ (non negatività)
- $s(x, y) = s(y, x)$ (simmetria)
- $0 \leq s(x, y) \leq s(x, x)$ con $\forall x, y \in X$, con $s(x, y) = s(x, x) = 1$ solo se $x = y$

Metrica La funzione $d_{metric} : X \times X \to \mathbb{R}$ è chiamata metrica se $\forall x, y, z \in X$ rispetta le proprietà: [9]

- Non negatività: $d(x, y) \geq 0$.
- *Identity of indiscernibles*: $d(x, y) = 0$ se e solo se $x = y$
- Simmetria $d(x, y) = d(y, x)$
- Disuguaglianza Triangolare: $d(x, y) \leq d(x, z) + d(z, y)$

2.5.2 Distanza tra Cluster

Per esprimere distanze o similarità tra due oggetti complessi o caratterizzati da un estensione è necessario applicare formule diverse da quelle utilizzate per oggetti atomici puntuali, e questo è il caso del calcolo della distanza o la similarità tra Cluster o insiemi di oggetti.

Dati due Cluster $C_1 = \{y_1, y_2, ..., y_r\}$, $C_2 = \{z_1, z_2, ..., z_s\}$ sono due insiemi di oggetti di dimensione r, s, è possibile calcolare la loro distanza o similarità attraverso diverse strategie, le più popolari in letteratura sono: [19]

- Distanza basata sul punto mediano
- Distanza media dei punti
- Distanza maggiore dei punti
- Distanza minima dei punti

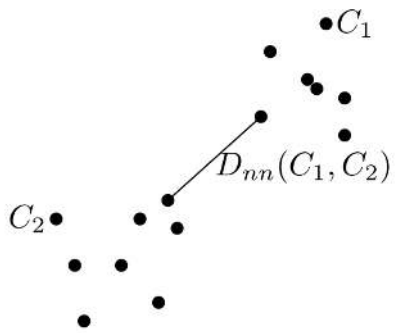
Distanza basata sul punto Mediano Un metodo piuttosto popolare in letteratura per calcolare la distanza tra due Cluster è considerare la distanza rispetto i rispetivi punti *mediani* (o centroidi): [19]

$$D_{mean}(C_1, C_2) = d(\mu(C_1), \mu(C_2)) \tag{2.1}$$

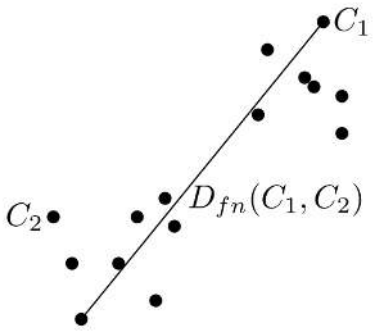
$$\mu(C_j) = \frac{1}{|C_j|} \sum_{x \in X} x, \ j = 1, 2 \tag{2.2}$$

Distanza media dei punti Un'altro metodo per calcolare la distanza tra Cluster è quella di calcolare la distanza tra tutti i punti dei due Cluster e quindi calcolarne la media:

$$D_{avg} = (C_1, C_2) = \frac{1}{rs} \sum_{i=1}^{r} \sum_{j=1}^{s} d(y_i, y_j) \tag{2.3}$$



Nearest neighbor distance between two clusters.



Farthest neighbor distance between two clusters.

Figura 2.15: Confronto tra distanza tra Cluster: tra i punti più vicini (in alto) e i punti più lontani (in basso) [19]

Distanza maggiore dei punti Un approccio differente può essere quello di considerare solo i punti più distanti:

$$D_{fn} = (C_1, C_2) = \max_{1 \leq i \leq r,\ 1 \leq j \leq s} d(y_i, z_j) \tag{2.4}$$

Distanza minima dei punti In fine l'approccio opposto è quello di considerare la distanza tra i punti più vicini:

$$D_{nn} = (C_1, C_2) = \min_{1 \leq i \leq r,\ 1 \leq j \leq s} d(y_i, z_j) \tag{2.5}$$

2.5.3 Lance-Williams Formula

Un utilizzo particolare delle distanze tra Cluster è quello applicato negli algoritmi di Clustering Gerarchico Agglomerativo, i quali hanno la necessità di calcolare la distanza tra i vecchi Cluster ed il nuovo Cluster formato a sua volta a due cluster.

Tale approccio è risolto da Lance e Williams nel 1967 [19] che propongono una formula di ricorrenza la quale fornisce la distanza tra un Cluster C_k e un Cluster C formato dalla fusione dei Cluster C_i e C_j, $C = C_i \cup C_j$. [19]

$$D(C_k, C_i \cup C_j) = \alpha_i D(C_k, C_i) + \alpha_i D(C_k, C_j) + \beta D(C_i, C_j) + \gamma |D(C_k, C_i) - D(C_k, C_j)| \tag{2.6}$$

Scegliendo opportunamente i parametri $\alpha_i, \beta_j, \gamma$ si possono ottenere diverse distanze tra Cluster annidati, soprattutto utilizzati nel Clustering Gerarchico Agglomerativo com'è possibile osservare nella figura 2.16 che mette a confronto diversi parametri e formule.

Algorithm	α_i	α_j	β	γ
Single-link	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{-1}{2}$
Complete link	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$
Ward's method	$\frac{n_i+n_j}{\Sigma_{ijk}}$	$\frac{n_j+n_k}{\Sigma_{ijk}}$	$\frac{-n_k}{\Sigma_{ijk}}$	0
Group average	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	0	0
Weighted group average	$\frac{1}{2}$	$\frac{1}{2}$	0	0
Centroid	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$\frac{-n_i n_j}{(n_i+n_j)^2}$	0
Median (weighted centroid)	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{-1}{4}$	0

Figura 2.16: Parametri per *Lance-Williams Formula* [19]

2.6 Approcci alternativi basati sulla densità

In letteratura sono disponibili un gran numero di algoritmi basati sulla densità alternativi al DBSCAN, alcuni basati su quest'ultimo, mentre altri completamente nuovi.

Le criticità del DBSCAN possono essere riassunte in tre punti: [42] [40]

- densità multiple: l'impossibilità di identificare Cluster con densità diverse;
- costo computazionale: il costo computazionale del DBSCAN non è lineare;
- scelta dei parametri: alta sensibilità del risultato rispetto ai parametri di input e difficoltà nella loro scelta;

Nelle sezioni successive sono descritte alcune soluzioni che ho considerato interessanti in termini di approccio alla risoluzione delle principali criticità del DBSCAN. Le tecniche successive si basano su:

- l'utilizzo di griglie;
- la costruzione di gerarchie di Cluster;
- l'identificazione di picchi di densità;

2.6.1 Griglie

Una delle soluzioni per risolvere il problema della *densità multipla* e della *scelta dei parametri* consiste nell'utilizzo di griglie che suddividono lo spazio in parti discrete, rendendo possibile l'identificazione di Cluster con densità diverse.

Nelle sezioni successive saranno descritti due algoritmi che a mio avviso illustrano in modo semplice come l'utilizzo delle griglie possa permettere di estrarre Cluster di densità diverse:

- GRIDBSCAN; [73]
- GMDBSCAN; [70]

GRIDBSCAN

L'algoritmo GRIdDensity-Based Spatial Clustering of Applications with Noise (GRIDBSCAN) proposto nel lavoro [70] propone una soluzione per l'identificazione di Cluster con densità diverse attraverso l'applicazione di un Processo di clustering su tre step:

- nel primo step viene creata la griglia appropriata in modo che la densità sia omogenea in ciascuna cella;
- nel secondo step vengono unite le celle con densità simili e identifica i valori più adatti di Eps e $MinPts$ in ogni griglia ottenuta con la fusione;
- nel terzo step viene applicato l'algoritmo DBSCAN con i parametri Eps e $MinPts$ identificati per ogni griglia ottenuta dalla fusione;

Il metodo così proposto risulta più accurato del DBSCAN base, ma computazionalmente più costoso.

GMDBSCAN

L'algoritmo Multy Density DBSCAN Cluster Based on Grid (GMDBSCAN) proposto nel lavoro [73] suggerisce di migliorare la scelta dei parametri di input su dataset a densità multipla attraverso l'applicazione di una griglia e di un SP-Tree.

L'algoritmo GMDBSCAN divide inizialmente lo spazio in una griglia attraverso il quale ottiene i parametri *Eps* ed $MinPts$ medi della griglia per ogni cella successivamente applica il DBSCAN alle singole celle e successivamente effettua il merge dei Cluster identificati sulla base della della loro similarità.

2.6.2 Gerarchie

L'algoritmo DBSCAN proposto in [14] organizza i punti del dataset in un Clustering piatto (flat) in cui tutti i Cluster possiedono la stessa densità minima, ma l'area centrale di un Cluster è un'area più densa rispetto ai bordi dello stesso Cluster, dunque è possibile che esistano densità diverse in aree diverse dello stesso Cluster.

Molte applicazioni degli algoritmi necessitano di Clustering piatto anche se basato su gerarchie e algoritmi gerarchici basati sulla densità come OPTICS [2] organizzano lo spazio in una gerarchia basata sulla densità suggerendo solo un modo per *tagliare l'albero* ad un unica soglia di densità. [7]

Basandoci su queste criticità diventa chiaro che l'utilizzo e la costruzione di gerarchie basate sulla densità sono esplorare dalla letteratura, senza però risolvere in modo esaustivo tutte le problematiche degli algoritmi basati sulla densità oggi disponibili.

É in questo contesto che si posiziona l'algoritmo Hierarchical DBSCAN (HDBSCAN) ed la rispettiva variante Hierarchical DBSCAN with Excess Of Mass (HDBSCAN-EOM).

HDBSCAN

L'algoritmo Hierarchical DBSCAN (HDBSCAN) [7] [8] è un Processo di clustering basato sulla densità che eredita in parte dal DBSCAN [14] solo le definizioni di *densità*.

Mentre è inteso come un miglioramento algoritmico rispetto al metodo di ordinamento OPTICS [2].

HDBSCAN realizza una gerarchia basata sulla densità rispetto ad un unico valore di input che chiama m_{pts} (equivalente a $MinPts$ della sezione 2.3.2), attraverso il quale viene estratta una gerarchia semplificata composta solo dai Cluster più significativi.

Attraverso l'applicazione di una *misura di stabilità dei Cluster* l'algoritmo suggerisce più *tagli* dell'albero generato permettendo l'estrazione di Cluster a densità diversa.

Dunque fissato m_{pts} (dato in input), HDBSCAN produce tutti possibili Clustering tagliando l'albero a corrispondenti valori di ϵ diversi.

Applicando il concetto di EOM applicato a HDBSCAN si ottiene l'algoritmo HDBSCAN-EOM che estrae un Clustering piatto formato da Cluster a densità variabile.

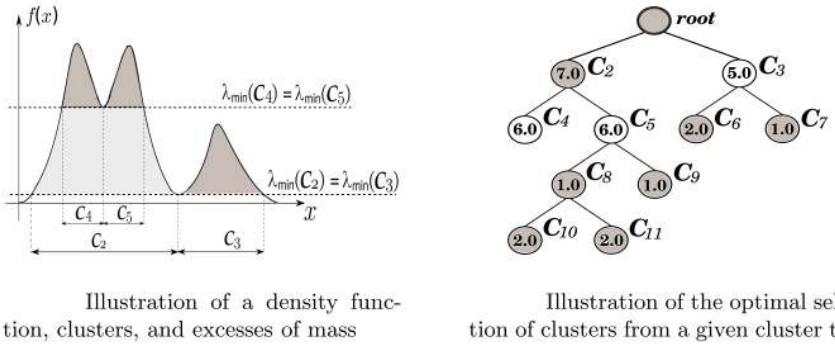


Illustration of a density function, clusters, and excesses of mass

Illustration of the optimal selection of clusters from a given cluster tree

Figura 2.17: Variazione della densità dei Cluster applicando il concetto diEOM [7]

Il concetto di EOM consiste nel prendere un Cluster significativo e, fissato m_{pts}, diminuire progressivamente il raggio ϵ così che il Cluster diverrà sempre più piccolo sino a scomparire oppure si scomporrà in ulteriori Cluster.

I Cluster che sopravviveranno più a lungo a questa graduale diminuzione di ϵ, e quindi all'aumentare della densità, saranno estratti al fine di formare il Clustering non gerarchico ottimale. [7]

2.6.3 Picchi di densità

Una delle caratteristiche più citate in letteratura, relativamente ai concetti di densità, è la presenza di picchi di densità intervallati da aree a bassa densità, ma non sempre le soluzioni proposte sono riuscite ad identificarle con successo.

É proprio su questo concetto che si basa il Processo di clustering proposto in [59] che diversamente dai metodi precedenti propone una soluzione di Clustering basato sull'identificazione di aree caratterizzate da picchi di densità proponendo al contempo una definizione differente di *densità* rispetto a quella proposta da [14] con il DBSCAN (vedi sezione 2.3.2).

Idea di base

L'approccio si basa sul fatto che esistono Punti centrali dei cluster dei Cluster che sono caratterizzati da:

- un'alta densità rispetto al suo Vicinato
- un'elevata distanza rispetto al altri Punti centrali dei cluster

Tale idea emerge in altri articoli in forma diversa. [7]

Definizione di densità

Fissata una distanza soglia $d_c > 0$ ed un punto i-esimo, la densità p_i equivale al numero di punti che sono distanti meno del valore d_c rispetto al punto i-esimo.

$$p_i = \sum_j \chi(d_{ij} - d_c) \tag{2.7}$$

$$\chi(x) = \begin{cases} 1 & \text{se } x < 0 \\ 0 & \text{se } altrimenti \end{cases}$$

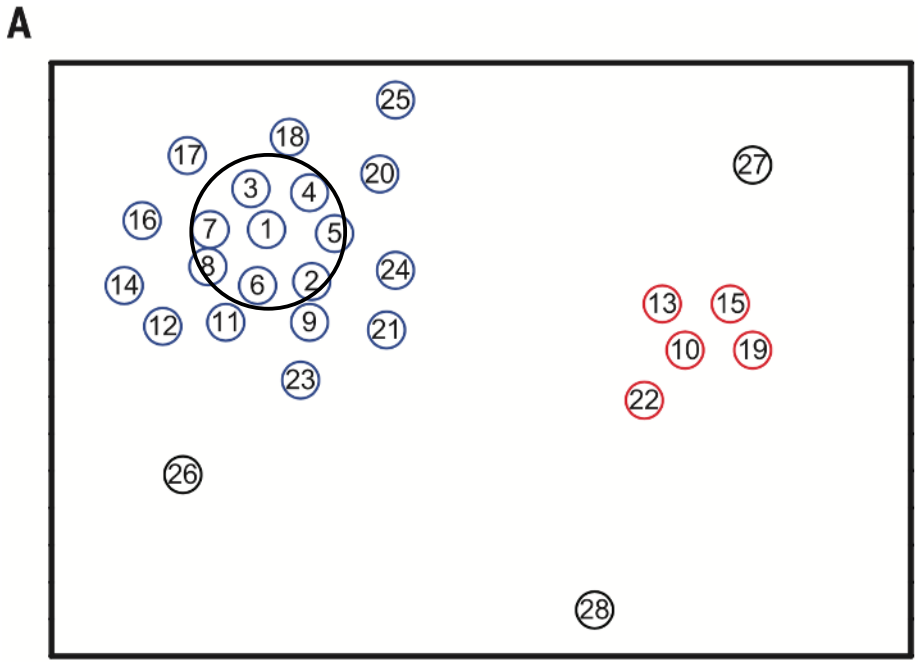


Figura 2.18: Punti distribuiti nello spazio numerati per densità. Ad esempio il punto (1) rispetto alla distanza soglia d_c ha una densità $p_1 = 7$; [59]

Ad esempio il punto (1) (in figura 2.18) rispetto alla distanza soglia d_c rappresentata dal cerchio ha una densità $p_1 = 7$, e rappresenta il punto con la densità locale più alta della collezione.

Punti Centrali ad alta densità

Il secondo parametro è δ_i distanza minima tra il punto i e qualsiasi altro punto caratterizzato da la più alta densità.

Tale valore sarà molto più elevato tra Punti centrali dei cluster diversi piuttosto che tra punti i del Vicinato. I Punti centrali dei cluster sono così scelti tra quelli con un δ_i elevato in modo anomalo.

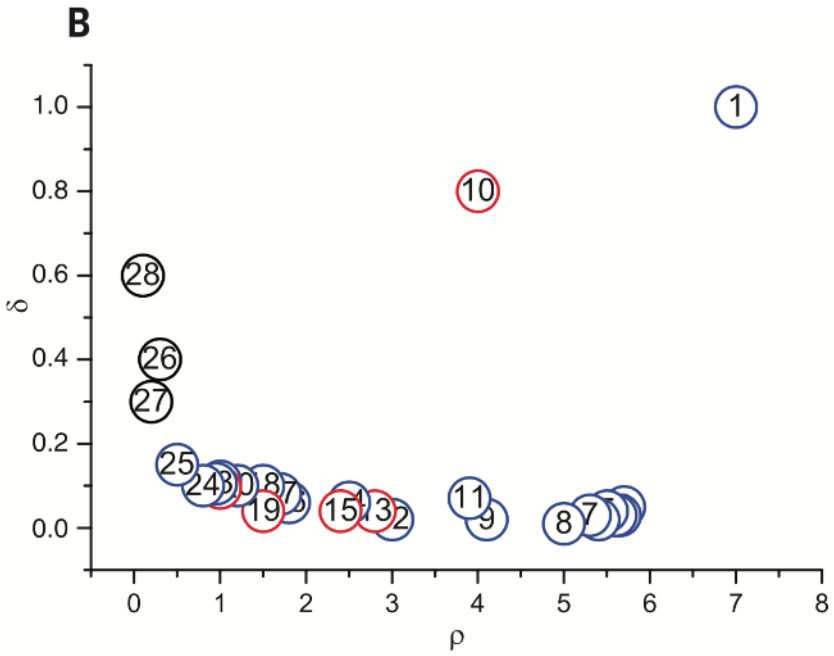


Figura 2.19: Grafico che mette a confronto i valori di densità p e δ che enfatizza i punti (1) e (10) come Punti centrali dei cluster; [59]

Costruzione dei Cluster

L'ultimo step è la formazione dei Cluster a partire dai Punti centrali dei cluster identificati con i parametri precedenti. Semplicemente a partire dai Punti centrali dei cluster ed una ulteriore distanza soglia d_d, verranno selezionati i punti finché saranno meno distanti della distanza appena data.

Commenti

Interessante risultano sia la definizione di densità che il metodo proposto per identificare i Punti centrali dei cluster attraverso l'identificazione di variazioni anomale di densità.

Seppur interessanti tale approccio non risolve il problema dell'utilizzo di più parametri di input per una corretta applicazione.

Parte II

Il Progetto

Capitolo 3

Acquisizione dei dati

3.1 Google Maps

Per il presente lavoro sono state utilizzate come sorgenti dati le Application Program Interface (API) offerte da Google attraverso le Google Place API[1]. Queste ultime, offrono una parte dei contenuti geografici che possono essere apprezzati nella loro completezza attraverso il servizio Google Maps[2].

Google Maps (in precedenza chiamato Google Local) è il servizio di mappatura del globo terrestre realizzato da Google Inc., attraverso il quale vengono rese disponibili, in via gratuita, diverse funzionalità ed informazioni relativamente alle mappe che vanno dal percorso più breve per raggiungere un luogo, alla mappatura topografica e fotografica, fino alla localizzazione dei luoghi più rilevanti (detti Place), proprio quest'ultima funzionalità è quella che più interessa a questo lavoro.

Una delle peculiarità di Google Maps sta nel fatto che può essere definito come un Geographic Information System (GIS) frutto di complessi metodi di raccolta dati provenienti da satelliti, informazioni amministrative ed un gran numero di contribuiti degli utenti (VGI) che attraverso le varie piattaforme diventano sia fruitori e contributori delle informazioni.

Le informazioni raccolte sono rese disponibili in misura e modalità differenti attraverso applicazioni e API differenti, ad esempio l'applicazione mobile per iOS e Android, oppure Google Earth per Desktop.

[1] `https://developers.google.com/places/`
[2] `https://maps.google.it`

3.1.1 Le informazioni geografiche

Google rende disponibili un gran numero di informazioni geografiche per mezzo di diverse API tutte appartenenti alla famiglia delle API di Google Map che si suddividono in:

- Web API;
- Web Service API;
- Mobile API.

Tipologia di informazioni

Vengono offerte una vasta varietà di informazioni geografiche: la posizione dei luoghi di interesse, attività commerciali, trasporti, strade e percorsi più brevi per raggiungere un determinato luogo, nonché tutti i dati geografici più classici quali confini, elevazione e immagini satellitari.

Region Code	Country/Region	Map Tiles	Geocoding	Traffic Layer	Driving Directions	Biking Directions	Walking Directions	Speed Limits
•••	•••							
IL	Israel	●	●	●	●	–	●	○
IT	Italy	●	●	●	●	–	●	●
JM	Jamaica	●	●	●	●	–	●	○
JP	Japan	●	●	●	●	–	●	–
•••	•••							
GB	United Kingdom	●	●	●	●	●	●	●
US	United States	●	●	●	●	●	●	●
UY	Uruguay	●	●	●	●	–	●	○
UZ	Uzbekistan	●	●	●	●	–	●	○
•••	•••							

Figura 3.1: Estratto delle coperture dei vari servizi geografici di Google Maps rispetto ad alcune aree geografiche: buona qualità (simbolo •), qualità approssimativa (simbolo ○), scarsa qualità (simbolo -)

Tali informazioni non sono tutte ugualmente disponibili per la totalità della superficie terreste[3], la disponibilità si divide in: funzionalità presente con dati di buona qualità, funzionalità presente, ma con con dati di qualità approssimativa, funzionalità non presente o dati di scarsa qualità (vedi figura 3.1).

[3] `https://developers.google.com/maps/coverage`

Come vengono raccolte le informazioni geografiche

Ai fini della presente ricerca è importante conoscere la provenienza dei dati che Google offre attraverso le proprie piattaforme, infatti esse sono ottenute tramite una combinazione di:

- raccolta automatica (i.e. satelliti o web crawler);
- partner commerciali;
- dati pubblici e governativi;
- fonti Open-source;
- contribui degli utenti.

Può capitare che l'uso dei alcuni dati geografici siano influenzati da contesti legislativi locali[4].

Google, per la denominazione dei paesi e dei territori si basano principalmente sullo standard ISO-3166[5].

Mentre i risultati delle attività commerciali nel contesto italiano potrebbero essere in parte forniti da SEAT Pagine Gialle SPA[6].

Partner Google

Google sottolinea quanto siano importanti le partnership in materia di informazioni geografiche, al fine di fornire informazioni sempre aggiornate e corrette ai loro utenti. A tal proposito invita non solo gli utenti finali, ma anche partner governativi, locali e internazionali a collaborare attraverso una specifica pagina [7]. Enfatizzando il fatto che le informazioni provenienti da *chi un certo luogo lo conosce* sono potenzialmente più accurate.

Per questo motivo Google permette l'inserimento, la modifica, la gestione e l'eliminazione di qualsiasi luogo di interesse a tutti i suoi utenti. Mentre non permette l'inserimento o la modifica di informazioni di tipo strettamente geografico-topografico quali il nome di una via, il percorso di una strada o l'elevazione di un certo luogo.

[4] https://www.google.com/intl/en_ALL/help/legalnotices_maps.html

[5] https://www.iso.org/iso-3166-country-codes.html

[6] https://www.paginegialle.it

[7] http://maps.google.com/help/maps/mapcontent/basemap/index.html

3.1.2 Google Maps API Place

L'API Places è il servizio di Google che fornisce informazioni sui Place attraverso richieste HTTP. Può restituire informazioni in formato JSON o XML in base alle necessità dell'utente e come accade spesso con altre API pubbliche è sempre necessario utilizzare una chiave di accesso (API KEY) che in questo caso è fornita gratuitamente da Google.

Place Nelle piattaforme di Google Map i luoghi sono detti genericamente Place e sono definiti come stabilimenti, località geografiche e punti di interesse. Per identificare in modo univoco ognuno di essi viene utilizzato un ID univoco alfanumerico di lunghezza variabile detto *place-id* (vedi 3.2).

```
{
    "html_attributions" : [],
    "results" : [
    {
        "geometry" : {
            "location" : {
                "lat" : -33.870775,
                "lng" : 151.199025
            }
        },
        ...
        "place_id" : "ChIJrTLr-GyuEmsRBfy61i59si0",
        ...
    }
    ],
    "status" : "OK"
}
```

Figura 3.2: Estratto di un JSON relativo ad un Place di Google contentente informazioni sulla posizione geografica e l'identificatore univoco (place-id)

Servizi Web Le varie informazioni dei Place di Google sono rese disponibili attraverso diversi servizi, ognuno accessibile da una specifica Uniform Resource Locator (URL) su protocollo HTTP. Per i nostri scopi utilizzeremo solo i seguenti servizi:

- Place Search: che restituisce un elenco di luoghi in base alla posizione dell'utente o alla stringa di ricerca;
  https://maps.googleapis.com/maps/api/place/nearbysearch/

- Place Details: che restituisce informazioni più dettagliate su un luogo specifico, comprese le recensioni degli utenti;
  https://maps.googleapis.com/maps/api/place/details/

Oltre ad essi sono disponibili altri servizi che non saranno utilizzati in questo lavoro, quali:

- Place Photos;
- Place Autocomplete;
- Query Autocomplete.

Limitazioni Google applica una limitazione generale di 1.000 richieste gratuite ogni 24 ore, calcolate come somma delle richieste lato Client e lato Server su tutti i servizi precedentemente elencati. In oltre, per ogni servizio esistono ulteriori limitazioni specifiche rispetto al tipo di informazioni restituite, queste ultime saranno trattate nelle sezioni successive.

Formato URL Gli URL dei servizi messi a disposizione da Google possiedono una struttura standardizzata composta da due parti (vedi 3.3):

- la Web Service URL di base;
- alcuni parametri aggiuntivi della URL, che si compongono di:
  - diversi parametri specifici del servizio;
  - un API KEY che invece rimane fissa per tutti i servizi;

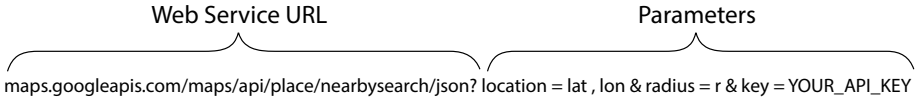


Figura 3.3: Esempio di URL e parametri di base del servizio *Place Search*

Place Search

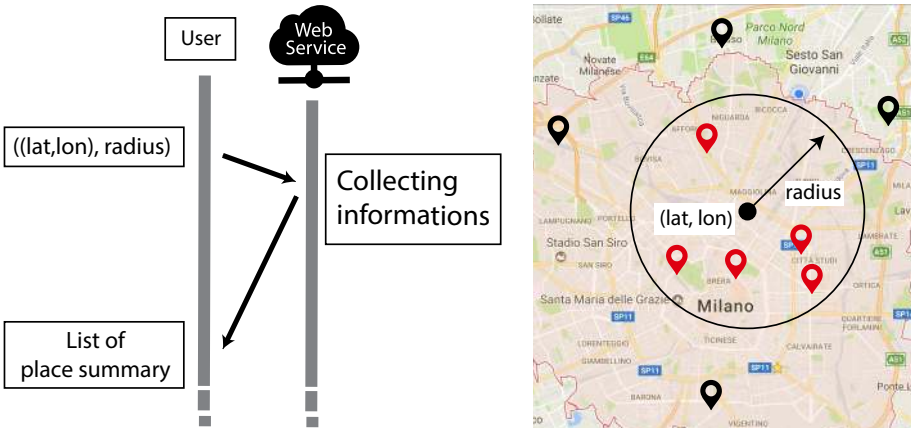
Il servizio Place Search restituisce una lista di Place rispetto ad un luogo geografico indicato dall'utente, il quale può scegliere di utilizzare una posizione espressa in latitudine e longitudine oppure una query testuale che fa riferimento ad una specifica area della mappa (i.e. Duomo di Milano), per i nostri scopi ci concentreremo sulle richieste effettuate tramite l'uso di longitudine e latitudine.

Richiesta La richiesta dei luoghi in una data area è effettuata comunicando le coordinate GPS ed un raggio di interesse entro il quale Google raccoglierà le informazione e formulerà la risposta in una lista di Place formattati in un file in formato JSON (vedi 3.4).

I parametri di base della richiesta sono:

- coordinate geografica espresse in latitudine e longitudine: `location=<lat>,<lon>`;
- raggio di interesse espresso in metri: `radius=r`;

(a) URL del servizio API Place Search



(b) Visualizzazione di una generica richiesta di un'area tramite API Place Search

Figura 3.4: Richiesta della lista dei Place tramite API Place Search di Google di centro lat, lon e raggio r

Risposta Se la richiesta di centro lat, lon e raggio r è formulata correttamente, il Web Service si occuperà di collezionare i Place entro l'area circolare richiesta e di restituirli in forma di lista di Place, attraverso un file JavaScript Object Notation (JSON) (figura 3.4). Ma tali Place elencati conterranno solo una parte delle informazioni complessivamente disponibili sui Place rappresentando un riassunto che chiamiamo `Place_Summary` (vedi 3.5).

I `Place_Summary` contengono solo alcune delle informazioni del Place, tra i quali: la posizione geografica, il nome, ed il `place_id` (vedi 3.5).

Limitazioni Oltre alle $1.000 richieste/24h$ massime, Google impone altre limitazioni, come nel caso del servizio Place Search che si limita a fornire un massimo di 60 Place divisi per gruppi di 20 per pagina.

Dunque supponendo che una data area di centro lat, lon e raggio $radius$ contenga 35 Place, sarà necessario effettuare una prima richiesta

```
{
   "html_attributions" : [ ],
   "results" : [
      <Place_Summary>,
      <Place_Summary>,
      <Place_Summary>,
      {
        "geometry" : {
        "location" : {
            "lat" : -33.870775,
            "lng" : 151.199025
            }
        },
        "icon" : "<URL>.png",
        "id" : "21a0b251c95b4aa...",
        "name" : "Rhythmboat Cruises",
        "opening_hours" : { "open_now" : true },
        "photos" : [ <Photo>, <Photo>, ... ],
        "place_id" : "ChIJyWEHuETkapTCrk...",
        "scope" : "GOOGLE",
        "alt_ids" : [...],
        "reference" : "CoQBdQu9h9wk_NCBMk...",
        "types" : ["restaurant", "food" ],
        "vicinity" : "Pyrmont Ba..."
        },
      ...
   ],
   "status" : "OK"
}
```

Figura 3.5: Struttura della risposta al servizio Place Search di Google contente del versioni riassuntive dei Place

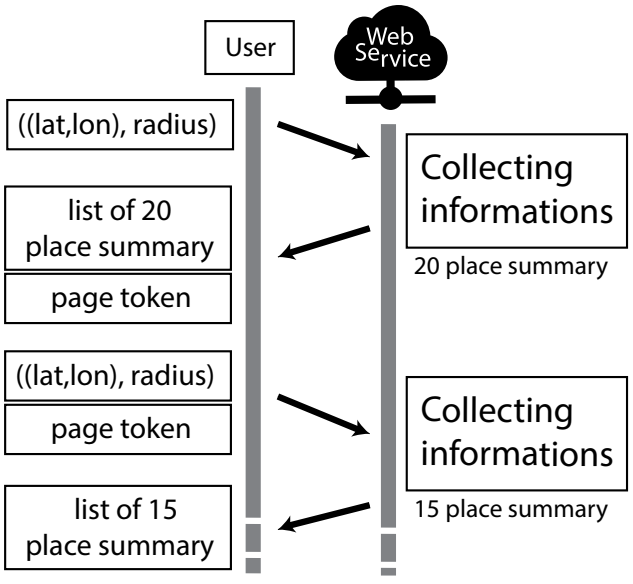


Figura 3.6: Richiesta multipla su più pagine al servizio Place Search tramite page_token

la quale conterrà un cosiddetto `page_token` con il quale effettuare una seconda richiesta ed ottenere i restanti 15 Place (vedi figura 3.6).

Place Detail

Place Detail è il servizio che permette di ottenere i dettagli aggiuntivi dei Place tramite i rispettivi `place_id`, ottenuti precedentemente attraverso il servizio Place Search.

Formato richiesta La richiesta delle informazioni dettagliate di un Place (i.e. Duomo di Milano) è effettuata, come le precedenti, utilizzando l'apposito URL su protocollo HyperText Transfer Protocol (HTTP) con l'unica differenza che invece di utilizzare latitudine e longitudine come parametri, si utilizza il `place_id` (vedi figura 3.7).

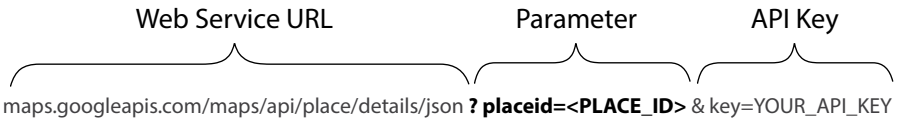


Figura 3.7: Formato richiesta al servizio Place Detail di Google

Formato risposta Se la richeista è corretta il servio Place Detail fornirà in risposta un file JSON contenente i dettagli aggiuntivi relativi al Place come le valutazioni degli utenti e URL delle foto.

Limitazioni É importante notare che, seppur sia possibile recuperare ulteriori informazioni relativamente al place, esse comunque non raggiungono il livello di completezza delle informazioni fornite tramite interfaccia Web (vedi figura 3.9).

Ad esempio nella versione Web di Google Maps è possibile consultare la stima dell'affluenza delle visite che nelle informazioni dettagliate tramite API è completamente assente (figura 3.9a). Mentre per quanto riguarda le recensioni degli utenti l'interfaccia tramite API ne limita l'accesso alle sole ultime 5, mentre ovviamente tramite interfaccia Web è possibile consultarle tutte attraverso gli appositi controlli (figura 3.9b).

Per quanto riguarda le fotografie relative ai Place esse sono tutte disponibili da interfaccia web, mentre solo le ultime 10 consultabili tramite API. In oltre nella risposta dettagliata in formato JSON si dispone solo del *id* relativo alla foto con il quale è possibile effettuarne il download tramite il servizio Place Photo, che in questo non contesto non tratteremo nel dettaglio. Ma basti pensare che ha un funzionamento del tutto simile a Place Datail.

```
{
   "html_attributions" : [],
   "result" : {
      "address_components" : [
         {
            "long_name" : "Piazza del Duomo",
            "short_name" : "Piazza del Duomo",
            "types" : [ "route" ]
         }, ...
      ],
      "geometry" : {
         "location" : {
            "lat" : 45.4640976,
            "lng" : 9.191926499999999
         },...
         }
      },
      ...,
      "name" : "Duomo di Milano",
      "opening_hours" : {
         "open_now" : true,
         "periods" : [...],
         "weekday_text" : [...]
      },
      "photos" : [<Photo>,<Photo>,...],
      "place_id" : "ChIJoTZGw67GhkcREy4aECdOf6s",
      "rating" : 4.7,
      "reviews" : [<Review>,<Review>,...],
      ...
      "url" : "https://maps.google.com/?cid=12357681832208772627",
   },
   "status" : "OK"
}
```


Figura 3.8: Esempio di risposta del servizio Place Detail di Google

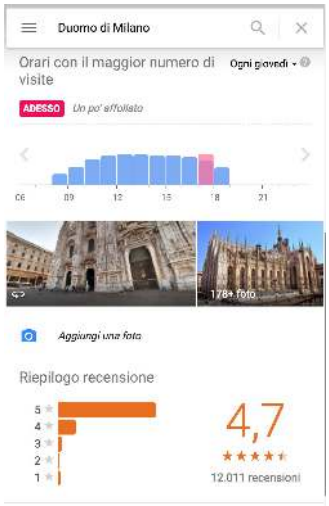
3.1.3 Proprietà dei Place in dettaglio

Il Place ID

Il `place_id` è un identificatore testuale alfanumerico di lunghezza variabile che Google utilizzate per referenziare univocamente un Place.

Ogni `place_id` si riferisce sempre ad un unico Place, ma ogni Place può avere più di un `place_id`, ad esempio nel caso in cui un Place si trasferisce in un nuova sede oppure oppure viene creato ex-novo da un utente. Invece nel caso in cui un Place viene eliminato può accadere che il `place_id` non corrisponda ad alcun Place.

Dunque data la natura del `place_id`, esso è il campo utilizzato per identificare i Place duplicati durante la cattura, al contrario delle coordinate GPS che non sono univoche, ma possono corrispondere a più luoghi di interesse come nel caso di un palazzo a più piani in cui una stessa posizione geografica corrisponde a più attività commerciali.



(a) Affluenza del Duomo di Milano

(b) Valutazioni del Duomo di Milano

Figura 3.9: Informazioni dettagliate relative ad un Place consultabili tramite interfaccia web

Le Recensioni

Google permette di recensire qualsiasi Place presente nella mappa. Esse sono disponibili nell'oggetto JSON fornito dal servizio Place Detail relativamente ad un Place sotto la proprietà `reviews`.

Tra le informazioni più importanti di una `reviews` troviamo opzionalmente il nome dell'autore, la lingua, la valutazione compresa tra 0 e 5, una descrizione testuale e l'ora in cui è stata creata la recensione in formato timestamp (vedi 3.10).

```
{
    "author_name" : "Nome autore",
    "author_url" : "https://www.google.com/maps/cont...",
    "language" : "it",
    "profile_photo_url" : "...",
    "rating" : 5,
    "relative_time_description" : "Descrizione",
    "text" : "Luogo simbolo della citta...",
    "time" : 1525732721
},
```

Figura 3.10: Estratto di una recensione di un Place di Google ottenuta tramite servizio Place Detail

I Tipi

Un'informazione interessante che Google fornisce relativamente ai Place è rappresentata dai i tipi attraverso la proprietà `type`, essi forniscono un elenco di termini che caratterizzano un luogo allo stesso modo dei tag o le categorie nel conteso Web.

Le principali peculiarità della proprietà `type` sono:

- ogni Place ne contiene almeno uno;
- non sono liberamente realizzati dall'utente, ma sono scelti da una lista fornita da Google[8]
  - tale elenco di possibili `type` è ulteriormente suddiviso in due tabelle: Tabella 3.1 e Tabella 3.2;

La tabella 3.1 rappresenta la lista di `type` utilizzabili come chiave di ricerca nel servizio Place Search oppure suggeribili dall'utente durante la fase di inserimento di un Place.

accounting	airport	amusement_park
aquarium	art_gallery	atm
bakery	bank	bar
beauty_salon	bicycle_store	book_store
bowling_alley	bus_station	cafe
campground	car_dealer	car_rental
car_repair	car_wash	casino
cemetery	church	city_hall
clothing_store	convenience_store	courthouse
dentist	department_store	doctor
electrician	electronics_store	embassy
fire_station	florist	funeral_home
furniture_store	gas_station	gym
hair_care	hardware_store	hindu_temple
home_goods_store	hospital	insurance_agency
jewelry_store	laundry	lawyer
library	liquor_store	local_government_office
locksmith	lodging	meal_delivery
meal_takeaway	mosque	movie_rental
movie_theater	moving_company	museum
night_club	painter	park
parking	pet_store	pharmacy
physiotherapist	plumber	police
post_office	real_estate_agency	restaurant
roofing_contractor	rv_park	school
shoe_store	shopping_mall	spa
stadium	storage	store
subway_station	supermarket	synagogue
taxi_stand	train_station	transit_station
travel_agency	veterinary_care	zoo

Tabella 3.1: Elenco dei Type che possono essere utilizzati nei Place

[8] `https://developers.google.com/places/supported_types`

La tabella 3.2 rappresenta una lista di possibili ulteriori `type` che Google si riserva di utilizzare nei vari Place attraverso un processo di inserimento interno. L'unione delle tabelle 3.1 e 3.2 rappresenta l'insieme dei possibili `type` che si possono incontrare nei vari Place di Google.

administrative_area_level_1	administrative_area_level_2
administrative_area_level_3	administrative_area_level_4
administrative_area_level_5	colloquial_area
country	establishment
finance	floor
food	general_contractor
geocode	health
intersection	locality
natural_feature	neighborhood
place_of_worship	political
point_of_interest	post_box
postal_code	postal_code_prefix
postal_code_suffix	postal_town
premise	room
route	street_address
street_number	sublocality
sublocality_level_4	sublocality_level_5
sublocality_level_3	sublocality_level_2
sublocality_level_1	subpremise

Tabella 3.2: Elenco dei Type che possono essere utilizzati nei Place, ma solo da parte del personale d Google

Infine la 3.3 attraverso cui Google propone una lista di `type` dichiarati deprecati, ma che comunque è possibile incontrare in alcuni Place.

Tabella 3 dei Type deprecati)	
establishment	finance
food	general_contractor
grocery_or_supermarket	health
place_of_worship	

Tabella 3.3: Tabella 3 di Google che elenca i Type deprecati, ma che potrebbero ancora essere trovati nei Place

3.2 Sfide e criticità

La raccolta dei dati è realizzata partendo dalle modalità imposte dalle API di Google Maps, utilizzando una strategia di raccolta dei dati intelligente, in grado di operare in modo autonomo anche in presenza di criticità. Prima di vedere nel dettaglio l'algoritmo di acquisizione affronteremo in questa sezione alcune sfide preliminari che sono state affrontate in una fase precedente all'implementazione dell'algoritmo vero e proprio per la raccolta dei dati.

Ci si è dunque posti l'obiettivo di realizzare un algoritmo di raccolta dati massivo dei Place di Google Maps che necessitasse di un input ed un

supporto utente minimo durante tutta la sua esecuzione, che fosse in oltre in grado di adattarsi a qualsiasi forma di distribuzione spaziale della città ed in fine che fosse in grado di ottimizzare il numero di richieste rispetto ai limiti e le modalità di accesso verso le API imposte da Google.

Tali obiettivi sono stati risolti affrontando i seguenti aspetti;

- una suddivisione della superficie terreste che fosse compatibile con aree circolari ed allo stesso tempo in grado di coprire l'intera area;
- una strategia di esplorazione che fosse in grado di assecondare la morfologia urbana;
- tecniche di variazione della granularità dell'esplorazione che ottimizzasse il numero di richieste.

3.2.1 Suddivisione della superficie

Dato l'obiettivo di raccogliere la totalità dei Place dell'area di Milano attraverso richieste limitate ad un area circolare contenete un massimo di 60 Place (vedi figura 3.11), è stato necessario affrontare il problema della suddivisione della superficie terreste in modo da distribuire adeguatamente le richieste alle Place di Google.

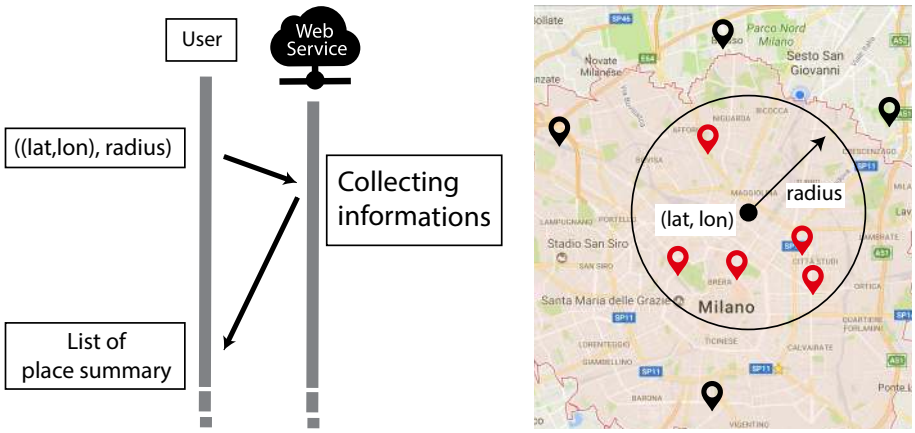


Figura 3.11: Richiesta tramite API di Google Maps, limitate ad un area circolare di centro lat, lon e raggio r

É chiaro che affiancando semplicemente le circonferenze e non è possibile ottenere una copertura totale della superficie (vedi figura 3.12), è infatti necessario sovrapporre le circonferenze al fine di coprire tutte le aree. Dato il contesto, l'obiettivo è stato individuare una suddivisione della superficie che minimizzasse la sovrapposizione.

Griglia esagonale La soluzione è stata quella di approssimare la circonferenza ad un esagono e realizzare una suddivisione della

superficie attraverso una griglia esagonale (detta in alcuni casi tassellatura esagonale). [27]

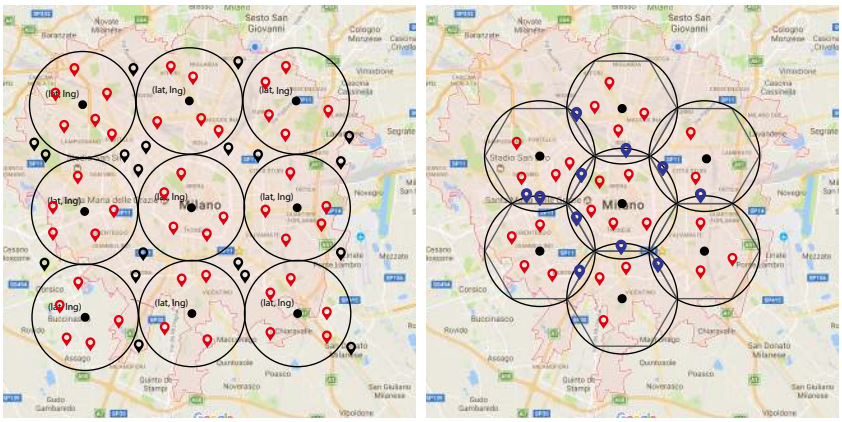
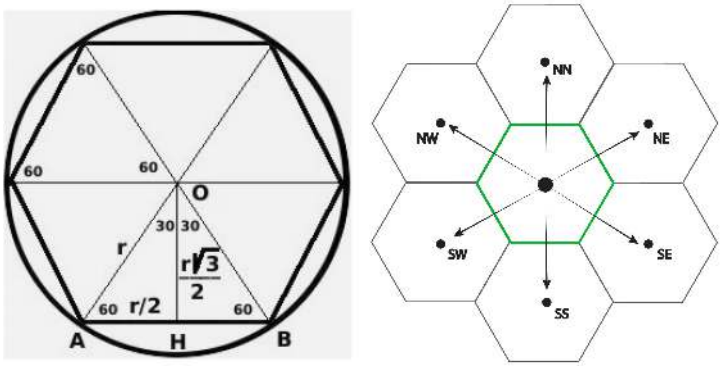


Figura 3.12: Confronto tra griglie a sinistra quella ottenuta con l'affiancamento di circonferenze, mentre a destra quella quella attraverso una griglia esagonale

Tale soluzione permette di coprire la totalità di un'area attraverso circonferenze di pari dimensione, minimizzando le aree esplorate più volte.

Costruzione della griglia esagonale Da ora in poi ci riferiremo alle aree circolari anche con il termine area esagonale, costruita all'interno di una circonferenza di raggio r e di centro $c_0 = (lat_0, lon_0)$, dove i lati dei triangoli equilateri che formano l'esagono valgono r mentre l'altezza dei triangoli vale $h = \frac{r\sqrt{3}}{2}$ (vedi figura 3.13a).



(a) (b)

Per costruire la griglia esagonale a partire da un centro è stato realizzato un sistema per calcolare le 6 aree esagonali intorno attraverso 6 direzioni (vedi figure 3.13b).

Considerando una circonferenza c di raggio$= r$, centro $c_0 = (x, y)$, ed altezza $h = \frac{r\sqrt{3}}{2}$ si possono derivare i relativi centri delle aree esagonali

attraverso un insieme che chiamiamo anello (vedi figure 3.13b)

$ring_1 = \{c_{NN}, c_{NE}, c_{SE}, c_{SS}, c_{SW}, c_{SW}\},\ c_x = (lat_x, lon_x)$

dove i relativi centri valgono:

$$\begin{aligned} c_{NN} &= \{x + 2h,\ 0\} \\ c_{NE} &= \{x + r + (\tfrac{r}{2}),\ y + h + (\tfrac{r}{2})\} \\ c_{SE} &= \{x + r + (\tfrac{r}{2}),\ y - h + (\tfrac{r}{2})\} \\ c_{SS} &= \{x - 2h,\ 0\} \\ c_{SE} &= \{x + r + (\tfrac{r}{2}),\ y - h + (\tfrac{r}{2})\} \\ c_{SW} &= \{x - r + (\tfrac{r}{2}),\ y - h + (\tfrac{r}{2})\} \\ c_{NW} &= \{x - r + (\tfrac{r}{2}),\ y + (\tfrac{r}{2})\} \end{aligned}$$

3.2.2 Strategia di esplorazione

Il passo successivo alla costruzione della griglia esagonale è stato scegliere una strategia di costruzione della stessa e quindi di esplorazione dello spazio.

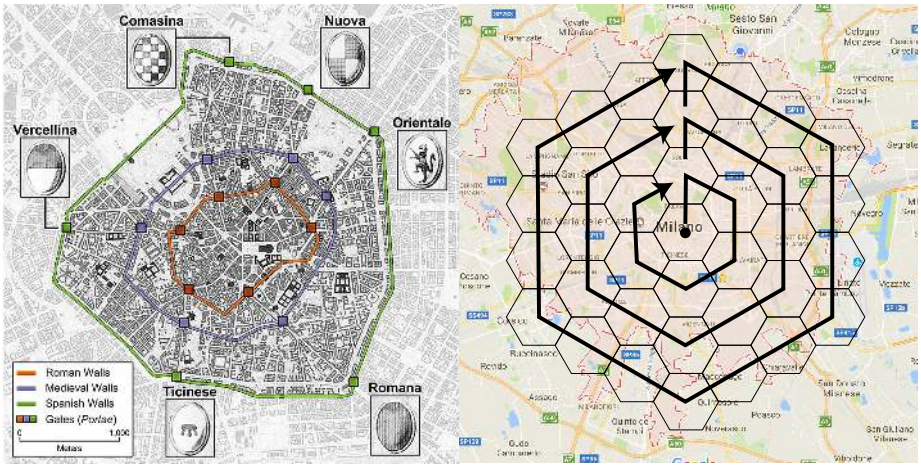


Figura 3.14: Confronto tra la struttura urbana di Milano e la griglia esagonale con movimento a spirale

Considerando la struttura delle vecchie mura della città di Milano, si è scelto di realizzare una strategia esplorativa per mezzo di un movimento a spirale che, partendo da un punto centrale, mano-a-mano costruisce le aree esagonali intorno aggiungendo anelli sempre più esterni (vedi figura 3.15).

Partendo da un'area esagonale centrale di centro $c_0 = (x, y)$ e le direzioni relative di $\{c_{NN}, c_{NE}, c_{SE}, c_{SS}, c_{SW}, c_{SW}\},\ c_x = (lat_x, lon_x)$ possiamo derivare il movimento a spirale con gli step nell'esempio seguente:

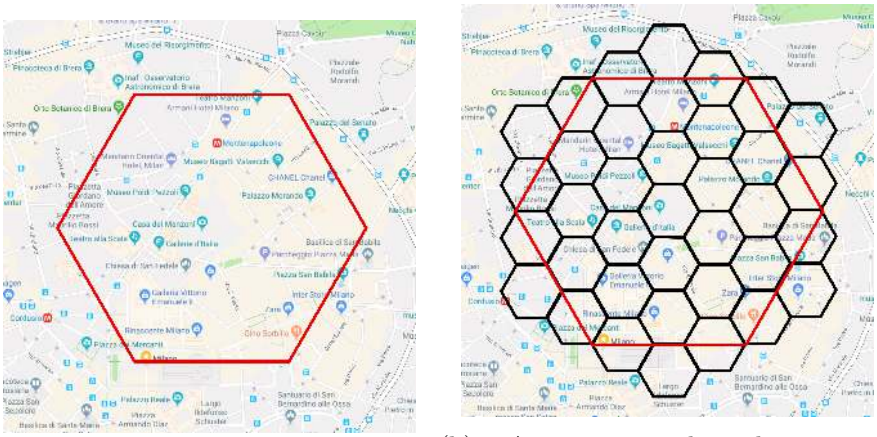
- Step 0: $c_0 = (x, y)$;

Figura 3.15: Costruzione della griglia esagonale attraverso un movimento a spirale

- Step 1: $c_0 = (x, y)$, NN, SW, SS, NW, NE;
- Step 2: $c_0 = (x, y)$, NN, NN, SW, SW, SS, SS, NW, NW, NE;
- Step 3: $c_0 = (x, y)$, NN, NN, NN, SW, SW, SW, SS, SS, SS, NW, NW, NW, NE, NE;
- Step a: $c_0 = (x, y)$, $(a)*$ NN, $(a)*$ SW, $(a)*$ SS, $(a)*$ NW, $(a-1)*$ NE;

3.2.3 Tecniche di variazione della granularità

Data la limitazione di 60 Place per ogni richiesta verso l'API di Google, è stato necessario prevedere il caso in tale numero non fosse sufficiente ad acquisire tutti i Place di un'area troppo densamente servita da servizi, dovendo necessariamente attuare una strategia di analisi differente.



(a) Area esagonale iniziale

(b) Area esagonale ulteriormente suddivisa

Figura 3.16: Variazione della granularità della griglia esagonale per una specifica area esagonale della città

A tal proposito è stato realizzato un sistema che, in caso di necessità, realizza una griglia esagonale secondaria formata da aree più piccole circoscritte nell'area interessata (vedi figura 3.16b) in modo da analizzare in modo più dettagliato l'area.

Al termine dell'analisi dell'area attraverso la griglia secondaria, l'algoritmo, tornerà ad analizzare la mappa utilizzando nuovamente la griglia esagonale iniziale. Ripetendo l'operazione di analisi approfondita ogni volta che sarà necessario.

3.2.4 Le coordinate geografiche

Fino a questo punto, tutte le lunghezze e gli spostamenti sono stati trattati come fossero coordinate di un piano euclideo, ma quando si utilizzano latitudine e longitudine è necessario applicare alcune trasformazione per adattarsi al sistema di coordinate geospaziali.

Nelle coordinate geospaziali accade che uno spostamento di 1° verso nord oppure verso sud valga sempre la stessa quantità di metri in qualsiasi luogo sulla terra, mentre uno spostamento di 1° verso est oppure verso ovest equivale ad un numero di metri differenti in base alla latitudine in cui si applica lo spostamento (vedi figura 3.17). Questo dipende dal raggio dell'arco su cui si applica lo spostamento, ad esempio data la latitudine lat_1 il raggio della circonferenza su cui calcolare lo spostamento equivale a $R_{lat_1} = R * \cos(\pi * \frac{lat_1}{180})$.

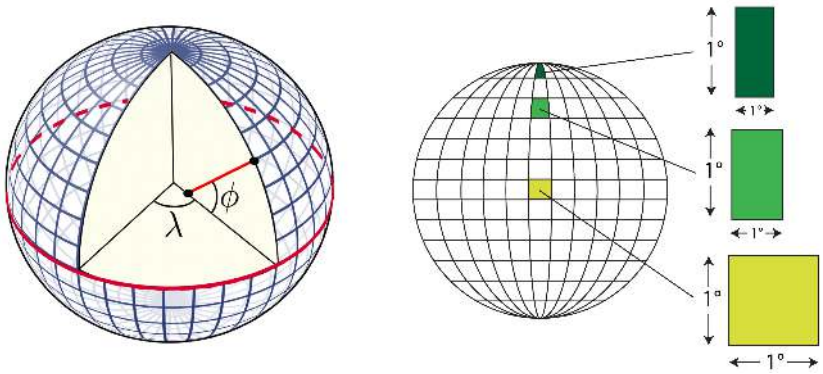


Figura 3.17: Sistema di coordinate geo-spaziale e variazione dell'equivalenza gradi/metri

Ipotizzando di voler ottenere un'area esagonale di centro lat_2, lon_2 posizionata a Nord-Est rispetto ad un'area di centro lat_1, lon_1, entrambi di raggio r (vedi figura 3.18), si ha che:

- la latitudine vale $lat_2 = lat_1 + d_{lat}$
- la longitudine vale $lon_2 = lon_1 + d_{lon}$

mentre logli spostamenti laterali d_{lat} e d_{lon} valgono

$$d_{lat} = \frac{\text{spostamento in metri}}{R}$$

$$d_{lon} = \frac{\text{spostamento in metri}}{R_{lat_1}} = \frac{\text{spostamento in metri}}{R*\cos(\pi*\frac{lat_1}{180})}$$

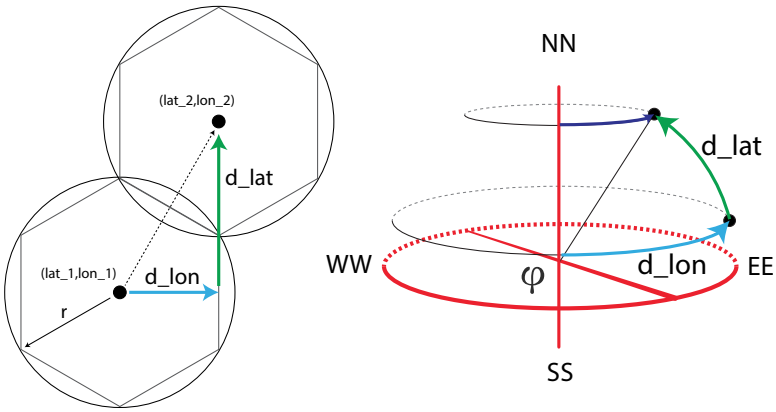


Figura 3.18: Spostamento laterale in coordinate geografiche

3.3 L'algoritmo in dettaglio

La parte di software dedicata alla raccolta dati è ottenuta per mezzo di quattro componenti, ognuna responsabile di un preciso compito (vedi figura 3.19).

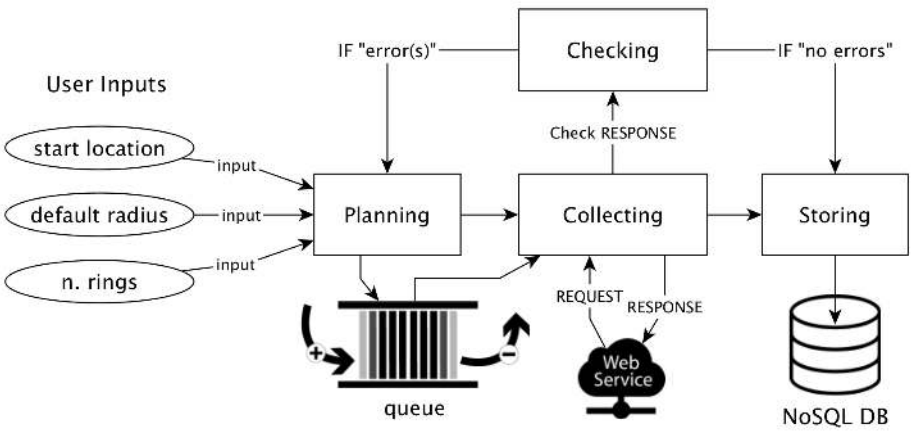


Figura 3.19: Componenti del software dedicata alla Raccolta Dati

Le componenti del processo di raccolta dati sono:

- Planning: si occupa della generazione della griglia esagonale e la relativa pianificazione delle richieste verso le API di Google;

- Collecting: si occupa dell'esecuzione delle richieste verso le API di Google;
- Checking: si occupa della verifica delle risposte ottenute delle richieste effettuate verso le API di Google;
- Storing: si occupa della memorizzazione in un database NO-SQL i Place raccolti.

Input del processo di raccolta La raccolta dati avviene in modo totalmente automatizzato a partire da un input composto da:

- `location_center: lat, lon`, coordinate geografiche dell'area esagonale di partenza;
- `default-radius: meters`, raggio dell'area di default della griglia principale;
- `n-ring: integer`, il numero di anelli concentrici corrispondenti agli strati della spirale esagonale.

Il raggio della area finale composta dalla griglia principale sarà pari a: (numero di anelli $*$ 2) $*$ raggio della cella (vedi 3.20).

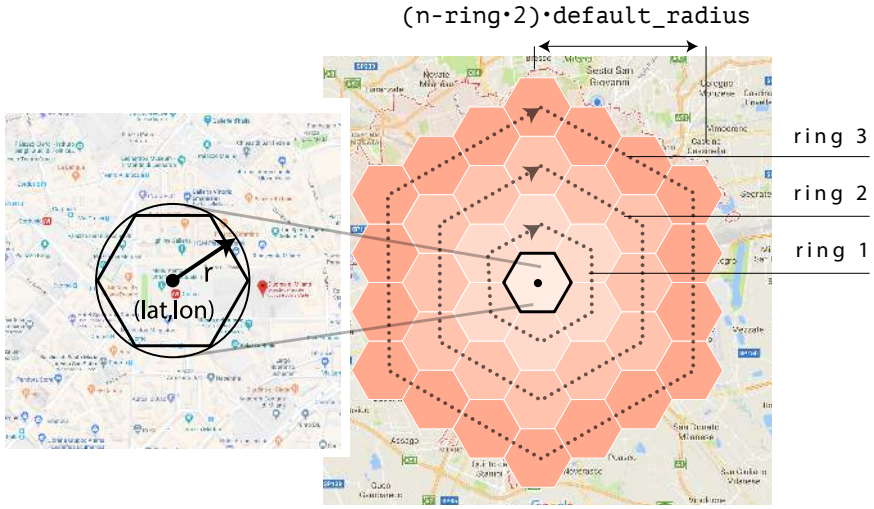


Figura 3.20: Costruzione della griglia esagonale e pianificazione delle richieste a partire dagli input dell'algoritmo di Raccolta Dati

Gli input appena descritti vengono utilizzati dalla componente di pianificazione che genera la griglia esagonale composta di `n` anelli concentrici intorno all'area centrata in `location_start` e di raggio `radius`. Dove ogni singola area esagonale di raggio `radius` della griglia corrisponderà all'area che verrà analizzata tramite una richiesta alle API di Google (vedi figura 3.20).

Panoramica dell'algoritmo di raccolta dati L'algoritmo generale di raccolta dei dati ha inizio con la pianificazione delle richieste tramite il metodo: `Planning.Area( )` che produce una lista di posizioni da analizzare basati su una griglia esagonale organizzate in una coda FIFO.

Quest'ultima permette ad un secondo componente di effettuare sequenzialmente l'analisi della superficie attraverso le apposite API di Google, le quali risposte verranno controllate al fine di verificare eventuali errori o la presenza di un numero eccessivo di Place (vedi l'algoritmo 3).

```
Data: start : (lat, lon), radius, n_ring
Result: set of (places) in Area of Interest
QUEUE := Planning.Area(start : (lat, lon), radius, n_ring) ;
while (location,radius) : = QUEUE.pop() do
    response := Collecting((location,radius)) ;
    check := Checking ( response ) ;
    if check := OK then
        Storing ( response ) ;
    else
        if check == TOO-PLACE then
            QUEUE.pushAll( Planning.Sub-Area( (location,
             radius) ) ) ;
        else
            if check == ERROR then
                QUEUE.pushOne ( Planning.OnePlanning(
                 (location, radius) )) ;
            end
        end
    end
end
```

Algoritmo 3: Algoritmo di Raccolta dati Generale

Nelle sezioni successive saranno descritti in dettaglio tutte e quattro le componenti di Planning, Collecting, Checking e Storing relative al processo di raccolta dati.

3.3.1 Pianificazione

La componente di pianificazione delle richieste (Planning) realizza la griglia esagonale calcolando le coordinate di ogni area esagonale a partire dal una posizione geografica, un raggio di delle celle esagonali ed il numero di anelli concentrici di cui sarà composta la griglia

Attraverso il movimento a spirale descritto nella sezione 3.2.2 vengono calcolate le posizioni di ogni esagono ed aggiunta ad una coda FIFO. Dove il primo punto in coda sarà quello più centrale, mentre gli ultimi saranno

quelli appartenenti agli anelli più esterni (vedi figura 3.21).

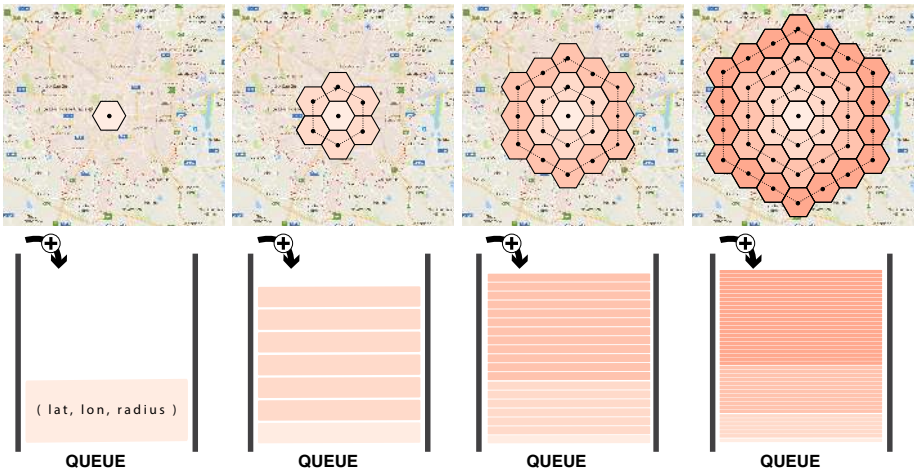


Figura 3.21: Pianificazione delle aree da analizzare attraverso le API di Google, dove ogni posizione è inserita nella coda

Data: $center : (lat, lon)$, $radius$, n_{ring}
Result: queue of $(lat, lon, radius)$ locations
for $i_{ring} \leftarrow 0$ **to** n_{ring} **do**
  hexCenters[:.] := Calc_centers_ring($center$, $radius$, i_{ring}));
  forall $location \leftarrow hexCenters$ **do**
    queue.push(location, i_{ring});
  end
end
return queue

Algoritmo 4: Algoritmo Planning() in dettaglio: costruisce la griglia di esagoni e ne restituisce i relativi centri organizzati in una coda.

Griglie esagonali di dimensione variabile Una particolarità della componente Planning risiede nella capacità di pianificare griglie di ampiezza e granularità variabili semplicemente variando i parametri input, dunque, come vedremo successivamente, è possibile richiamare la componente Planning in qualsiasi istante del processo di raccolta allo scopo di pianificare un'area qualsiasi della mappa secondo una griglia di granularità e dimensione a piacere i.e. :

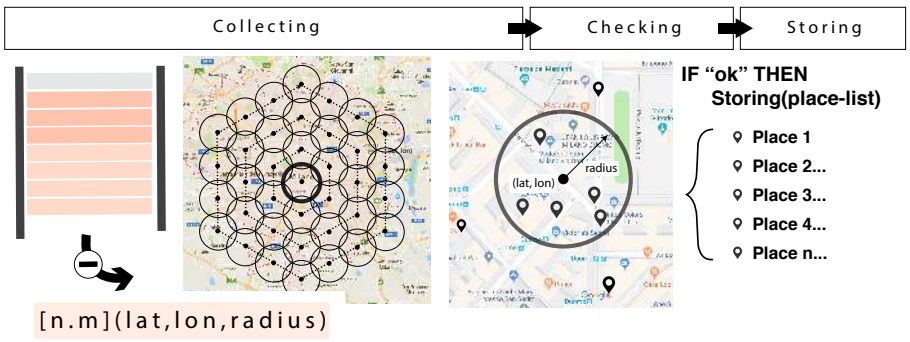
$$Planning.Area(start : (lat, lon), radius, n_{ring}).$$

3.3.2 L'esecuzione delle richieste

La componente `Collecting()` si occupa dell'analisi delle aree collezionando i Place presenti, essa estrae sequenzialmente la prima

posizione geografica presente nella coda e la utilizza per effettuare una richiesta verso le API di Google per mezzo del servizio Place Search (vedi figura 3.22a)

```
location_radius := queue.pop()
output := Collecting(location_radius)
```



(a) Planning esagonale

(b) Esempio URL e parametri di base del servizio *Place Search*

Al termine della richiesta, l'output sarà passato al componente Checking che si occuperà di controllarlo e quindi scegliere la procedura successiva in base al contenuto dell'output analizzato.

3.3.3 Verifica delle richieste

La componente `Checking()` si occupa di verificare che la risposta ottenuta dall'GlsAPI sia valida e non sia composta da un numero di Place eccessivo, quindi può restituire due tipi di errore:

- errore generico: nel caso venga riscontrato un errore di comunicazione ad esempio una mancata ricezione, un errore di formattazione oppure un API KEY non valida;
- numero eccessivo di Place: nel caso venga riscontrato un numero di Place vicino al limite dei 60 imposto da Google.

Errore generico Nel caso venga riscontrato un errore generico la richiesta nella posizione geografica interessata sarà pianificata nuovamente dalla componente di `Planning()`, quindi, aggiunta alla nuovamente alla coda (vedi figura 3.23).

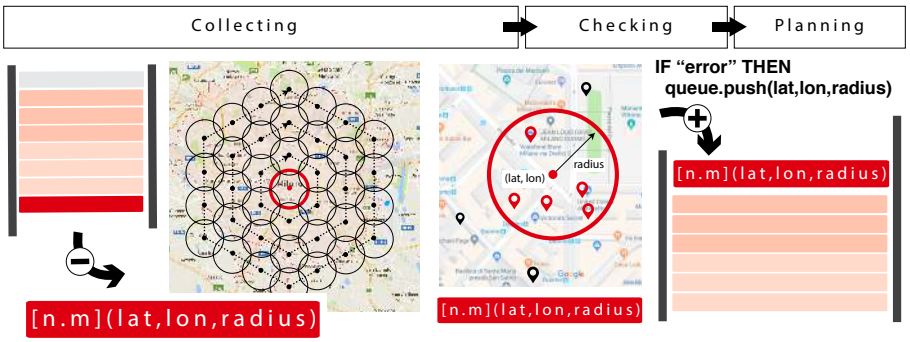


Figura 3.23: Errore generico e nuova pianificazione dell'area esagonale

Numero eccessivo di Place Mentre nel caso in cui viene rilevato un numero vicino o pari al limite di 60 viene richiesto alla componente `Planning()` di pianificare una cattura a grana più fine della sola area interessata, costruendo quindi una griglia composta da celle più piccole che coprono la sola area interessata, ognuna delle quali aggiunte alla coda per essere utilizzate (vedi figura 3.24).

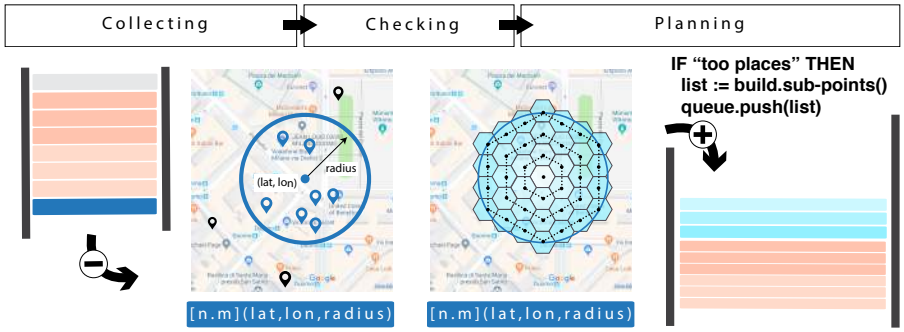


Figura 3.24: Griglia con celle più piccole della sola area interessata

Attualmente il software crea un griglia secondaria composta di 4 anelli di esagoni con raggio pari ad 1/3 rispetto al raggio dell'area di cui si necessita un'analisi più approfondita, ma tale rapporto di granularità può essere anche diverso.

3.3.4 Memorizzazione

La memorizzazione dei Place viene effettuata su database NO-SQL MongoDB che si adatta perfettamente agli output delle risposte in formato JSON del servizio di Google.

Duplicati A causa della sovrapposizione delle circonferenze approssimate a degli esagoni è inevitabile avere certo numero di Place duplicati, che sono comunque ridotti al minimo. Tale verifica dei

duplicati è effettuata in un secondo momento rispetto alla raccolta dei dati.

Un esempio di Place duplicati è visibile nella figura 3.25, dove marker verdi rappresentano i Place catturati da una sola richiesta, mentre i marker rossi i Place catturati da più richieste, queste ultime rappresentate dalle piccole circonferenze verdi.

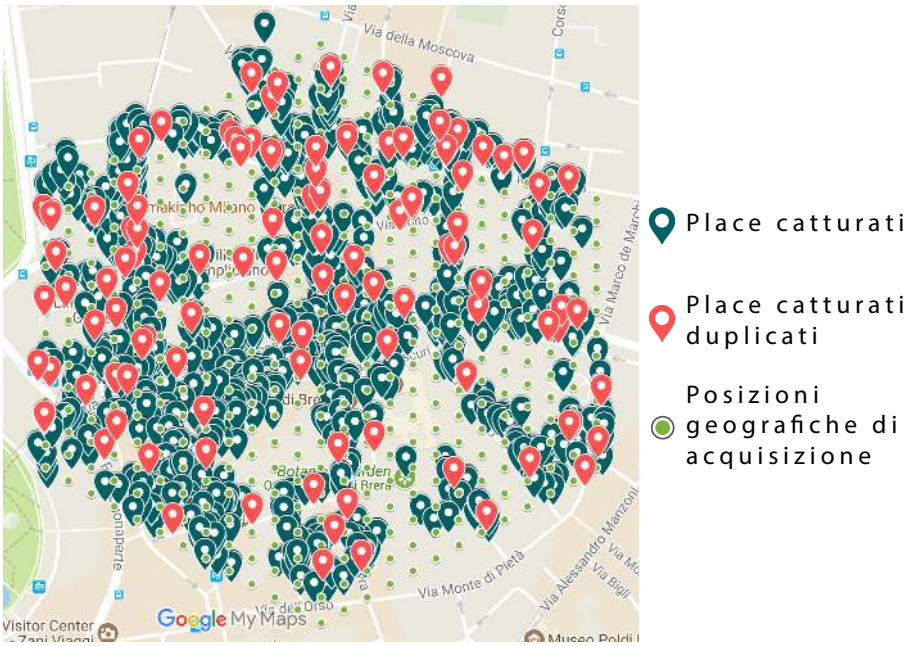


Figura 3.25: Place catturati utilizzando circonferenze approssimate ad esagoni dove le icone rosse sono i Place catturati da più richieste

Acquisizione multiple La Raccolta dati complessiva dell'area di Milano è avvenuta in fasi separate invece che in unica esecuzione per monitorare meglio l'esecuzione e quindi poter intervenire tempestivamente. In oltre a causa della natura sperimentale e dei lunghi tempi di esecuzione dell'algoritmo abbiamo ritenuto utile poter disporre ed analizzare i Place dell'area centrale della città mentre l'algoritmo di Raccolta dati eseguiva la raccolta delle aree più esterne.

Le fasi di Raccolta dati sono: dal anello 0 al 15, 16-20, 21-26, 27-32, 33-50, 51-61, 62-73, 74-84, 85-140, 140-180, 180-200, 200-230 (vedi figura 3.26).

Osservando i numeri degli anelli è possibile osservare in quelli più esterni alcune ripetizioni (i.e. 85-140, 140-180), esse non sono un errore ma bensì volutamente ripetuti per motivazioni di sicurezza ed accuratezza dei dati. Non avendo a disposizione tempo a sufficiente per i test si è preferito essere certi di catturare ogni area della città.

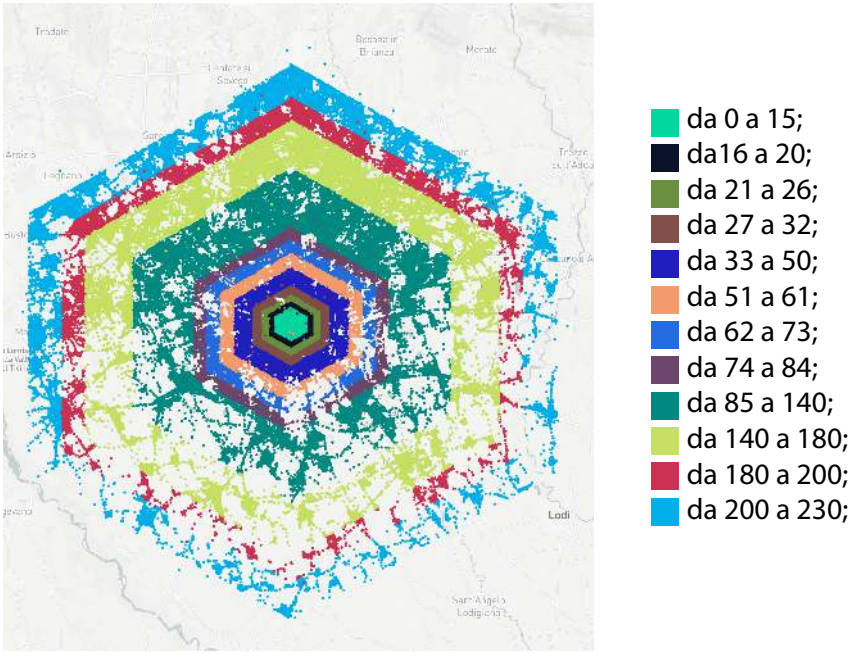


Figura 3.26: Raccolta Dati dei Place nell'area di Milano divisa per step

Struttura Database Ogni fase di Raccolta dati $n \rightarrow m$ è memorizzata in un *Database_n_m* differente dove per ogni *Database_n_m* informazioni relative ai dati acquisiti, alle aree analizzate ed eventuali errori di acquisizione. In particolare troviamo quattro collection principali:

- Collection Response: utilizzata per memorizzare ogni singola risposta ottenuta dal servizio di Google;
- Collection Places: utilizzata per memorizzare i Place estratti;
- Collection RequestHexagon: utilizzata per memorizzare informazioni sulle aree analizzate durante la cattura;
- Collection Log: utilizzata per memorizzare ogni singolo evento di interesse avvenuto durante la Raccolta dati;

3.4 Analisi dei risultati

La Raccolta dati attraverso le API di Google ha interessato un'area pari ad una circonferenza di circa $27,6km$ centrata nel Parco Sempione, attraverso una griglia primaria composta da celle esagonali di raggio pari $60metri$.

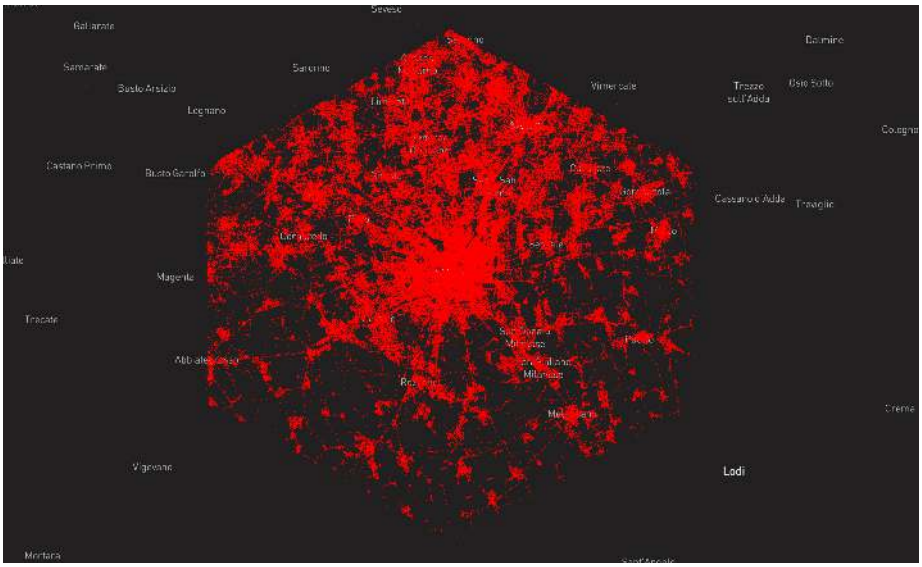


Figura 3.27: Tutti i Place raccolti attraverso l'algoritmo di Raccolta Dati

L'intera area, come accennato precedentemente, è stata analizzata per step composti da un numero variabile di anelli, quest'ultimo è stato scelto di volta in volta in modo empirico in base all'andamento dell'analisi precedente, ad esempio se l'ultima analisi eseguita aveva impiegato troppo tempo nella analisi successiva veniva ridotto il numero di anelli.

Anelli	Celle	Place Tot.	*avg* Place/Cella	Place Dupl. (%)	Griglia secondaria
$0 \rightarrow 15$	721	54531	48,5	75,6%	Si
$16 \rightarrow 20$	540	27511	50,9	39,1%	Si
$21 \rightarrow 26$	846	27267	32,2	30,1%	Si
$27 \rightarrow 32$	1062	20674	19,4	18%	Si
$33 \rightarrow 50$	4482	38660	8,6	21%	Si
$51 \rightarrow 61$	3696	14760	3,9	34,4%	No
$62 \rightarrow 73$	4860	16951	3,4	34,2%	No
$74 \rightarrow 84$	5214	18351	3,5	31,5%	No
$85 \rightarrow 140$	37800	86994	2,3	34,8%	No
$140 \rightarrow 180$	39360	73128	1,8	37,4%	No
$180 \rightarrow 200$	23940	33482	1,39	38,1%	No
$200 \rightarrow 230$	39990	52845	1,3	39%	No
$0 \rightarrow 230$	Totali: 290000 Place distinti in $27.6km$				

Tabella 3.4: Dettgli sulle varie fasi della Raccolta Dati sul territorio di Milano

La tabella 3.1 elenca alcuni dettagli ulteriori delle varie fasi della

raccolta, in particolare mostra il numero di anelli pianificati, il numero di celle pianificate, i Place totali raccolti, il numero medio di Place per cella, la percentuale di duplicati ed infine se è stato necessario pianificare almeno una griglia secondaria. Si osserva che l'area centrale della città ha un numero medio di Place per cella molto più alto delle aree più esterne, come anche il numero di duplicati. In oltre dall'anello 51 in poi non è stato più necessario realizzare alcuna analisi approfondita, dunque le celle esagonali di raggio di 60 metri erano più che sufficienti.

3.4.1 Analisi visiva dei Place

In questa fase iniziale di osservazione del dataset è stata inoltre fatta un'analisi visiva della distribuzione del Place sul territorio di Milano per capire quale tipo di deduzioni o analisi potessero essere fatte successivamente su un dataset di questo tipo.

A colpo d'occhio è interessante osservare nella figura 3.27 come sia più densamente servita da servizi l'area centrale della città rispetto alle aree periferiche, in oltre l'area sud è caratterizzata certamente da aree con quasi assenza di servizi in modo nettamente superiore all'area Nord. Si notano in modo abbastanza chiaro anche le vie di comunicazioni principali che partono dal centro di Milano per diramarsi verso le località circostanti attraverso una concentrazione evidente di servizi.

Mentre osservando la figura 3.28 è possibile osservare visivamente alcune aree *vuote* che per chi conosce la città di Milano riesce a riconoscere senza problemi.

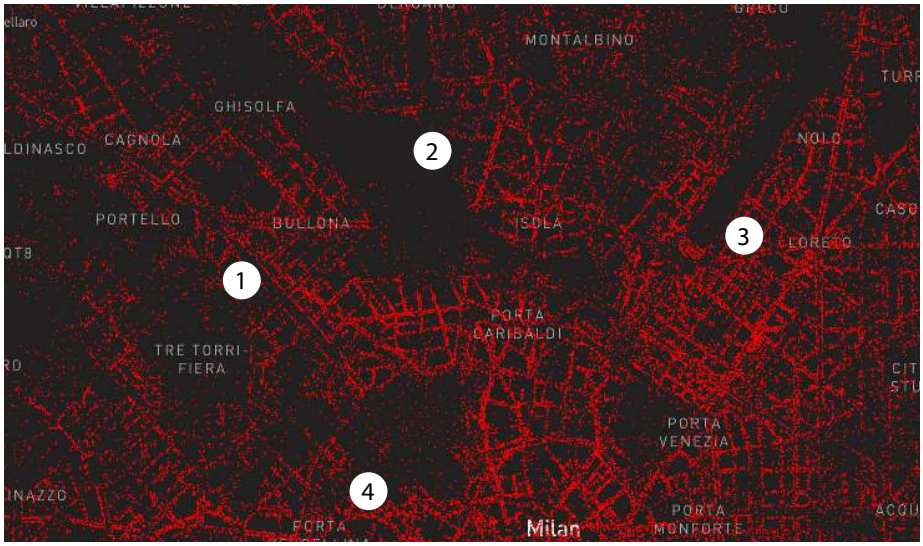


Figura 3.28: Aree apparentemente vuote e prive di servizi: l'area delle "Tre Torri" (1), Monumentale e Stazione Garibaldi (2), Stazione Centrale (3) e l'area del Castello Sforzesco e Parco Sempione (4)

I trasporti

Dalle immagini precedenti è interessante osservare già visivamente come la distribuzione dei Place e conformazione strade siano strettamente correlate.

Ad esempio estraendo i soli Place relativi ai trasporti (vedi figura 3.29 che comprende fermate della metro, del tram, dei bus, ed aeroporti e stazioni ferroviarie si evince un reticolo delle aree più servite dai servizi di trasporto che ricalca le vie principali principali della città. Anche in questo caso l'area centrale sembra essere più servita rispetto alle aree più esterne, tal volta disegnando vie e forme immediatamente riconoscibili.

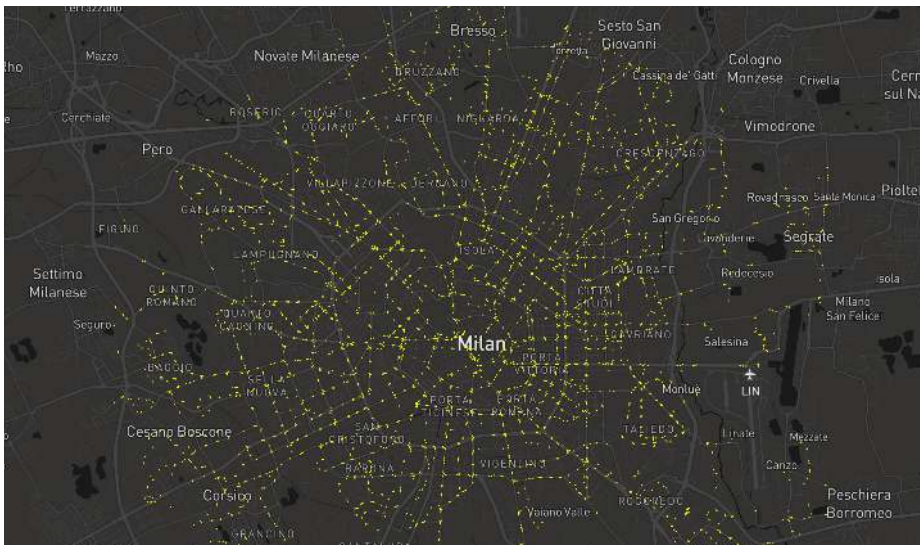


Figura 3.29: Place relativi a servizi di trasporto a Milano

Ulteriore analisi effettuata è stata quella di enfatizzare il confronto tra quantità di Place totale e presenza di un servizio di trasporto (vedi figura 3.30).

Le aree con più servizi (quelle in bianco) sembrano essere maggiormente concentrate intorno alle vie di comunicazione principali e in punti maggiormente servite da un servizio di trasporto.

3.4.2 Analisi dei Tipi

Nelle sezioni precedenti si è parlato del campo `type` dei Place che contiene alcuni termini che identificano il tipo di servizio. Una delle attività svolte è stata appunto quella di analizzare i Place catturati dal punto di vista dei `type`. Google non fornisce molte informazioni su di essi, tranne quali valori si possono incontrare. Nella tabella 3.5 si può osservare un elenco del numero di Place catturati per singolo `type`.

Figura 3.30: Confronto tra presenza di servizi generici (in bianco) e presenza di servizi di trasporto

Place	type	Place	type	Place	type
240914	establisment	1374	church	240	university
240914	point_of_interest	1351	gas_station	226	storage
53591	route	1343	travel_agency	221	car_wash
43453	store	1243	shoe_store	195	movie_theater
19914	food	1160	park	179	cemetery
17125	health	1131	pharmacy	163	neighborhood
9927	restaurant	1081	car_dealer	157	locksmith
9385	finance	1022	local_government_office	141	pet_store
7049	clothing_store	997	premise	138	embassy
6998	bar	857	political	103	subway_station
6827	transit_station	857	hospital	88	train_station
6217	home_goods_store	810	laundry	85	roofing_contractor
5086	bus_station	728	shopping_mall	74	convenience_store
5020	general_contractor	724	post_office	64	stadium
4970	car_repair	706	light_rail_station	50	amusement_park
4854	lawyer	686	locality	44	courthouse
4799	hair_care	660	spa	34	department_store
4179	lodging	654	meal_takeaway	31	movie_rental
3300	doctor	626	city_hall	29	airport
2848	dentist	623	parking	29	mosque
2798	school	605	florist	27	fire_station
2589	accounting	603	night_club	22	campground
2588	furniture_store	545	car_rental	15	casino
2434	grocery_or_supermarket	533	art_gallery	13	rv_park
2410	electronics_store	514	funeral_home	11	subpremise
2405	beauty_salon	514	moving_company	11	taxi_stand
2298	bank	483	book_store	8	bowling_alley
2280	insurance_agency	469	veterinary_care	6	aquarium
1900	real_estate_agency	464	library	6	sublocality
1883	atm	446	hardware_store	6	sublocality_level_1
1866	cafe	444	painter	4	zoo
1865	jewelry_store	383	museum	3	synagogue
1854	bakery	330	liquor_store	3	natural_feature
1716	plumber	291	meal_delivery	2	hindu_temple
1707	electrician	256	bicycle_store		
1695	place_of_worship	253	police		
1626	gym	244	physiotherapist		

Tabella 3.5: Elenco dei Type ordinati per numerosità di Place

Su un totale di 294505 si è osservato che si dividevano in due gruppi completamente divisi, il primo composto da 53591 Place identificati come Strade attraverso il `type = road`, ed un secondo gruppo di 240914

Place identificati come servizi e Point of Interest (POI) di varia natura identificati dal `type = point_of_interest`. Tutti gli altri `type` possibili fanno parte di uno o dell'altro gruppo.

Google non indica la presenza di alcuna gerarchia dei `type` anche se in alcuni casi esiste una forte correlazione tra alcuni `type`. Ad esempio il `type = point_of_interest` è onnipresente in tutti Place mentre il `type = transport` è sempre presente in tutti i Place relativi ai trasporti, sia che esso sia una stazione de metropolitana se se esso è una stazione ferroviaria. La stessa cosa accade con alcuni servizi relativi al cibo dove il `type = food` è correlato spesso con i `bar` o `restaurant`.

Un estratto delle combinazioni di type più popolose di Place è disponibile con la tabella 3.6.

type (type1, type2...)	n. type	n. Place
[u'point_of_interest']	1	102687
[u'route']	1	53591
[u'point_of_interest', u'store']	2	10951
[u'food', u'point_of_interest', u'restaurant']	3	7780
[u'clothing_store', u'point_of_interest', u'store']	3	6589
[u'health', u'point_of_interest']	2	5375
[u'bar', u'point_of_interest']	2	5151
[u'bus_station', u'point_of_interest', u'transit_station']	3	4924
[u'lawyer', u'point_of_interest']	2	4742
[u'general_contractor', u'point_of_interest']	2	4274
[u'lodging', u'point_of_interest']	2	3974
[u'hair_care', u'point_of_interest']	2	3663
[u'car_repair', u'point_of_interest']	2	3300
[u'doctor', u'health', u'point_of_interest']	3	3031
[u'finance', u'point_of_interest']	2	2863
[u'point_of_interest', u'school']	2	2702
[u'dentist', u'health', u'point_of_interest']	3	2696
[u'home_goods_store', u'point_of_interest', u'store']	3	2491
[u'accounting', u'finance', u'point_of_interest']	3	2484
[u'furniture_store', u'home_goods_store', u'point_of_interest', u'store']	4	2419
[u'food', u'grocery_or_supermarket', u'point_of_interest', u'store']	4	2208
[u'insurance_agency', u'point_of_interest']	2	2172
[u'food', u'point_of_interest', u'store']	3	1906
[u'jewelry_store', u'point_of_interest', u'store']	3	1733
[u'food', u'point_of_interest']	2	1700
...	...	...

Tabella 3.6: Elenco delle combinazioni di Type più numerose

3.4.3 Commenti

I `type` offrono una visione particolare della città, seppur sembrano coprire un ampio spettro di settori dal ludico all'istruzione, è chiaro che i settori più curati siano quelli relativo al tempo libero, questo è probabilmente dovuto ad un più forte interesse di questo settore ad utilizzare la piattaforma Google per scopi pubblicitari.

Un'altra categoria molto curata è quella relativa ai trasporti, forse dovuto ad un'automazione o sistematizzazione della gestione e gestione dei Place, motivo per cui si trova non solo un alto grado di precisione anche un alto grado di correlazione di `type`. In oltree non è dato sapere se un `type` o un Place è creato da un essere umano o una macchina.

I `type` con cui si possono caratterizzare i Place sono decisi da Google, questo implica una punto di vista preciso verso determinate categorie piuttosto che altre.

Capitolo 4

Data Mining

Introduzione

Il passo successivo di analisi dei dati è il Processo di clustering dei *Place* geolocalizzati che ci permette di ridurre il dataset ad un numero ragionevole di sottogruppi, a loro volta di dimensione sufficientemente contenuta da permette l'analisi e la caratterizzazione della città nelle sue diverse aree.

I metodi di Clustering possono essere usati in diverse modalità, ma in questo caso specifico è utilizzato principalmente per ridurre il dataset in unità discrete sulle quali è possibile applicare ulteriori metodi di analisi.La scelta finale è ricaduta sull'algoritmo basato sulla densità DBSCAN grazie alla sua predisposizione naturale verso i dati spaziali ed il basso grado di conoscenza del dominio richiesto per poter essere applicato.

Suddividere la Città Lo scopo di questa sezione e descrivere come è stata suddivisa la città allo scopo di caratterizzare le diverse aree della città. In particolare viene proposta una suddivisione della città bottom-up che emerge dai dati stessi, che in questo preciso contesto sono rappresentati dai Place raccolti sul territorio di Milano.

Realizzando così una suddivisione della città di Milano alternativa, rispetto alle suddivisioni top-down della sezione 1.3.1, ottenuta per mezzo di un Processo di clustering iterativo basato sul DBSCAN.

4.1 Requisiti e libreria

Il DBSCAN, come descritto nella sezione 2.3.3, è un algoritmo di Clustering basato sulla densità molto popolare e versatile, applicabile ad un gran numero di dataset, tra i quali anche quelli geospaziali.

La versatilità del DBSCAN è basata sulla possibilità di applicare il metodo a qualsiasi dataset caratterizzato da item dotati di coordinate spaziali, attraverso l'applicazione del concetto di densità per mezzo di una formula per il calcolo della distanza adeguata al tipo di coordinate.

Al fine di applicare correttamente l'algoritmo di Clustering è stato necessario analizzare i dati di input e la formula adeguata per il calcolo della distanza e scegliere una delle implementazioni dell'algoritmo disponibili. Descriverò brevemente questi aspetti nelle sezioni successive.

4.1.1 I dati geospaziali

Il dataset utilizzato è quello ottenuto attraverso il processo di Raccolta dati (sezione 3.2) rappresentato da un insieme D di item p_i dotati di coordinate geospaziali, composte da *latitudine* e *longitudine*, tale che $D = \{p_1, p_2, ..., p_i\}$, $p_i = (lat_{p_i}, lon_{p_i})$, $lat_{p_i} = latitudine$, $lon_{p_i} = longitudine$ entrambe espresse in radianti.

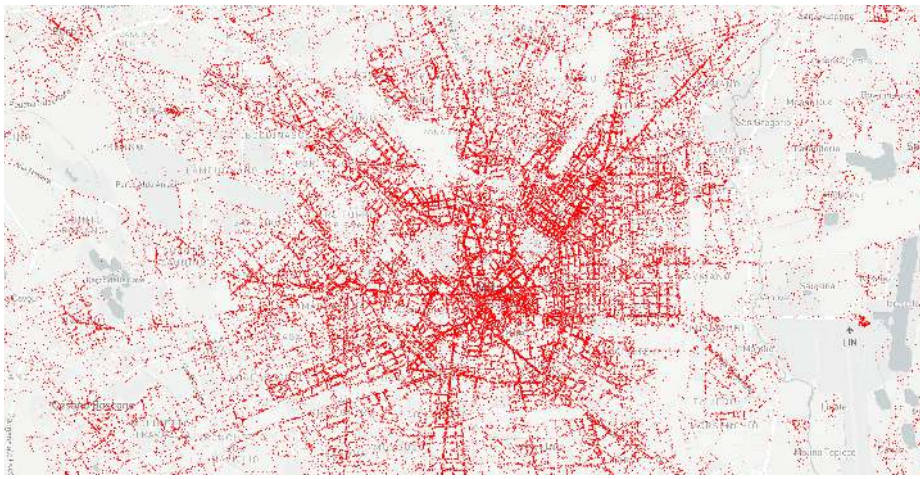


Figura 4.1: Estratto dei Place raccolti su territorio di Milano che rappresentano parte del dataset utilizzato per il processo di Clustering

4.1.2 Il calcolo della distanza

La terra è un geoide ovvero una sfera leggermente schiacciata ai poli approssimata in prima istanza da un elissoide di riferimento ed in secondo luogo da una sfera di raggio circa 6.371 km, per tale motivo è necessario utilizzare opportune formule per calcolare le distanze tra punti.

Distanza di cerchio massimo Un modo per misurare la distanza tra due punti geografici sulla terra è quello di calcolare la lunghezza dell'arco della circonferenza più grande che passa tra questi due punti, tale circonfenrenza è detta distanza di cerchio massimo. [9]

Essa è detta anche distanza sferica, è la distanza più breve tra due punti sulla terra calcolata sulla sfera di riferimento di raggio medio 6.371 km.

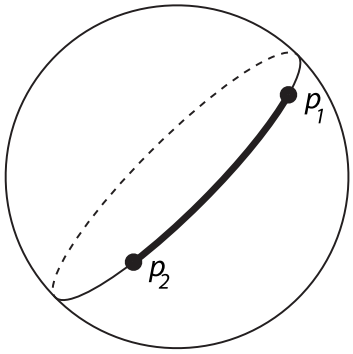


Figura 4.2: Distanza di cerchio massimo calcolata sulla circonferenza più grande che passa tra due punti

Dati due punti $p_1 = (\varphi_1, \lambda_1)$ e $p_2 = (\varphi_2, \lambda_2)$ con coordinate espresse rispettivamente in $(latitudine, longitudine)$

La distanza sferica d_{circle} è definita come:

$d_{circle} = r\,\Delta\sigma$ dove l'arco $\Delta\sigma$ corrisponde a

$\Delta\sigma = \arccos\big(\sin\varphi_1 \cdot \sin\varphi_2 + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \cos(\Delta\lambda)\big)$

Formula di haversine Da un punto di vista computazionale la distanza di cerchio massimo approssimabile attraverso la Forumla di Haversine, dove la distanza tra due punti geografici diventa:

$$d_{hav} = 2r \arcsin\left(\sqrt{\sin^2\left(\tfrac{\varphi_2-\varphi_1}{2}\right) + \cos\varphi_1 + \cos\varphi_2 \sin^2\left(\tfrac{\lambda_2-\lambda_1}{2}\right)}\right)$$

4.1.3 La libreria scikit-learn

Per questo progetto si è scelto di utilizzare l'implementazione DBSCAN resa disponibile dalla libreria Python `scikit-learn`[1] che, attraverso diversi parametri di input, offre un'ampia possibilità di configurazioni.

I parametri minimi per ottenere un Clustering attraverso questa implementazione del DBSCAN sono la distanza *eps*, il numero di punti minimo *minPts* e il dataset di input formattato in un array di tuple $X = [(x, y)_1, (x, y)_2, ...]$.

```
DBSCAN(eps=<eps>, min_samples=<minPts>).fit(X)
```

Nel nostro caso, dove i punti sono espressi su coordinate geospaziali,

[1] `http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html`

è necessario esprimere la formula di distanza e la struttura dati da utilizzare per l'ordinamento dei punti, che diventano rispettivamente la Distanza di Haversine attraverso il parametro `metric='haversine'` e la struttura dati *ball tree* attraverso il parametro `algorithm='ball_tree'`, ottenendo, così, le seguenti righe di codice utilizzate per realizzare i Clustering di questo progetto:

```
clustering = DBSCAN(
  eps = <eps>,
  min_samples = <minPts>,
  metric = 'haversine',
  algorithm = 'ball_tree',
  n_jobs = <1, 2, 3 ...>
  )
```

Disponendo di una CPU multi-core, è possibile migliorare l'esecuzione parallela con il parametro `n_jobs`, che può essere configurato fino ad un valore massimo equivalente al numero di core disponibili sul proprio computer.

Il Clustering Al termine del Processo di clustering, l'algoritmo DBSCAN avrà etichettato ogni elemento del dataset X con il relativo numero del Cluster di appartenenza, che sarà:

- ≥ 1 se appartenenti ad un Cluster valido
- $= 0$ se identificati come rumore

In generale l'implementazione restituisce etichette di Cluster comprese tra $0 \rightarrow n$ attraverso la variabile `labels = clustering.labels_`, dove Cluster $= 0$ conterrà tutti i punti etichettati come rumore umore.

Silhouette Score L'implementazione `scikit-learn` offre in oltre la possibilità di calcolare alcuni valori utili alla valutazione della qualità del Clustering come ad esempio l'utile *Silhouette Score* che è calcolabile, al termine del processo, importando adeguatamente il modulo `metrics` e richiamando l'opportuno metodo $silhouette_{score}$:

```
metrics.silhouette_score(X, labels))
```

4.2 Clustering Gerarchico-Iterativo basato sul DBSCAN

Il metodo proposto in questa sezione realizza una suddivisione della città sulla base della densità dei place attraverso un processo di clustering spaziale gerarchico per mezzo di un processo iterativo (vedi figura 4.3).

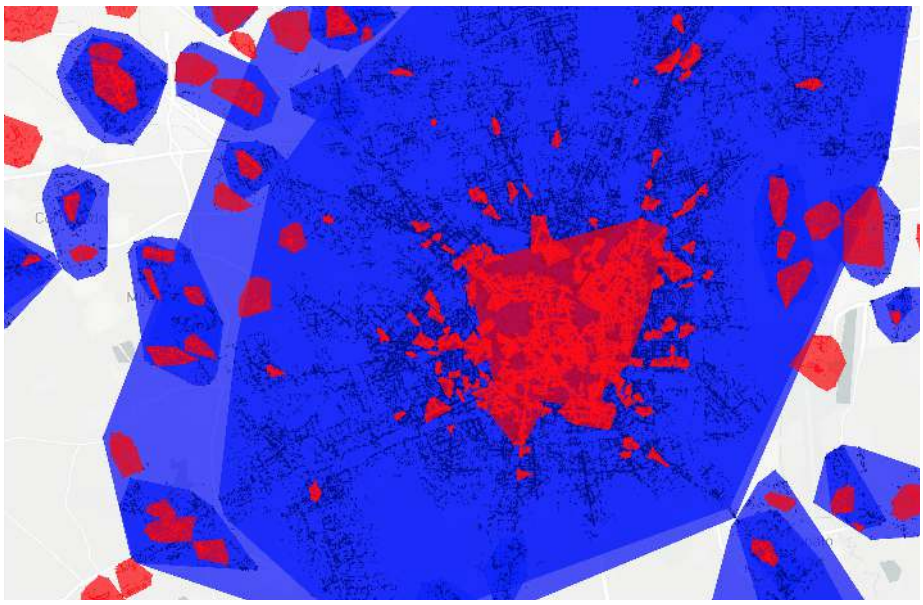


Figura 4.3: Estratto reale del clustering realizzato sul territorio di Milano attraverso il processo di Clustering gerarchico-iterativo basato sul DBSCAN

Più specificatamente, è un processo di clustering basato sul DBSCAN che divide iterativamente il dataset in cluster di densità diversa organizzandoli in una gerarchia di cluster incapsulati tra loro attraverso un processo iterativo basato su tre concetti fondamentali:

- insieme di clustering diversi calcolati su uno stesso di input;
- l'uso di metriche e valutazioni dei clustering dell'insieme;
- riutilizzo di cluster come input di nuovi processi di clustering.

4.2.1 Proprietà del Clustering di output

Il metodo di realizza un clustering gerarchico del dataset attraverso cluster di densità diversa organizzati in un albero non necessariamente bilanciato, partendo dal cluster radice rappresentato dal dataset iniziale a densità minima, sino ad arrivare ai cluster foglia con una densità via-via superiore, questi ultimi non necessariamente posizionati alla stessa profondità (vedi figura 4.4).

Data la complessità dell'algoritmo che verrà descritto in queste sezioni saranno utilizzate immagini grafiche non corrispondenti al reale output del clustering (i.e. 4.4), ma rappresentative di quello che verrà descritto allo scopo di semplificare la descrizione.

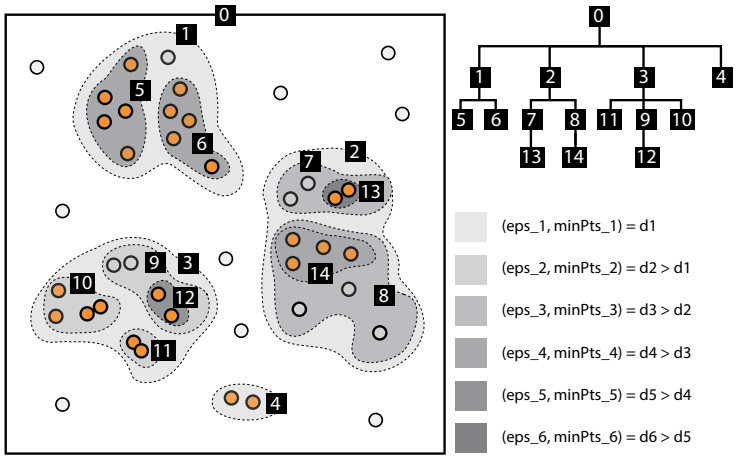


Figura 4.4: Clustering gerarchico-iterativo con Cluster foglia a densità diversa

Nei paragrafi successivi verranno affrontati alcuni concetti di base che caratterizzano il clustering gerarchico di output al fine di fornire una terminologia comune delle sezioni successive. Il clustering gerarchico realizzato si caratterizza per 3 fatti fondamentali:

- ogni sotto-albero ha profondità diverse;
- ogni sotto-albero è caratterizzato da densità diverse;
- estraendo i soli cluster foglia essi hanno potenzialmente densità diverse.

Profondità dei sotto-alberi

Ogni cluster appartiene ad un determinato livello dell'albero in base all'iterazione in cui è stato estratto, caratterizzandolo come *figlio*, *padre* o *fratello* di altri cluster, ad esempio considerando l'immagine in figura 4.5 è possibile analizzare il clustering per diverse altezze dell'albero.

Preso un cluster intermedio c_x con padre z_{cx} e figlio f_{cx}, si ha che $f_{cx} \subseteq c_x \subseteq z_{cx}$, ovvero che ogni sotto-albero è composto degli stessi punti del cluster che rappresenta la radice del sotto-albero. Ad esempio prendendo il cluster $c_x = c_7$ della figura 4.5 si ottiene che $c_{13} \subset c_7 \subset c_2$ e $((c_{13} \subset c_7) \cup c_8) \subset c_2$.

Dunque, un cluster figlio è sempre più piccolo o al massimo uguale al cluster padre, mentre non c'è correlazione tra fratelli o tra cluster di pari livello, ovvero è possibile dedurre che la numerosità dei Cluster valga $|c_5| < |c_1|$, ma allo stesso tempo non è possibile dedurre che la numerosità dei Cluster valga $c_5 > c_6$ oppure $c_5 < c_6$.

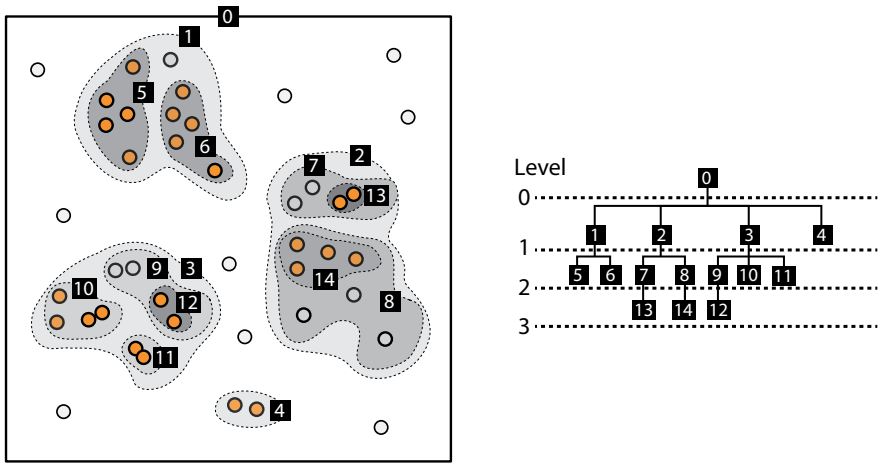


Figura 4.5: Clustering gerarchico-iterativo rappresentato in un albero suddiviso per livelli

Infine ogni sotto-albero è disgiunto dagli altri sotto-alberi, questo vale anche per Cluster appartenenti ad uno stello livello dell'albero. Ma non sono mai disgiunti due cluster dove uno dei due è antenato dell'altro.

Ad esempio i Cluster 1,2,3 e 4 della figura 4.5 sono disgiunti tra di loro, lo sono anche i relativi sotto-alberi perchè essi sono derivati dai Cluster 1,2,3 e 4, ma i Cluster 2 e 14 non sono disgiunti perché uno è antenato dell'altro, quindi $c_14 \subset c_2$.

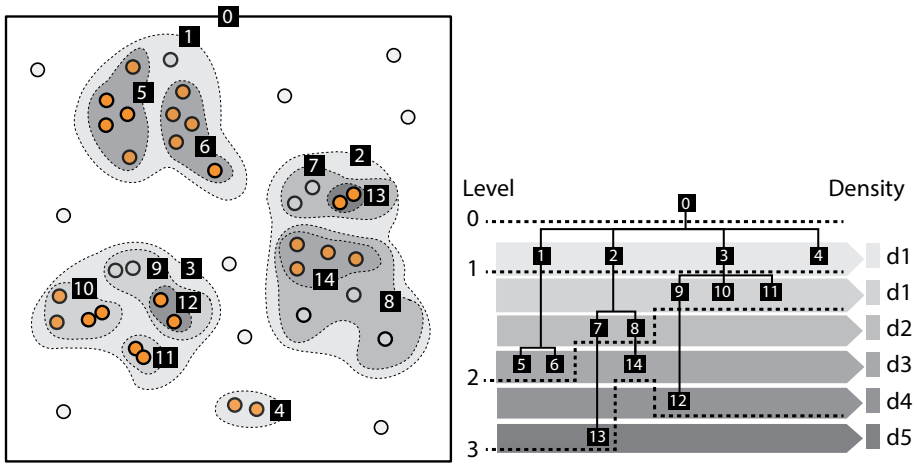
Densità dei Cluster

Definita la struttura dell'albero gerarchico dei cluster possiamo passare a definire la densità relativa dei vari cluster. Una diretta conseguenza del metodo di estrazione dei cluster del presente algoritmo è che i cluster di pari livello, ad esempio $Cluster_{level\_1} = c_1, c_2, c_3, c_4$, non è detto che siano caratterizzati dalla stessa densità.

Più specificatamente, preso un cluster generico c_x con padre z_x e figlio f_x:

- la densità di un figlio è sempre maggiore o al massimo uguale alla densità del padre $density(f_x) > density(c_x) > density(z_x)$
- la densità della radice è sempre più bassa delle densità di qualsiasi altro cluster $density(root) > \forall c_x$
- la densità relativa tra cluster di uno stesso livello, ma di sotto-alberi, non ha alcuna correlazione perché dipende dai rispettivi padri;

Prendendo ad esempio i cluster del primo livello $Cluster_{level\_1} = c_1, c_2, c_3, c_4$ si ottiene che otteniamo che $density(c_1) = density(c_2) =$



(a) Albero dei cluster per livelli (b) Albero dei Cluster per densità

Figura 4.6: Clustering gerarchico-iterativo per livelli e densità

$density(c_3) = d1 = density(c_4) = d1$ perché sono tutti figli dello stesso padre

Mentre prendendo i cluster del terzo livello $Cluster_{level\_3} = \{c_{12}, c_{13}, c_{14}\}$ si ottiene che $density(c_{12}) = d4 \neq density(c_{13}) = d5 \neq density(c_{14}) = d3$ ovvero che a priori hanno tutti densità diversa perché sono tutti figli di padri diversi, se invece esiste una corrispondenza di densità tra Cluster di sotto-alberi diversi, essa è solo un effetto casuale, come nel caso dei cluster c_5, c_{14} con $density(c_{14}) = density(c_5)$.

Cluster foglia e cluster intermedi

Infine i concetti chiave del clustering descritto in queste sezioni sono i:

- cluster foglia, chiamati Cluster indivisibili;
- cluster intermedi, chiamati sotto-dataset.

I Cluster foglia, sono cluster che non è possibile suddividere ulteriormente, ad esempio analizzando la gerarchia in figura 4.7 composta dai Cluster 4, 5, 6, 10, 11, 12, 13 e 14 essi sono caratterizzati da:

- padri differenti, a meno di non essere fratelli;
- densità tra loro differenti, ma sempre maggiore dei loro padri (o antenati);
- distanza dalla radice tra loro differente, a meno di non essere fratelli;

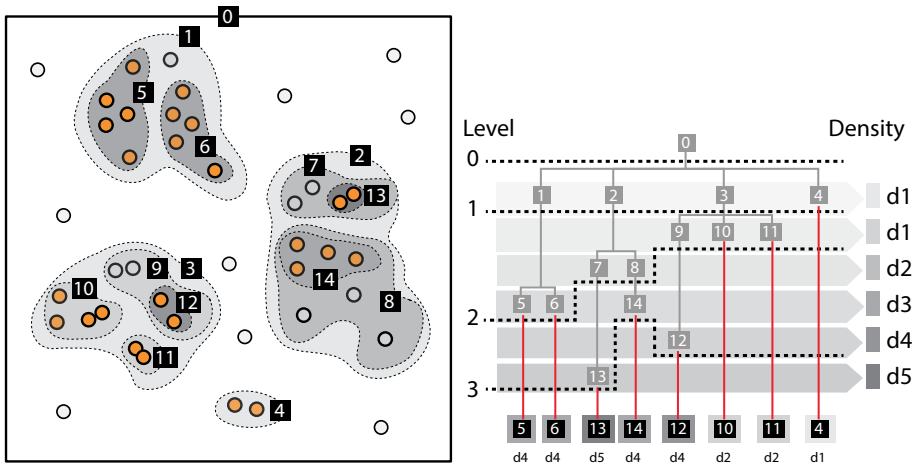


Figura 4.7: Cluster foglia a densità diversa non ulteriormente divisibili

Mentre i cluster intermedi sono detti sotto-dataset, perché oltre a essere Cluster sono anche dataset utilizzati come input per sotto-processi di clustering.

4.2.2 Il processo iterativo di Clustering

Il processo di Clustering che estrae ed organizza i cluster in una gerarchia incapsulata è un processo iterativo di clustering composto da tre componenti principali (vedi figura 4.8):

- la componente `Building` che realizza l'insieme di clustering attraverso l'esecuzione di più processi di DBSCAN;
- la componente `Selecting` che seleziona un Clustering dall'insieme di clustering attraverso un metodo di selezione basato su KPI e query;
- in fine la componente `Extracting` che estrae e decide quali cluster siano adeguati diventare sotto-dataset per ulteriori processi clustering.

Tale processo iterativo è modificabile nel suo comportamento attraverso tre input utente:

- range di valori, $eps = \{\epsilon_1, \epsilon_2, ..., \epsilon_u\}$ e $minPts = \{m_1, m_2, ..., m_j\}$;
- metodo di selezione del clustering migliore basato su KPI ed indicatori, $KPI_{solution} = \{k_{s1}(), k_{s2}(), ..., k_{st}()\}$

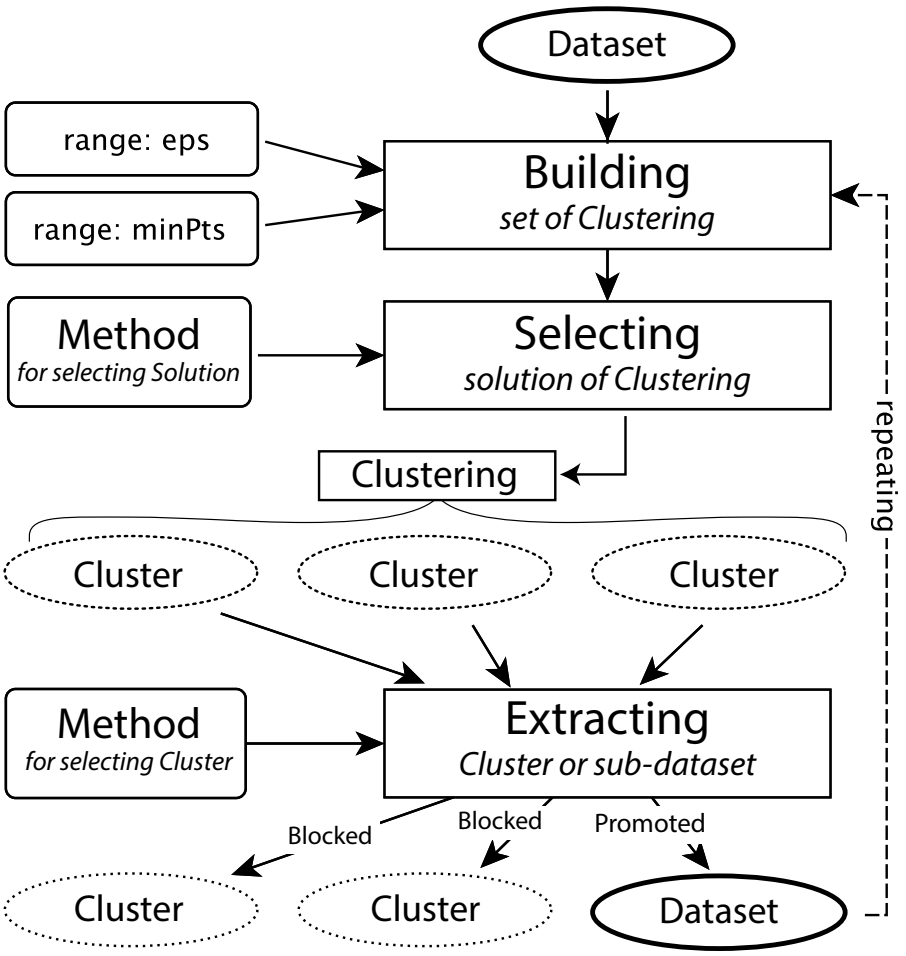


Figura 4.8: Processo iterativo di Clustering basato sul DBSCAN che realizza un Clustering gerarchico

- metodo di valutazione dei cluster compatibili ad essere utilizzati come sotto-dataset, metodo anch'esso basato su indicatori e KPI, $KPI_{cluster} = \{k_{c1}(), k_{c2}(), ..., k_{cr}()\}$

Tale processo iterativo è applicabile a qualsiasi cluster dell'albero, purché sia considerato ulteriormente divisibile dalla componente `Extracting` e quindi promosso a sotto-dataset (vedi figura 4.8).

Ad esempio, considerando il cluster c_3 della figura 4.9 esso può essere ancora utilizzato come input per il processo iterativo di clustering e produrre i cluster foglia c_{10} e c_{11}, ed un nuovo sotto-dataset c_9 come input per un nuovo processo.

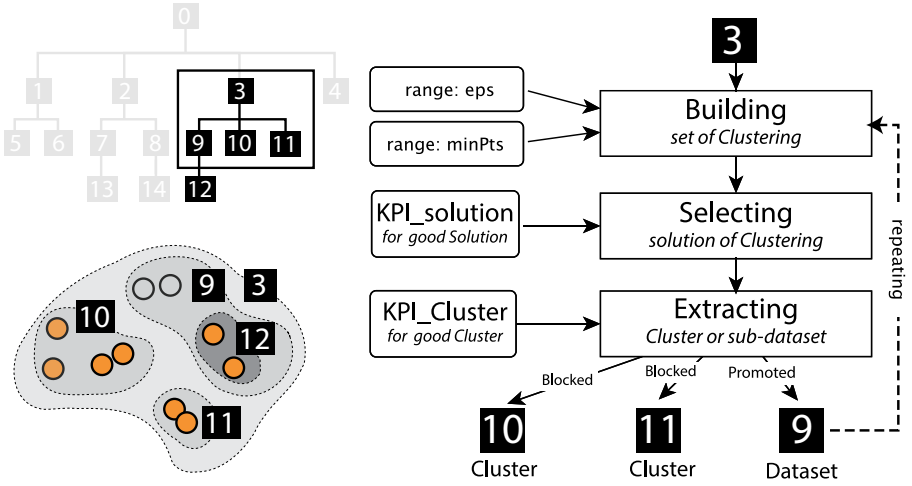


Figura 4.9: Porzione di Clustering gerarchico ed input/output del processo di Clustering iterativo

4.2.3 Insieme dei Clustering di uno stesso dataset

L'insieme dei clustering è il concetto che ha permesso la realizzazione del processo di clustering descritto in questa sezione e consiste in un insieme composto da più clustering ottenuti dall'applicazione di più esecuzioni dell'algoritmo DBSCAN ad uno stesso dataset di input, variando i parametri di configurazione *eps* e $minPts$.

Prendiamo ad esempio un dataset D al quale applichiamo un processo di clustering $DBSCAN(D, eps, minPts)$ con specifici parametri $eps, minPts$ otterremo uno specifico Clustering $C_{1(D,eps,minPts)}$ composto da determinati cluster $\{c_1, c_2, ..., c_n\}$.

Dunque, fissando D e variando al contempo $eps_2, minPts_2$ otterremo un clustering diverso dal primo $C_2 \neq C_1$, ma potenzialmente valido. Dunque, in assenza di conoscenza su quali valori di $eps, minPts$ utilizzare, quale dei due Clustering è il migliore?

La risposta consiste nel costruire un set di diverse soluzioni di clustering, che chiameremo insieme dei clustering, tale da fornire una panoramica sufficientemente chiara sulle soluzioni possibili e applicare una o più metriche tramite l'uso di Key Performance Indicator (KPI) e scegliere la soluzione di clustering migliore.

Definizione dell'Insieme di Clustering Fissato un dataset D ed un processo di $DBSCAN(D_{fixed}, \epsilon_e, \omega_m)$, a cui forniamo un set di valori $Eps = \{\epsilon_1, \epsilon_2, ..., \epsilon_e\}$ $MinPts = \{\omega_1, \omega_2, ..., \omega_m\}$ con $\omega_m \in N$, $\omega_m \in R$, è possibile applicare il $DBSCAN(D_{fixed}, \epsilon_e, \omega_m)$ per ogni valore di $Eps, MinPts$ ottenendo un set bidimensionale di clustering dello stesso

dataset.

(4.1)

$$SetOfClustering_{(D,EPS,MinPts)} =$$
$$\{C_{i,j} \mid C_{i,j} = DBSCAN(D_{fixed.}, \epsilon_i, \omega_j)\ \forall \epsilon_i \in Eps, \omega_j \in MinPts\} =$$
$$= \begin{bmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,m} \\ C_{2,1} & C_{2,2} & \dots & C_{2,m} \\ \vdots & \vdots & C_{i,j} & \vdots \\ C_{e,1} & C_{e,2} & \dots & C_{e,m} \end{bmatrix}$$

Dove, prendendo tre ipotetiche soluzioni $C_{5,2}, C_{7,5}, C_{7,8}$ esse saranno tre soluzioni differenti dello stesso dataset D con valori $\epsilon_i \in Eps, \omega_j \in MinPts$ differenti, ognuna con un suo potenziale di validità.

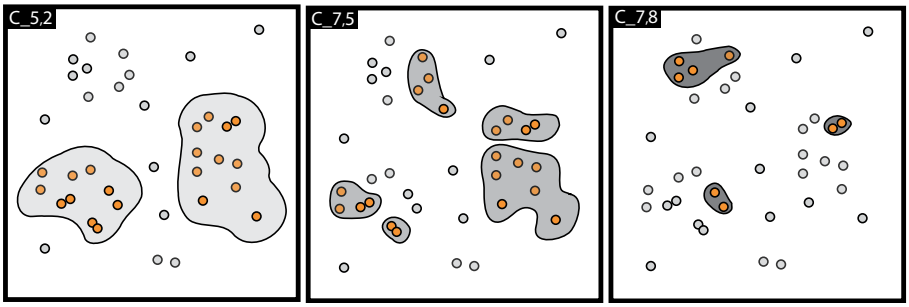


Figura 4.10: Diverse soluzioni di Clustering di uno stesso dataset prese dall'insieme dei Clustering

Una soluzione $C_{i,j}$ è uno specifico clustering del dataset ottenuto da una determinata combinazione di parametri $\epsilon_e, minPts$ presi rispettivamente dagli insiemi $\epsilon_i \in EPS$ e $\omega_j \in Min_{pts}$, (vedi figura 4.10).

Dunque, fissato D, quali e quante soluzioni $C_{i,j}$ possiamo avere nella matrice $SetOfClustering_{(D,EPS,MinPts)}$ dipende dai valori presenti negli insiemi $EPS, MinPts$ infatti la stessa matrice $SetOfClustering_{(D,EPS,MinPts)}$ ha dimensione $n \times m$ che dipende dalla dimensione dei due insieme $|EPS| = n$, $|MinPts| = m$.

4.2.4 Granularità dell'insieme dei Clustering

Una caratteristica dell'insieme dei clustering è data dalla possibilità di variarne la *granularità*, che equivale ad avere un numero di clustering maggiore o minore, rispetto ad uno stesso dataset di input.

Si può realizzare un insieme dei clustering a bassa granularità oppure uno a più alta granularità semplicemente variando la granularità

degli insiemi $EPS, MinPts$ input al processo di DBSCAN, ad esempio considerando due possibili range di valori $Eps_h, MinPts_h$ e $Eps_l, MinPts_l$ tale che:

$$|Eps_l| < |Eps_h|,\ |MinPts_l| < |MinPts_h|$$

si ottengono due insieme di Clustering a granularità differente:

$$SetOfClustering_{(D,Eps_l,MinPts_l)} = \\ = \begin{bmatrix} C_{1,1} & \dots & \dots & C_{1,m+r} \\ \vdots & \vdots & C_{i,j} & \vdots \\ C_{e+p,1} & \dots & \dots & C_{e+p,m+r} \end{bmatrix}$$

$$SetOfClustering_{(D,Eps_h,MinPts_h)} = \\ = \begin{bmatrix} C_{1,1} & \dots & \dots & \dots & \dots & C_{1,m+u} \\ \vdots & \vdots & \dots & \dots & \dots & \vdots \\ \vdots & \vdots & \dots & \dots & C_{k,z} & \vdots \\ \vdots & \vdots & \dots & \dots & \dots & \vdots \\ C_{e+t,1} & \dots & \dots & \dots & \dots & C_{e+t,m+u} \end{bmatrix}$$

Questo permette di scegliere il grado di precisione con cui generare l'insieme dei clustering, bilanciando complessità computazionale e precisione del risultato.

4.3 Valutazione dei Clustering

Realizzato un insieme di Clustering ottenuto da uno stesso dataset, è stato necessario realizzare un sistema che permettesse di caratterizzare tali Clustering in modo da poter scegliere quello che più fosse conforme alle aspettative. A tal proposito ad ogni Clustering sono stati associati uno o più indicatori che li caratterizzano, tale unione di Clustering e Indiciatori è chiamato Soluzioni.

Da Insieme dei Clustering a Insieme delle Soluzioni Dunque una soluzione non è altro che un Clustering caratterizzato da indicatori valutabili, dunque ricapitolando si ha:

- l'insieme dei Cluster, chiamato *Clustering*, che è un insieme di cluster di un dataset D tale che la loro unione è uguale ad dataset D;

- l'insieme dei Clustering che è l'insieme dei possibili *Clustering* di uno stesso dataset D ottenuto attraverso la variazione degli input $eps, minPts$;

- l'insieme delle Soluzioni è l'insieme dei possibili *Clustering* (ottenuti da uno stesso dataset) associati ad un insieme di Indicatori li caratterizzano;

4.3.1 Indicatori di qualità

Fissato un dataset D ed un insieme di valori $Eps = \{\epsilon_1, ..., \epsilon_n\}$ e $MinPts = \{\omega_1, ..., \omega_m\}$ l' i-esima soluzione $s_{i,j}$ è un oggetto complesso ottenuto da un unica esecuzione di $DBSCAN(D, \epsilon_i, \omega_j)$ tale che

$$DBSCAN(D, \epsilon_i, \omega_j) = Clustering_{i,j}$$

mentre la soluzione di tale Clustering è $s_{i,j} = \{Clustering, Indicators\}$ dove $Indicators$ sono indicatori del Clustering $Clustering_{i,j}$, quest'ultimo una specifica ri-organizzazione del dataset D in cluster rispetto ai parametri passati al DBSCAN.

Dunque una soluzione è $s_{i,j} = \{C_{i,j}, I(C_{i,j})\}$

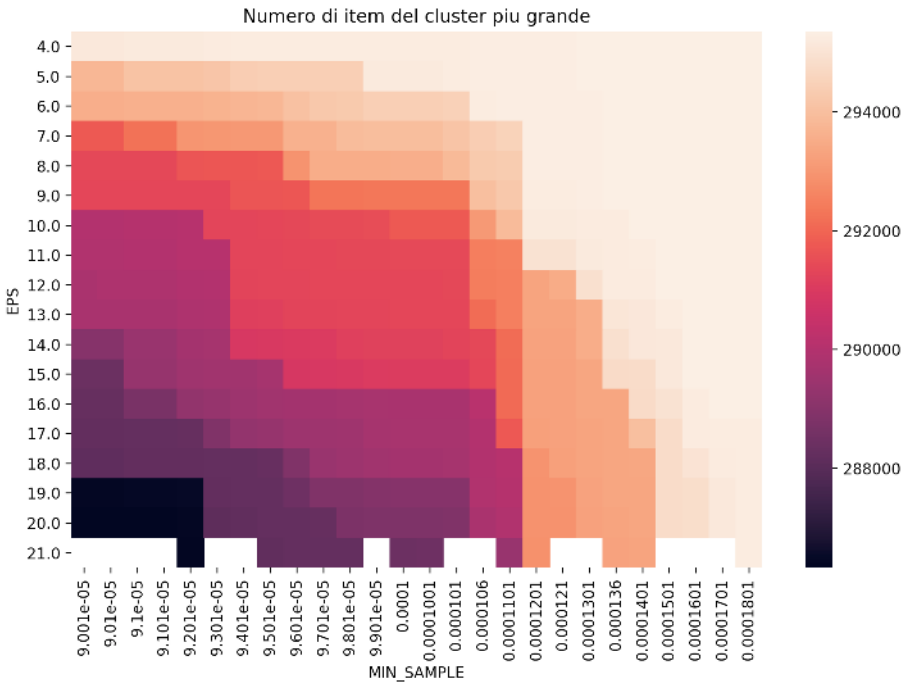


Figura 4.11: Numero di Place del Cluster più numeroso in un insieme di Clustering a bassa granularità

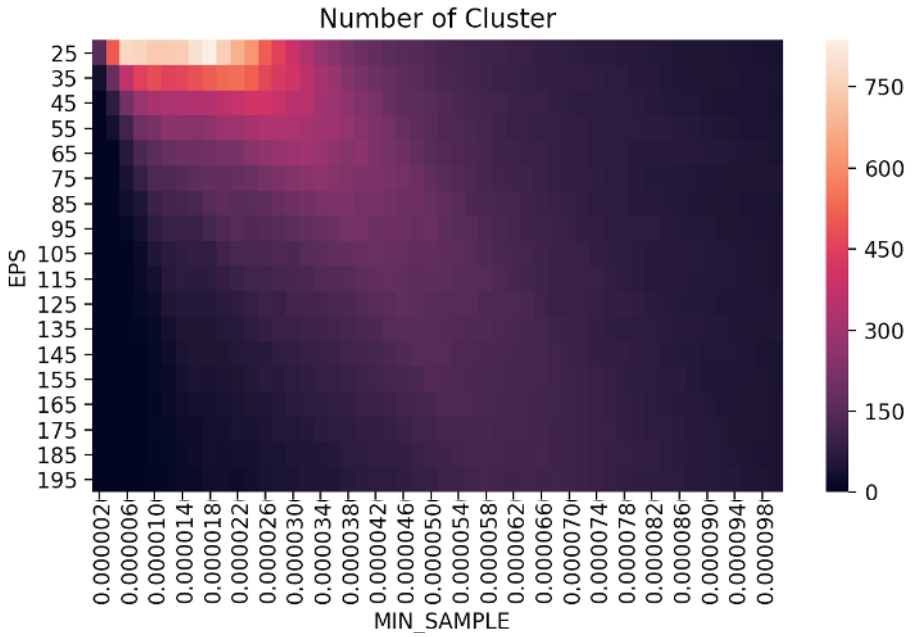


Figura 4.12: Numero di Cluster più numeroso in un insieme di Clustering a Bassa risoluzione

Formule per gli indicatori

In questo contesto utilizzeremo il termine indicatore come sinonimo di KPI. Un indicatore $I_x \in Indicators$ è una funzione $I_x(Clustering, I_l, ...I_g) = v$ che può riceve in input o lo specifico insieme dei cluster $Clustering = \{c_0, ...c_y\}$ oppure uno o più indicatori $< I_l, ...I_g >$ già calcolati e misura un valore che caratterizza lo specifico clustering per permettere il confronto e la scelta del clustering più adeguato.

Esempi di indicatori Di seguito sono elencati alcuni indicatori ideati sulla base delle esigenze emerse durante l'analisi dei cluster.

- Numero di cluster: $I_0 = |Clustering| - 1$
- Numerosità di cluster: $I_1 = \frac{|c_1|+|c_1|+\cdots+|c_y|}{|c_1|+|c_1|+\cdots+|c_y|+|c_0|}$
- Media della numerosità dei cluster: $I_2 = \frac{|c_1|+|c_1|+...+|c_y|}{|Clustering|-1}$
- Numerosità Massima: $I_3 = max(|c_1|, |c_2|, ..., |c_y|)$
- Numerosità Minima: $I_4 = min(|c_1|, |c_2|, ..., |c_y|)$
- Deviazione standard della numerosità : $I_5 = std(|c_1|, |c_2|, ..., |c_y|)$

- Numerosità dei primi 3 grandi cluster: $I_6 = |c_h0, c_h1, c_h2|$ dove c_h0, c_h1, c_h2 sono i 3 cluster più numerosi;
- Numero di cluster al di sotto della "Media della numerosità dei cluster": $I_6 = |c_{under\_avg}|,\ c_{under\_avg} = \{c_k\ t.c\ |c_k| < I_2\}$

Oltre a questi semplici esempi se ne potrebbero ideare molti altri basati anche su caratteristiche più complesse come ad esempio la distanza media tra i punti mediani di ogni cluster.

Essendo gli indicatori, in questo caso, valori numerici si prestano ad essere rappresentati graficamente così da permettere la valutazione dei Clustering da un punto di vista diverso da quello classico del raggruppamento dei punti. Ad esempio una rappresentazione grafica del numero di place nel cluster più numeroso è rappresentato dalla figura 4.11, in questo caso attraverso un insieme dei clustering a bassa granularità.

Un ulteriore esempio di rappresentazione grafica di un indicatore è raffigurato nella figura 4.12 che raffigura il numero di cluster identificati nel clustering attraverso un insieme dei clustering a più alta granularità rispetto alla figura 4.11.

4.3.2 Metodi di selezione

A partire dall'insieme dei clustering e dei relativi indicatori che formano un insieme delle soluzioni è possibile scegliere il Clustering migliore rispetto ad un Clustering obiettivo del quale, però, non conosciamo i parametri di configurazione del DBSCAN $Eps, MinPts$ che lo genera.

Il modo per scegliere il Clustering desiderato è definire:

- un Clustering obiettivo composto da valori obiettivo degli Indicatori $s_{goal} = \{I_{goal_0}, I_{goal_1}, ..., I_{goal_k}\}$
- un funzione di scelta sui valori obiettivo $Select_q(S, s_{goal}) = s_{target} = s_{i,j}$ che applica un certo metodo di selezione $Select_q()$ all'insieme $S_{D,Eps,MinPts}$ rispetto a certi valori obiettivo s_{goal}, selezionando un particolare Clustering $s_{i,j}$ dall'array $S_{D,Eps,MinPts}$.

In questo lavoro sono state sperimentate diverse funzioni di scelta che realizzano la selezione del Clustering per mezzo diverse strategie, in particolare si sono sperimentate

- selezione basata su distanza;
- selezione basata su query;
- selezione ibrida distanza-query.

Nel presente lavoro, seppur siano state sperimentate tutte e tre le tipologie di selezione, ci si è concentrati sui metodi basati su query confrontando quattro principali configurazioni e quindi scegliendone in fine una sola.

Selezione basata su distanza

Una selezione basata su distanze si basa sul formare vettori $s_{goal} = \{I_0, ..., I_g\}$ e selezionare la soluzione $s_{i,j} = \{I_0, ..., I_g\}$ tale che $distance(s_{goal}, s_{i,j})$ è minima, come nell'esempio della figura 4.13.

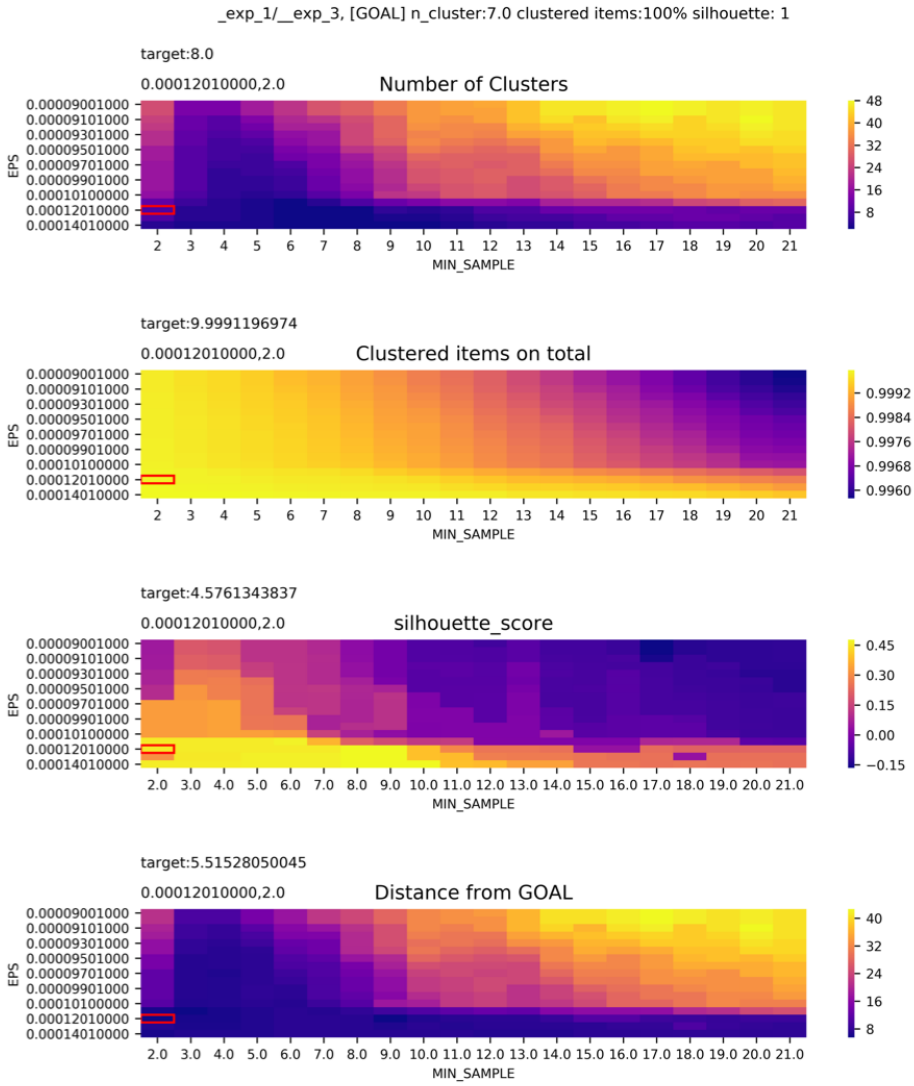


Figura 4.13: Insieme dei Clustering e selezione della Soluzione Migliore

l vantaggio principale di una tecnica di selezione basata sulla distanza è la semplice implementazione ed automazione, infatti molte distanze

sono già disponibili in moltissimi linguaggi di programmazione e sono applicabili a vettori di varia natura.

Mentre tra gli svantaggi principali troviamo la difficoltà nella scelta della formula di distanza più adeguata oppure la difficoltà di definizione del giusto peso che i singoli valori dell'array. Infatti nella sperimentazione si è notato che alcuni indicatori avevano troppa influenza nella calcolo della distanza tanto da vanificare l'utilizzo della stessa. Un metodo per mitigare questo effetto di influenza è la normalizzazione dei i valori degli indicatori.

Selezione basata su query

Un modo per mitigare l'influenza eccessiva di alcuni indicatori è quello di utilizzare una tecnica di selezione basata su query delle soluzioni utilizzando gli indicatori come parametri. Tale tecnica è stata quella utilizzata in questa ricerca e verrà discussa in dettaglio nelle sezioni successive.

Il vantaggio di una tecnica basa su query è quello di eliminare i casi estremi e selezionare con maggiore precisione la soluzione che ci interessa. Lo svantaggio, invece, risiede nel fatto che utilizzando una query troppo specifica si rischia di non avere alcuna soluzione target, se invece si utilizza una query troppo vaga il problema è l'opposto, ci si ritrova ad avere troppe soluzioni target.

Query 1 Tale query seleziona il Clustering dove, ordinando i Cluster per numerosità; ignorando il "più numeroso"; considerando la deviazione standard dei successivi 6 cluster; sceglie il Clustering con la Deviazione standard più bassa (vedi figura 4.14).

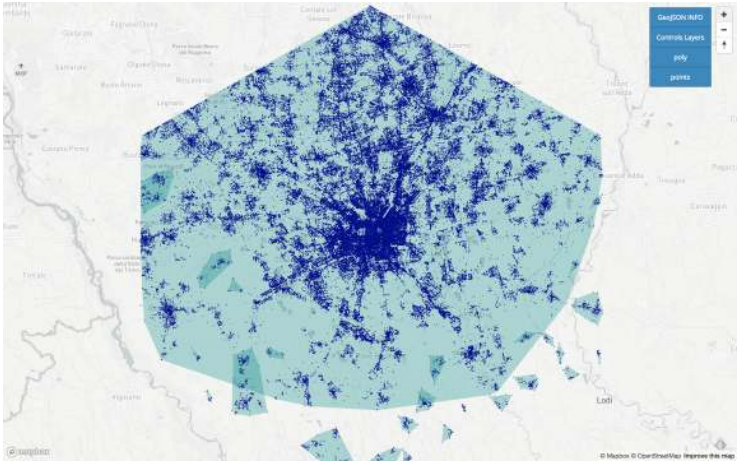


Figura 4.14: Output del Clustering selezionato attraverso la query n.1

Query 2 Tale query è simile alla precedente, ma opera in maniera opposta rispetto alla deviazione standard, infatti sceglie il Clustering ordinando i Cluster per numerosità; ignorando il "più numeroso"; considerando la deviazione standard dei successivi 6 cluster; sceglie il Clustering con la Deviazione standard più alta.

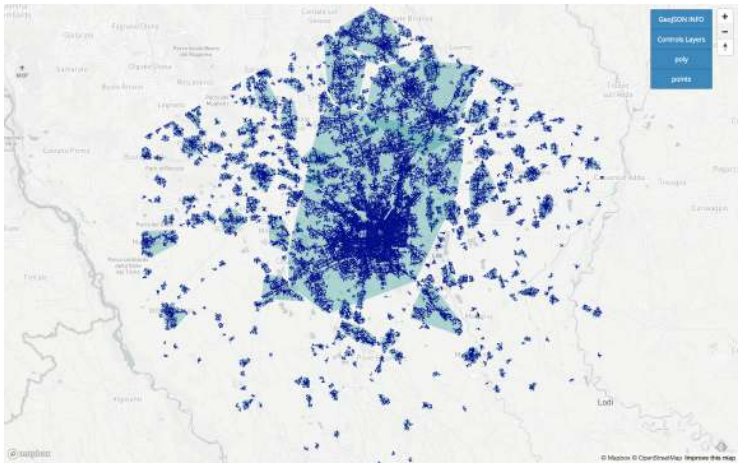


Figura 4.15: Output del Clustering selezionato attraverso la query n.2

Query 3 Tale query sceglie il Clustering ordinando i Cluster per numerosità; ignorando il più numeroso; considerando la somma della numerosità dei successivi 6 più numerosi; sceglie il Clustering con tale somma maggiore e che siano stati utilizzati almeno 87% dei punti in totale.

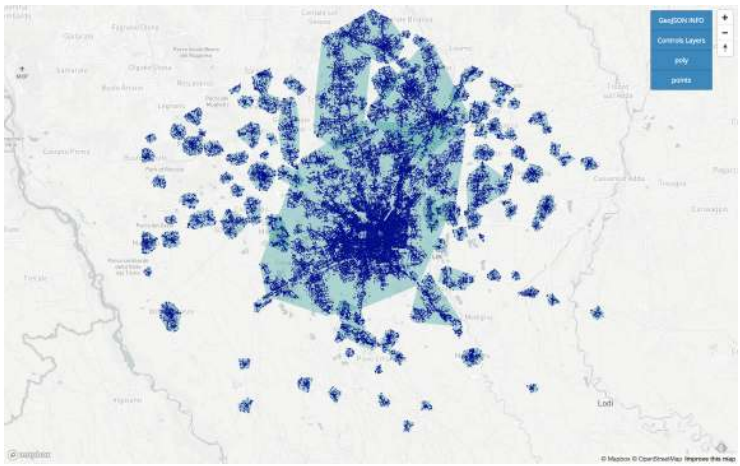


Figura 4.16: Output del Clustering selezionato attraverso la query n.3

Tale query è quella scelta per essere utilizzata in tutto il processo di Clustering gerarchico-iterativo appositamente realizzato allo scopo di

scegliere quel Clustering con una massa critica di Cluster considerevole, senza però essere troppo influenzati dalle aree centrali della città che spesso sono caratterizzate da picchi di densità eccessivi tali da influenzarne l'analisi (vedi sezione sui picchi di densità 2.6.3).

In oltre tale metodo di selezione, da un punto di vista visuale, sembra rappresentare approssimativamente la percezione delle aree urbane e sub-urbane di Milano e del vicino interland.

Tale metodo di selezione è stato applicato in modo identico a tutti i livelli del processo iterativo, ma nulla toglie che a livelli diversi dell'iterazione, o al presentarsi condizioni particolari, si possa modificare il metodo di selezione rendendolo più adeguato al tipo di Soluzioni e quindi ai Clustering a disposizione.

Query 4 Questa query predilige il Clustering con il numero di punti che fanno parte di almeno un cluster, più alto: scegliendo il Clustering con la somma della numerosità dei Clustering più alta.

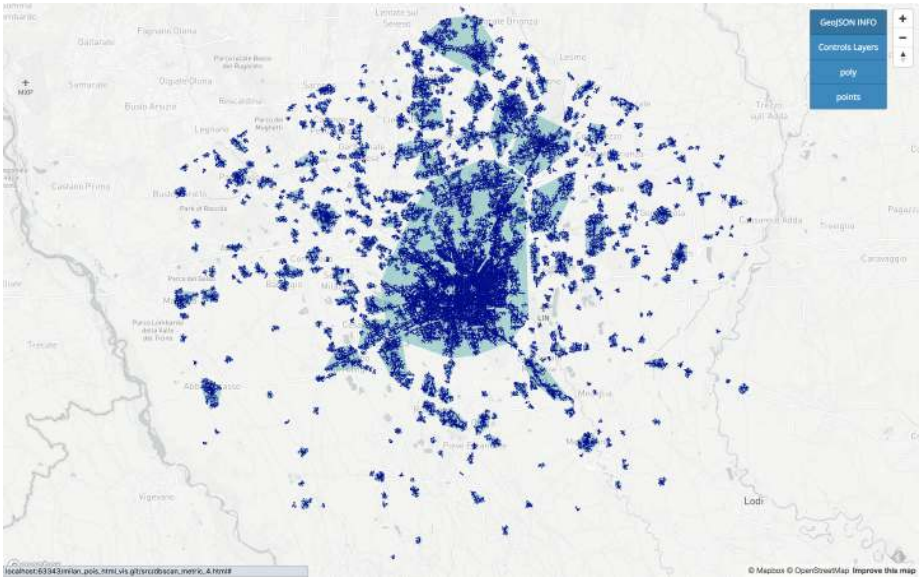


Figura 4.17: Output del Clustering selezionato attraverso la query n.4

Selezione ibrida query-distanza

Un ulteriore metodo non testato, ma realizzabile, è utilizzare un metodo che sfrutti contemporaneamente metodi basati su query e metodi basati sulla distanza ad esempio, utilizzando in una prima fase una query che elimina i casi estremi ed in un secondo momento applicare una funzione di distanza che selezionare in modo più preciso la soluzione target.

4.4 L'algoritmo

In questa sezione verrà descritto nel dettaglio l'algoritmo che realizza il Clustering gerarchico-iterativo implementando praticamente i concetti descritti nella sezione precedente.

4.4.1 Panoramica del processo e delle componenti

L'algoritmo di Clustering, pur mantenendo le stesse componenti, può essere analizzato da due punti di vista (vedi figura 4.18):

- come processo di clustering lineare: ovvero come processo in grado di selezionare una specifica suddivisione dello spazio da un insieme di Clustering ottenuti da uno stesso dataset;
- come processo iterativo di clustering: ovvero come processo in grado di selezionare iterativamente una specifica suddivisione dello spazio su diversi dataset gerarchicamente correlati, fino a costruire un Clustering gerarchico del dataset iniziale.

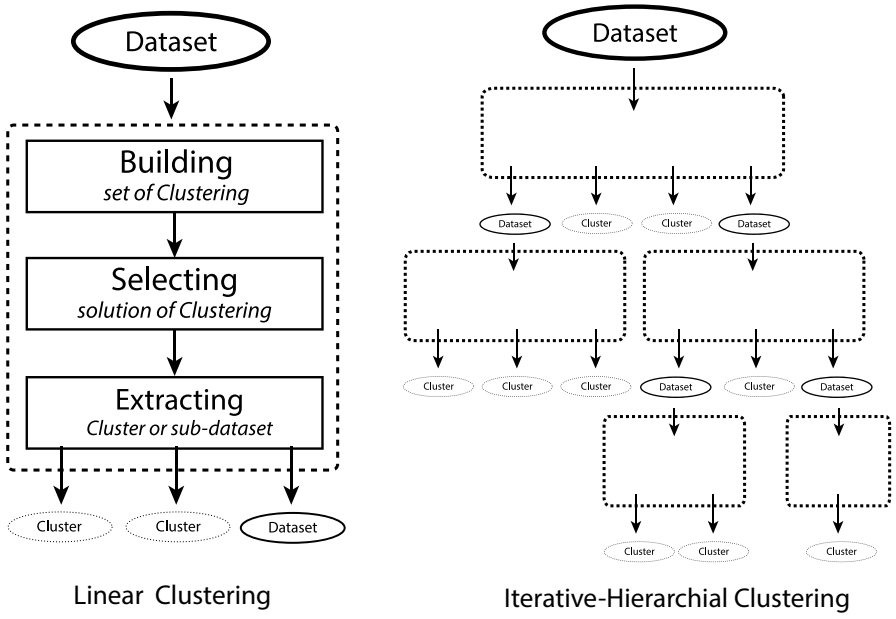


Figura 4.18: Confronto tra processo lineare di Clustering e processo iterativo-gerarchico di Clustering

Il Processo iterativo di clustering non è altro che l'esecuzione iterativa del Processo di clustering lineare basato sul DBSCAN, dove le componenti sono state utilizzate in modo da realizzare l'iterazione. Per chiarezza di esposizione, nelle sezioni successive, verrà prima descritto da un punto di vista lineare ed in seguito dal punto di vista iterativo.

Processo lineare

Il processo di Clustering da un punto di vista lineare trasforma un dataset di punti geolocalizzati in Cluster, selezionando il Clustering rispetto ad un insieme di parametri di configurazione. I Cluster estratti sono quindi etichettati come Cluster-indivisibili oppure come Sotto-dataset (vedi figura 4.19).

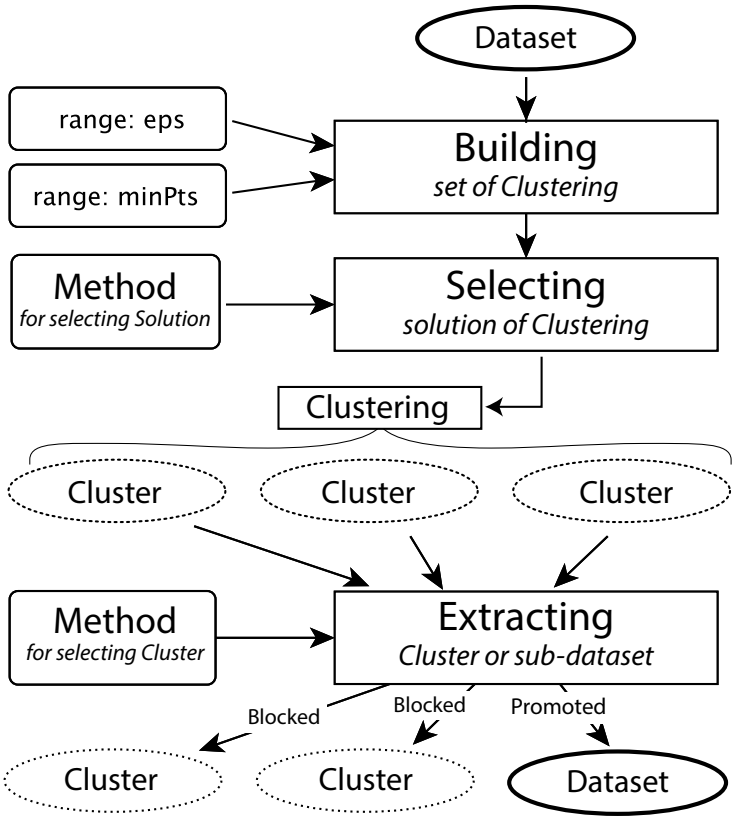


Figura 4.19: Panoramica lineare del processo di Clustering che a partire da un dataset di Place produce uno specifico Clustring scelto da una molteplicità di possibili Soluzioni

Come accennato nella sezione 4.2.2, tale processo, è suddiviso in tre componenti:

- Building:
  - $BuildingSolutions(EPS_n, MinPts_m, Dataset) = S_{n,m}$
  - componente che accetta in input un dataset di Place e due insiemi di valori EPS ed $MinPts$, eseguendo sequenzialmente $n \times m$ volte un Processo di clustering DBSCAN; restituendo in output un Insieme dei clustering corredate di Indicatori, realizzando l'insieme delle Soluzioni $S_{n,m}$;

- Selecting:
  - $SelectSolution(S_{n,m}, Method_{clustering}()) = s_{ij}$
  - componente che applica un metodo di selezione allo scopo di scegliere la Soluzione di Clustering migliore rispetto ad un Clustering obiettivo realizzato attraverso query su indicatori; restituendo in output uno specifico Clustering del dataset;
- Extracting:
  - $ExtractDataset(s_{ij}, Method_{dateset}()) = \{Clusters, Datasets\}$
  - componente che accetta in input un Clustering specifico e lo analizza allo scopo di etichettare i Cluster come sotto-dataset oppure congelarli come cluster-indivisibili; i Cluster etichettati come sotto-dataset;

Dunque, da un punto di vista lineare, il processo di Clustering accoglie in input un dataset di punti geolocalizzati ed in base ad una serie di parametri ne estrae una specifica suddivisione in Cluster che rispecchia la migliore soluzione rispetto alle aspettative formalizzate in una query.

Processo iterativo

L'idea del processo iterativo di Clustering si basa sul fatto che ogni Cluster $c_x \in Clustering = \{c_0, ..., c_y\}$ ottenuto dal processo lineare di Clustering è dello stesso tipo del dataset iniziale, ovvero composto da Place, per questo riutilizzabile come input per un nuovo Processo di clustering.

Per rendere il processo iterativo e realizzare l'albero di Cluster annidati, è stato necessario isolare le tre componenti precedentemente descritte e gestirle attraverso un Work Dispatcher ed una coda FIFO.

Il Work Dispatcher è la componente software che si occupa dell'organizzazione e della distribuzione dei compiti rispetto ai risultati ottenuti dalle componenti di Building, Selecting ed Extracting realizzando il Processo iterativo di clustering che costruisce l'albero dei Cluster a densità diversa (vedi figura 4.20).

Un Compito è un oggetto di tipo `dizionario`, caratterizzato da un campo `Type` che indica il tipo di attività da eseguire, ed un campo `Data` che contiene le variabili per eseguire tale attività come ad esempio la path per recuperare i dataset oppure i parametri per selezionare un preciso Clustering dal database.

Il campo `Type` può assumere 3 valori, ed indica cosa contiene il campo `data` e di consequenza a chi dovrà essere consegnato per essere utilizzato. I valori che può assumere `Type` sono:

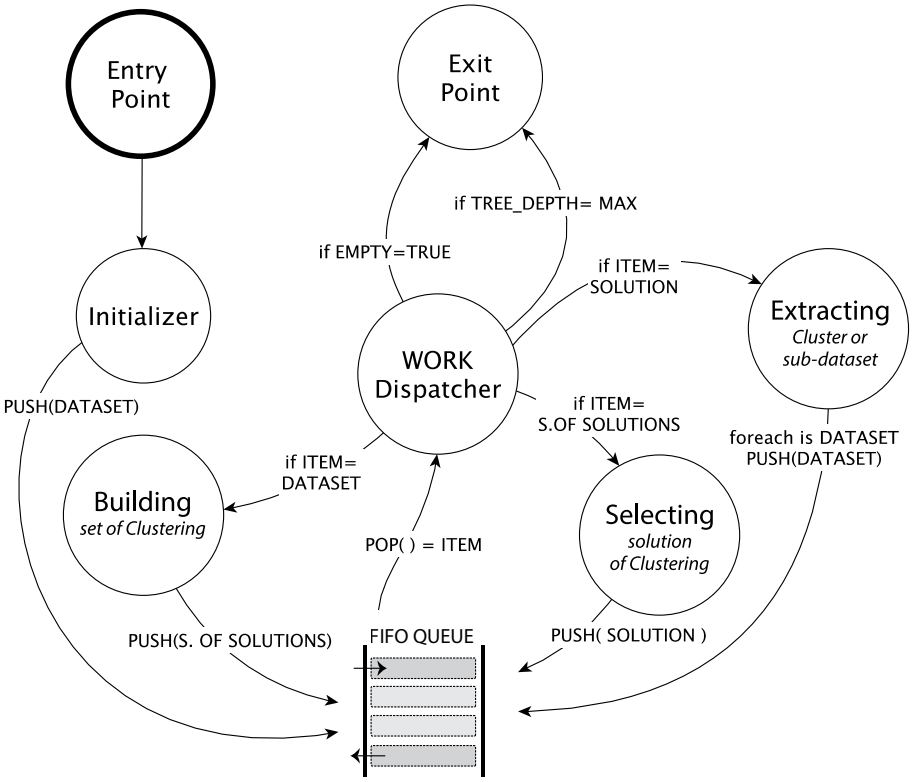


Figura 4.20: Processo iterativo di Clustering realizzato attraverso un Work Dispatcher ed una coda FIFO di compiti

- `DATASET`
- `SPACE OF SOLUTION`
- `SOLUTION`

DATASET Indica che deve essere costruito uno Insieme dei clustering a partire da un determinato dataset composto da un file indicato da una path, dove il nome del file è l'`id` del Cluster che lo ha generato, per convenzione il Cluster 0 è il dataset iniziale. Viene quindi richiamata la componente Building.

```
type: 'DATASET',
data: {
  filepath: del tipo '0/2.csv'
}
```

SPACE OF SOLUTIONS Indica che deve essere selezionata una Soluzione a partire da un Insieme dei clustering relativo ad un

particolare dataset identificato dal campo `id` e `filepath`. Viene quindi richiamata la componente Selecting.

```
type: 'SPACE OF SOLUTIONS',
data: {
  filepath: '0/2.csv',
  id: '0/2'
}
```

SOLUTION Indica che devono essere estratti dei sotto-dataset a partire da uno specifico Clustering di un determinato dataset identificato da un `id`, uno specifico valore di `eps`, uno specifico valore di `minPts` ed un file identificato da `solutionpath`, quest'ultimo contenente tutte le statistiche e gli indicatori dello specifico Clustering selezionato. Viene quindi richiamata la componente Extracting.

```
type: 'SOLUTION',
data: {
  filepath: '0/2.csv',
  id: '0/2',
  eps: '0.0035,'
  minPts: '12'
  solutionpath: '0/2/solution.csv'
}
```

Scenario iniziale del processo di Clustering Iterativo

Il processo ha inizio a partire da un dataset iniziale che per convenzione è identificato con l'`id=0` nel file `0.csv` che viene inserito nella coda attraverso un Compito iniziale del `'DATASET'`.

```
type: 'DATASET',
data: { filepath: '0.csv' }
```

Tale Compito viene quindi prelevato dal Work Dispatcher che istruisce il metodo atto alla costruzione dell'Insieme dei clustering (la componente Building) indicando la posizione del dataset da utilizzare. Tale componente eseguirà sequenzialmente diversi Processi di clustering allocando i Clustering e i relativi Indicatori nel database.

Al termine del processo, tale componente, creerà un Compito del tipo `'SPACE OF SOLUTIONS'`, nel quale è indicato quale Insieme dei clustering dovrà essere analizzato per selezionare la Soluzione.

```
type: 'SPACE OF SOLUTIONS',
data: {
```

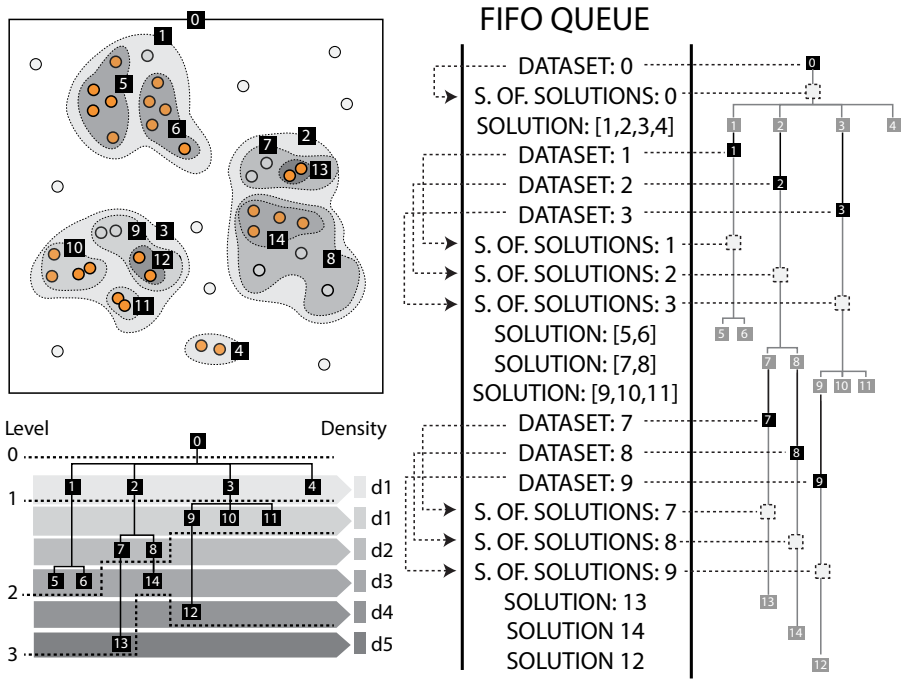


Figura 4.21: Processo iterativo di Clustering attraverso l'utilizzo di una coda di compiti

```
    filepath: '0.csv',
    id: '0'
  }
```

A questo punto il controllo è passato nuovamente al Work Dispatcher che preleverà il nuovo Compito ed istruirà il processo per la selezione della Soluzione il quale, attraverso l'*id*, sarà in grado di comprendere quali Clustering analizzare nel database.

La componente di Selecting sceglierà una specifica Soluzione di Clustering dal database che allocherà nel file system in una posizione prefissata e realizzerà un nuovo Compito del tipo `'SOLUTION'` in cui sarà indicata la posizione ed i parametri della Soluzione di Clustering scelta.

```
type: 'SOLUTION',
data: {
  filepath: '0.csv',
  id: '0',
  eps: '0.0035,'
  minPts: '12'
  solutiopath: '0/solution.csv'
}
```

Infine il Work Dispatcher preleverà il Compito di poco sopra ed istruirà il processo di estrazione dei dataset (la componente Extracting) che analizzando la Soluzione di Clustering indicata nel Compito sceglierà quali dei Cluster contenuti diventeranno nuovi Sotto-dataset quali invece saranno bloccati a Cluster-indivisibili (vedi figura 4.21).

Inserendo nella coda tanti Compiti di tipo `'DATASET'` tanti quanti sono i Cluster considerati ulteriormente divisibili indicando al suo interno il file `x.csv` in cui sono stati allocati i punti estratti, appunto, dai Cluster della specifica suddivisione estratta.

```
type: 'DATASET',
data: { filepath: 'x.csv' }
```

Il processo continuerà di questo passo finché saranno programmati dataset da analizzare.

4.4.2 Costruzione dell'insieme dei Clustering

L'Insieme dei clustering è un set di riorganizzazioni del dataset diversi, mentre l'insieme delle Soluzioni è un Insieme dei clustering correlati di Indicatori. Il numero di Soluzioni $s_{i,j}$ è pari alla cardinalità degli insiemi EPS moltiplicata per la cardinalità di $MinPts$, ovvero $S_{D,Eps,MinPts} = \{s_0, ..., s_p\}$, con $p = |Eps| \cdot |MinPts|$, infatti $S_{D,Eps,MinPts} = \{s_{1,1}, ..., s_{n,m}\}$ con $|Eps| = n$ e $|MinPts| = m$.

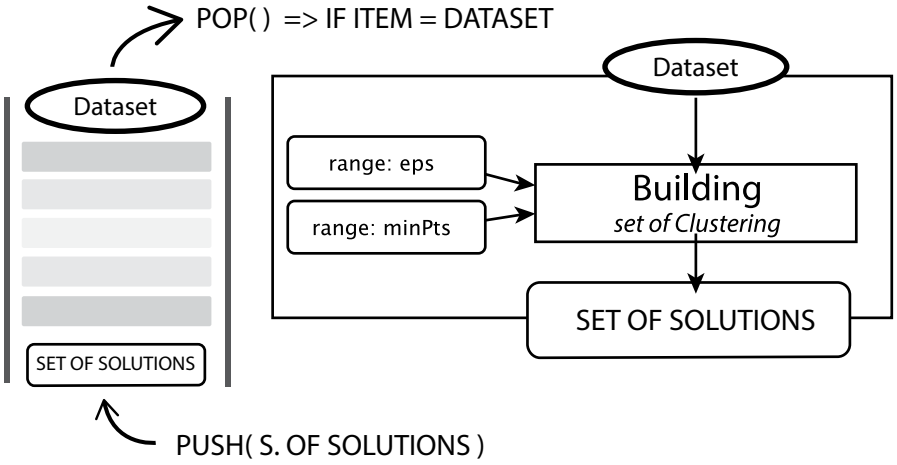


Figura 4.22: Processo di Costruzione dell'Insieme delle Soluzioni a partire da un dataset ed un range di valori di configurazione prefissato

Input Il dataset di input è un insieme di Place di coordinate $Place_i = (lat, lon)$ in formato `.csv`, del tipo:

```
45.46447571, 9.17169092
45.46867718,  9.17591398
45.45180727,  9.19868096
45.46224982, 9.19493479
45.47498121,  9.19384371
...
```

Gli insiemi $Eps = \{\epsilon_0, ..., \epsilon_n\}$ e $MinPts = \{\omega_0, ..., \omega_m\}$ sono inglobati nel codice della componente attraverso due specifici metodi che attraverso alcuni parametri di configurazione permettono di variare la *granularità* e *ampiezza* dei valori.

Output Per ogni coppia di valori $eps, minPts$ viene eseguito un particolare Processo di clustering DBSCAN, per ognuno dei Clustering calcolati vengono poi calcolati i relativi Indicatori che compongono, insieme ai Clustering, le varie soluzioni (vedi sezione 4.3.1), il tutto è allocato in memoria attraverso un database MongoDB.

Le $n \times m$ Soluzioni sono allocate in un cosiddetto Document di MongoDB [2] che si compone di due gruppi di parametri principali:

- `index`: parametri che servono a reperire le diverse soluzioni ottenute;
- `data`: parametro che contiene i Clustering e i relativi Indicatori.

Per motivi legati a problemi computazionali del processo, nell'oggetto `data` è stato aggiunto un parametro denominato `state` = *string* che indica lo stato di esecuzione del processo e può assumere quattro valori:

- `NONE`: corrisponde ad un Clustering non ancora eseguito;
- `OUT-OF-RANGE`: stato di errore che corrisponde ad un Clustering non utilizzabile;
- `OUT-OF-MEMORY`: stato di errore che corrisponde ad un Clustering che ha impiegato una quantità eccessiva di memoria;
- `OK`: corrisponde ad un Clustering eseguito correttamente;

Qualità e computabilità delle Soluzioni

Una criticità del progetto è rappresentata dal fatto che non tutti i dataset in relazione a determinate combinazioni di parametri $eps, minPts$ producono buone Soluzioni (o Clustering computabili), tali Clustering *problematici* possono essere di due tipi:

[2] `https://docs.mongodb.com/manual/core/document/`

- `OUT-OF-RANGE`: rappresentano quei Clustering che hanno indicatori che palesemente li identificano come non utili ad analisi future;

- `OUT-OF-MEMORY`: rappresentano quei Clustering che impiegano una quantità eccessiva di memoria tale da creare problemi di stabilità dell'intero processo;

Entrambe le situazioni hanno effetti negativi sull'esecuzione del Processo di clustering, motivo per cui si è realizzato un sistema per evitare di eseguire un numero eccessivo di processi di questo tipo.

Clustering OUT-OF-RANGE Rappresentano processi di Clustering che hanno prodotto in numero eccessivamente alto di Cluster, oppure eccessivamente basso, tale da renderli non utilizzabili. Tale situazione suggerisce in oltre che il dataset non è trattabile con un certo tipo di parametrizzazione del Processo di clustering.

Tendenzialmente, da quanto osservato, dopo un certo numero di Clustering di questo tipo compaiono anche quelli con un consumo di memoria eccessivo, ovvero quelli segnati come `OUT-OF-MEMORY`.

Clustering OUT-OF-MEMORY Essi sono Processi di clustering che consumano un eccessiva quantità di memoria tale da bloccare completamente il sistema. Spesso giungono dopo un certo numero di soluzioni `OUT-OF-RANGE`, ma non è detto che sia sempre così, dunque è stato realizzato un sistema che cerca di prevedere quali combinazioni di $eps, minPts$ produrranno situazioni analoghe (vedi figura 4.23).

Prevedere le soluzioni che danneggiano il processo di Clustering

Sia che un esecuzione consumi una quantità eccessiva di memoria, sia che il Clustering realizzato non possiede le caratteristiche minime di qualità, si è osservato empiricamente che tali situazioni si presentano in maniera graduale e costante al crescere dei valori eps e $minPts$, spostandosi verticalmente ed orizzontalmente nella matrice (vedi figura 4.23).

Per evitare di eseguire inutilmente processi con specifici parametri $eps, minPts$ dei quali ci aspettiamo già in anticipo che producono Clustering non utilizzabili o dannosi, sono stati previsti due controlli, uno longitudinale ed uno verticale, che al presentarsi di un numero minimo consecutivo di Clustering `OUT-OF-RANGE` oppure `OUT-OF-MEMORY` eviterà di processare le successive combinazioni di valori di input previste.

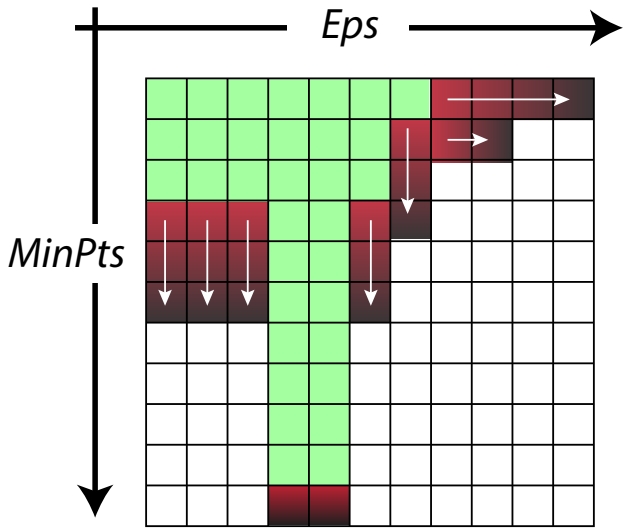


Figura 4.23: Sistema di previsione delle esecuzioni non utili o dannose per l'esecuzione del Processo di Clustering

Complessità computazionale Il numero di esecuzioni dei Processi di clustering DBSCAN è al massimo pari al numero degli elementi dell'Insieme dei clustering $S_{D,EPS,MinPts} = ...$ ovvero $n \times m$ nel caso siano tutte trattabili, mentre in caso contrario quindi in presenza di processi segnati come `OUT-OF-RANGE` oppure `OUT-OF-MEMORY`, il numero di esecuzioni sarà inferiore.

Sistema di controllo della computabilità del Clustering

Il sistema appena descritto è realizzato attraverso l'uso processi paralleli ed incapsulati che permettono di monitorare in tempo reale i Processi di clustering e gestire in modo ottimale la costruzione dell'Insieme dei clustering. Per motivi di prestazione della macchina su cui è stato realizzato il Processo di clustering iterativo è stato scelto di procedere con un Processo di clustering per volta, ma l'architettura permette di eseguire più istanze contemporanee.

Il componente Building si compone di un processo principale che a sua volta esegue :

- 1 processo di controllo: Clustering Monitor;
- n Processi di clustering sequenziali (o paralleli): Clustering Wrapper;
  - che a loro volta eseguono 1 Processo di clustering DBSCAN di `scikit-learn`;

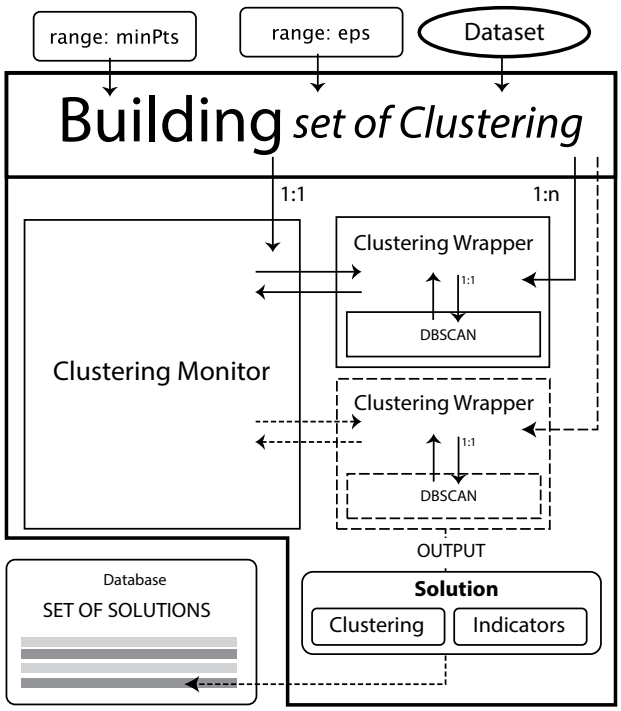


Figura 4.24: Sistema di controllo dei processi di Clustering per prevenire situazioni di memory overflow

Il processo principale Provvede ad inizializzare 1 sotto-processo Clustering Monitor ed n sotto-processi Clustering Wrapper, tanti quante sono le combinazioni di $eps, minPts$. Il primo conosce sempre quali e quanti sotto-processi Clustering Wrapper sono attivi in un dato istante attraverso una lista di puntatori ai processi.

Al termine di ogni sotto-processo Clustering Wrapper, il principale provvede a recuperarne la Soluzione ed allocarla nel database, oppure nel caso in cui il Clustering Wrapper è terminato in modo anomalo provvederà ad allocare le informazioni dell'anomalia (vedi figura 4.24).

Il processo principale continuerà ad eseguire istanze di Clustering Wrapper fino al termine di tutte le combinazioni $eps, minPts$ previste.

Il sotto-processo Clustering Monitor Questo sotto-processo viene eseguito dal processo principale e monitora lo stato dei Clustering Wrapper in esecuzione. Esso continuerà la sua esecuzione finché esistono istanze di Clustering Wrapper da monitorare. Il suo compito è quello di controllare a intervalli di pochi centesimi di secondo quali e quanti sotto-processi Clustering Wrapper sono in esecuzione, monitorandone l'andamento, il loro consumo di memoria e gli eventuali risultati (vedi figura 4.24).

Nel caso fosse identificato un consumo di memoria eccessivo o un qualsiasi altro errore di esecuzione, esso comunicherà tale situazione al processo principale che provvederà a chiudere il processo in questione dando spazio ad un nuovo Processo di clustering.

Il sotto-processo Clustering Wrapper Ha il compito primario di incapsulare il Processo di clustering DBSCAN vero e proprio e renderlo *innocuo* nel caso dovesse avere problemi di esecuzione o consumo di memoria.

Mentre il compito secondario, ma non meno importante, è quello di acquisire il risultato di una particolare esecuzione del Processo di clustering DBSCAN, valutandone l'output per calcolarne gli Indicatori (KPI) in tempo reale. Organizzando, così, Clustering ed Indicatori in oggetti detti Soluzioni che comunicherà al processo principale che a sua volta si occuperà di allocarli in memoria (vedi figura 4.24).

4.4.3 Selezione del Clustering

La scelta sella Soluzione di Clustering migliore, può avere luogo solo se è stato generato correttamente l'Insieme dei clustering (figura 4.25) ed è attuato dal processo di selezione (Selecting).

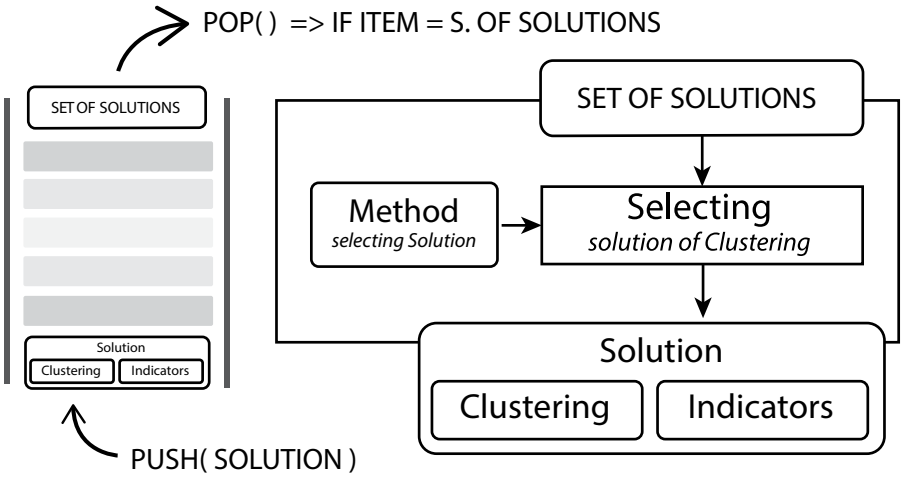


Figura 4.25: Componente per la selezione della Soluzione migliore rispetto all'Insieme dei Clustering

Esso accetta in input l'`id` di un dataset e recupera tutte le Soluzioni correttamente eseguite presenti nel database (ovvero etichettate con `state="ok"`), su queste applica un metodo di selezione per decidere quale tra di esse si avvicina maggiormente alla Soluzione obiettivo.

4.4.4 Estrazione dei dataset

L'output del processo di selezione produce un particolare Clustering composto da una precisa riorganizzazione del dataset, Soluzione che si avvicina maggiormente alla Soluzione obiettivo formalizzata in un metodo di selezione descritto nella sezione 4.4.3.

La componete di estrazione dei dataset ha il compito di etichettare i Cluster contenuti nel Clustering come Sotto-dataset riutilizzabili oppure come Cluster-indivisibili (figura 4.26).

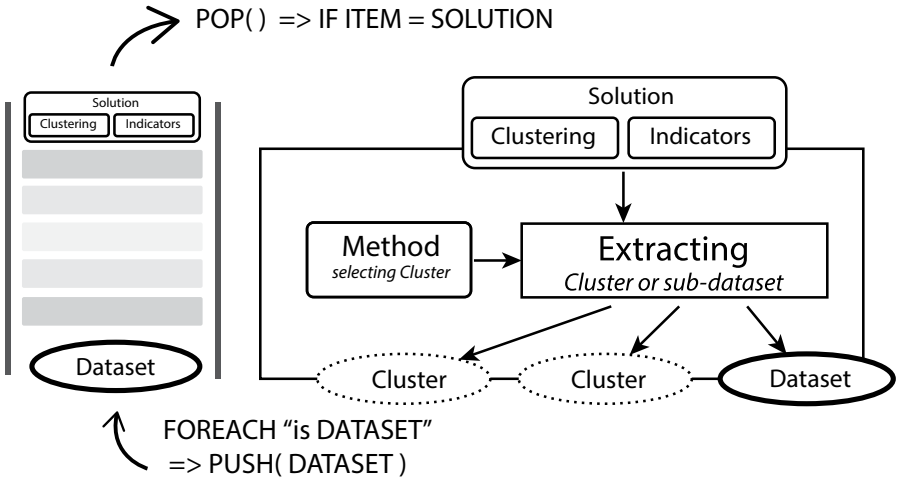


Figura 4.26: Componente per l'etichettatura degli item del Clustering come Cluster indivisibili oppure sotto-dataset

Etichettatura come Cluster indivisibile o sotto-dataset

Allo stato attuale del progetto la discriminante che permette di decidere quale dei Cluster presenti nella suddivisione della soluzione di Clustering selezionata è la numerosità dei place di ogni Cluster, attualmente imposto a 250 Place, ma tale scelta può essere ulteriormente articolata in modo analogo alla selezione della soluzione in base ad indicatori diversi relativi ai singoli Cluster.

Schedulazione dei compiti e estrazione dei dataset Per ogni Cluster etichettato come Sotto-dataset viene allocato un file `.csv` (vedi figura 4.27) composto da una lista dei Place di coordinate (lat, lon). Tale dataset viene aggiunto alla coda dei Compiti dal Work Dispatcher attraverso un oggetto precedentemente descritto. Mentre i Cluster etichettati come indivisibili vengono congelati per una successiva visualizzazione.

0	18.csv	28	37.csv	47	56.csv
0.csv	19	28.csv	38	47.csv	57
1	19.csv	29	38.csv	48	57.csv
1.csv	2	29.csv	39	48.csv	58
10	2.csv	3	39.csv	49	58.csv
10.csv	20	3.csv	4	49.csv	59
11	20.csv	30	4.csv	5	59.csv
11.csv	21	30.csv	40	5.csv	6
12	21.csv	31	40.csv	50	6.csv
12.csv	22	31.csv	41	50.csv	60
13	22.csv	32	41.csv	51	60.csv
13.csv	23	32.csv	42	51.csv	61
14	23.csv	33	42.csv	52	61.csv
14.csv	24	33.csv	43	52.csv	62
15	24.csv	34	43.csv	53	62.csv
15.csv	25	34.csv	44	53.csv	63
16	25.csv	35	44.csv	54	63.csv
16.csv	26	35.csv	45	54.csv	64
17	26.csv	36	45.csv	55	64.csv
17.csv	27	36.csv	46	55.csv	65
18	27.csv	37	46.csv	56	65.csv

Figura 4.27: Elenco di primo livello dei Cluster allocati come dataset in file csv

Il Declassamento da Sotto-dataset a Cluster indivisibile

Esiste un caso particolare in cui è necessario declassare un Cluster, precedentemente etichettato come sotto-dataset, a Cluster-indivisibile. Infatti non è detto che tutti i sotto-dataset sono buoni input per Processi di clustering ulteriori.

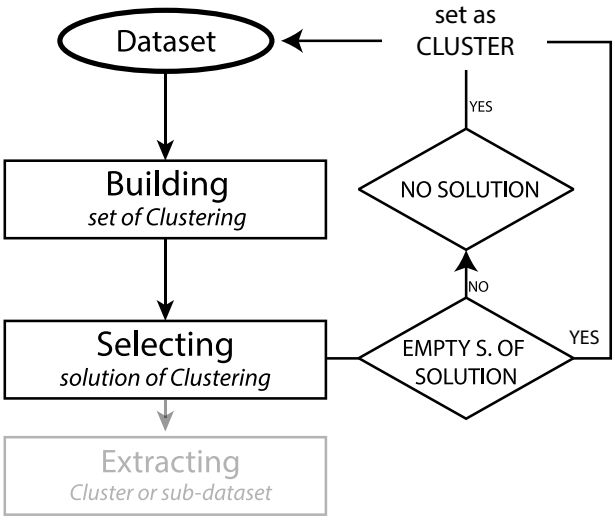


Figura 4.28: Declassamento di un Cluster precedentemente etichettato come Sotto-dataset a cluster-indivisibile effettuato nella fase di selezione della soluzione e non nella fase di estrazione dei dataset

In realtà tale caso eccezionale avviene nella fase di Selecting in cui l'algoritmo tenta estrarre la Soluzione migliore tra i Clustering prodotti. Se l'Insieme dei clustering è vuoto oppure il metodo di selezione non produce alcun risultato, allora il dataset in questione non è un vero Sotto-dataset ed è necessario declassarlo a cluster-indivisibile (vedi figura 4.28). In questi casi è come se l'algoritmo avesse preso un *abbaglio*.

4.4.5 La terminazione dell'algoritmo

Una parte essenziale del Processo di clustering è l'insieme dei metodi che ne garantiscono la terminazione, riassunti dalle seguenti funzionalità:

- il processo iterativo termina autonomamente quando non sono disponibili nuovi Compiti da effettuare nella coda gestita dal Work Dispatcher
- il declassamento dei Sotto-dataset a cluster-indivisibile permette di evitare che i dataset privi di Soluzioni valide vengano ri-schedulati nella coda;
- il sistema di controllo della sezione 4.4.2 per gestire i Clustering con problemi di efficienza e qualità permette all'algoritmo di non entrare in pericolosi loop
- infine se tutte le strategie precedenti hanno fallito è possibile impostare una profondità massima dell'algoritmo iterativo oltre la quale non verranno più estratti dataset;

Parte III

Discussione e Conclusioni

Capitolo 5

Discussione dei risultati

5.1 Un'analisi visiva dei Cluster

5.1.1 Analisi generale

A partire da un input di 240.000 place, il Processo di clustering gerarchico-iterativo, è stato estratto un albero di cluster di 5 livelli dal quale sono stati selezionati circa 300 cluster composti da almeno 25 place ciascuno. Tali cluster selezionati sono stati estratti tra quelli posizionati nelle posizioni foglia dell'albero (vedi cluster in rosso nella figura 5.1), ovvero quei cluster che, a differenza degli altri intermedi, sono stati considerati dall'algoritmo come non ulteriormente divisibili.

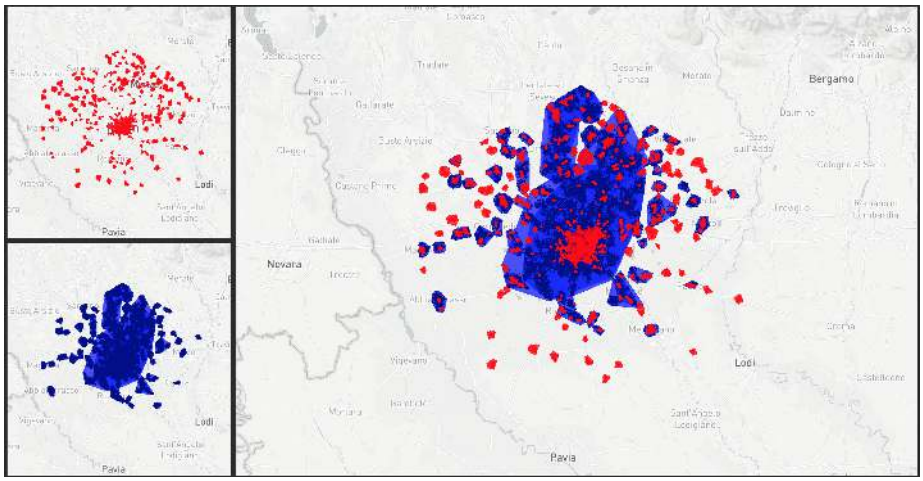


Figura 5.1: I cluster-indivisibili (di colore rosso) ed i sotto-dataset (di colore blu) che compongono l'albero di cluster prodotto dal processo di clustering gerarchico-iterativo

L'algoritmo gerarchico-iterativo ha concluso l'iterazione di

suddivisione iterativa dei cluster senza che intervenisse il limite di profondità massima imposto inizialmente, quest'ultimo inserito per evitare casi di loop, realizzando una gerarchia di cluster composta da:

- cluster foglia che non si sovrapposizioni mai tra loro;
- cluster intermedi che si sovrappongono solo con i propri padri o propri figli;
- cluster intermedi che non si sovrappongono mai con altri cluster intermedi di uno stesso livello;
- cluster foglia che hanno densità tra loro;

Quest'ultima caratteristica, relativa alla differente densità per i cluster foglia, è mostrata in figura 5.2 dove i cluster 8 e 1 visivamente appaiono vicini tra loro, ma non appartengono all'output della stessa iterazione (hanno un padri differenti).

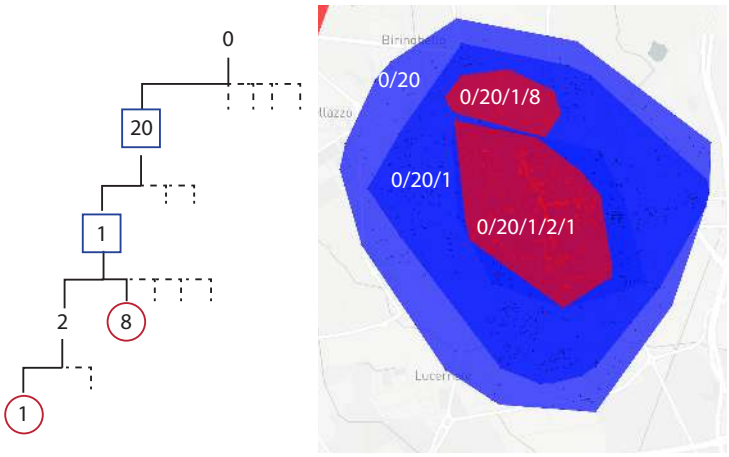


Figura 5.2: Cluster foglia vicini, ma di livelli e densità diversa

Dunque, tali cluster non sono stati estratti con la stessa combinazione di valori $Eps, MinPts$ di DBSCAN, ma sono il risultato del clustering applicato iterativamente a cluster di livelli precedenti considerati come ulteriormente divisibili.

I cluster considerati ulteriormente divisibili, in questo lavoro, sono chiamati Sotto-dataset (le forme di colore blu rappresentate nelle figure 5.2 e 5.1), non tutti tali cluster sono realmente tali, ovvero non è detto che un cluster considerato inizialmente come ulteriormente divisibile fornisca poi realmente, a seguito del processo di clustering, una buona suddivisione.

A tal proposito è stato introdotto un sistema che, inversamente dalla promozione di un cluster a Sotto-dataset, lo declassa da Sotto-dataset a cluster-indivisibile, ne è un esempio il caso raffigurato in figura 5.3.

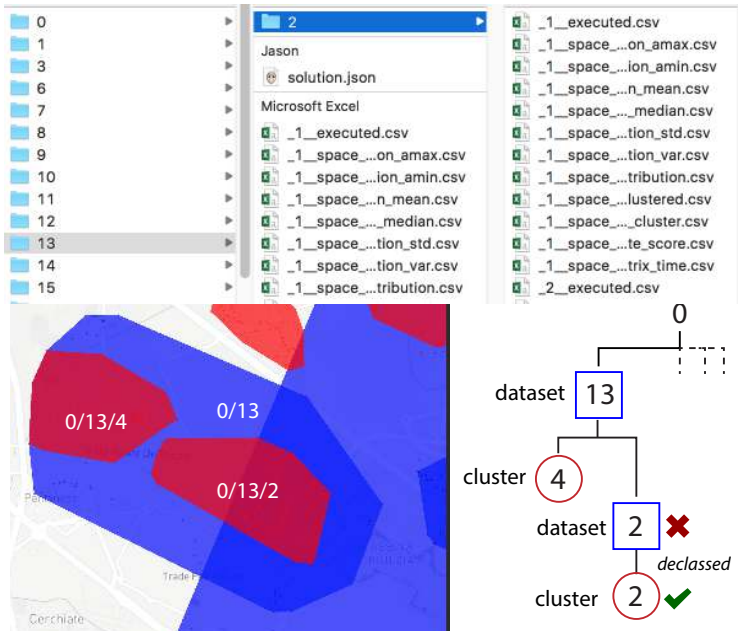


Figura 5.3: Declassamento da sotto-dataset a cluster-indivisibile

Nel caso raffigurato in figura 5.3, il cluster 2 è stato inzialmente considerato come ulteriormente divisibile e quindi promosso sotto-dataset. L'algoritmo, ha in fatti provato a costruire l'insieme dei clustering memorizzando i file con i diversi indicatori sui diversi clustering, ma a seguito della mancanza di una soluzione valida, il sotto-dataset, è stato declassato a cluster foglia.

Tale comportamento hanno da un lato permesso all'algoritmo di tornare sui propri passi e correggere una scelta fatta in precedenza, e dall'altro di continuare a cercare di suddividere un cluster che non fornisce alcuna soluzione valida rischiando di entrare in un potenziale loop.

5.1.2 Analisi per livelli

Dal livello 1 al livello 5

In figura 5.4 è visibile con maggiore chiarezza l'intera gerarchia dei Cluster che formano il Clustering gerarchico a densità diverse di Milano e del suo hinterland. In generale pare che le aree a nord abbiano richiesto maggiore lavoro rispetto a quelle a sud producendo anche più cluster.

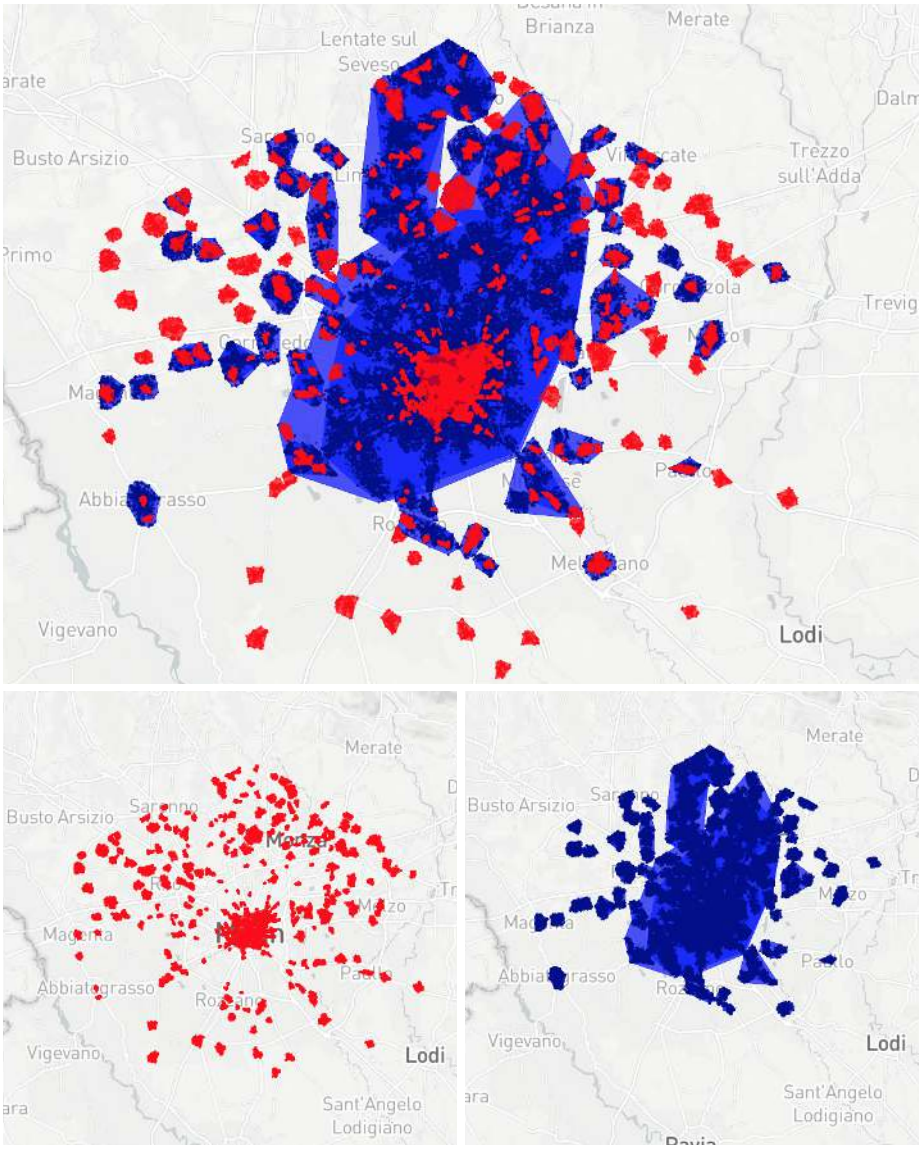


Figura 5.4: Visualizzazione di tutti e 5 i livelli della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

Livello 1

La figura 5.5 mostra il primo livello della gerarchia ottenuto dalla prima iterazione del processo attraverso il quale sono stati identificati i primi cluster-indivisibili e sotto-dataset. L'algoritmo rende evidente che i piccoli centri urbani più esterni alla città sono già identificabili con chiarezza, cosa non altrettanto valida per il le aree più interne di Milano e Monza.

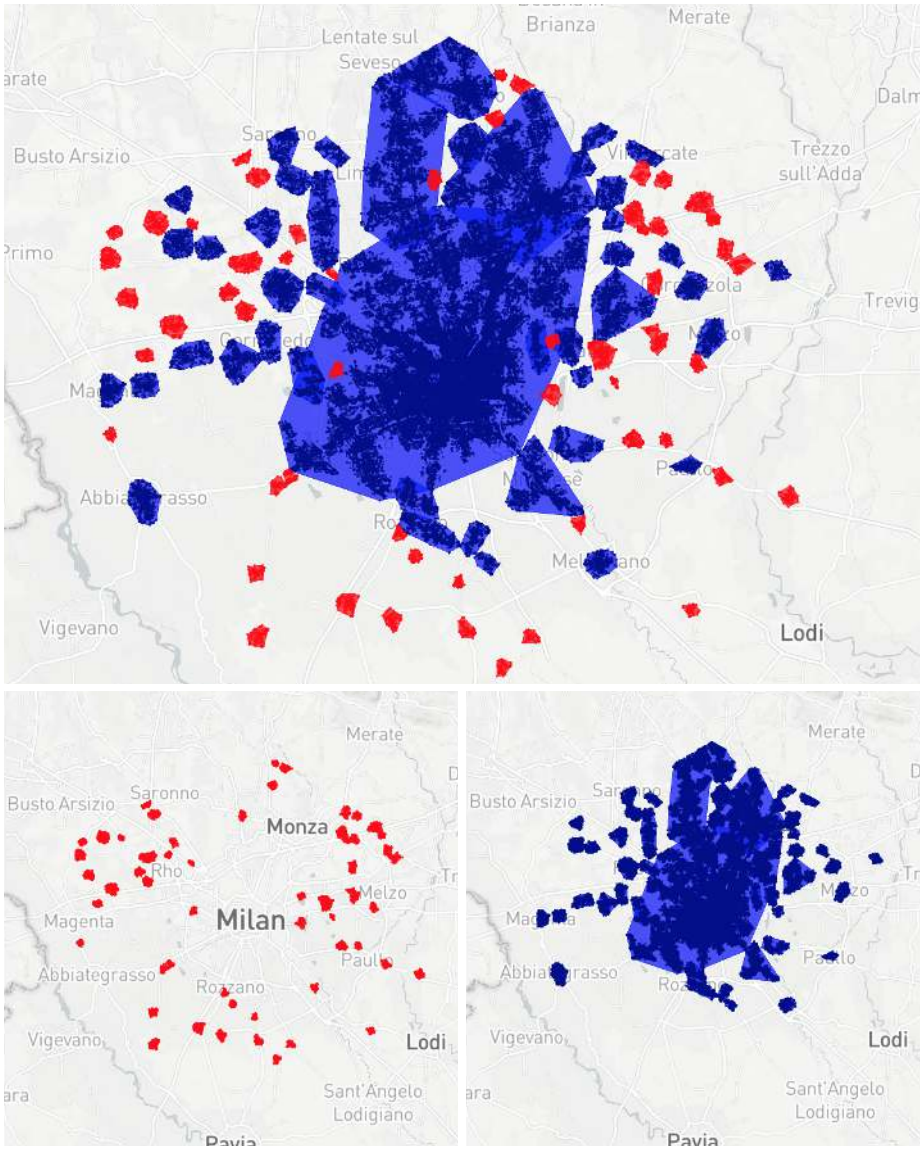


Figura 5.5: Visualizzazione del livello primo della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

Livello 2

La figura 5.6 fa riferimento al secondo livello della gerarchia, in essa è possibile osservare i primi cluster appartenenti a dataset diversi, ottenuti dall'iterazione precedente. I cluster estratti in questo livello, alcuni sono stati etichettati come sotto-dataset altri invece come cluster-indivisibili. Anche in questo livello le aree interne di Milano e Monza sembrano necessitare di ulteriori suddivisioni. Mentre alcune aree periferiche sembrano essere già sufficientemente rappresentate da cluster.

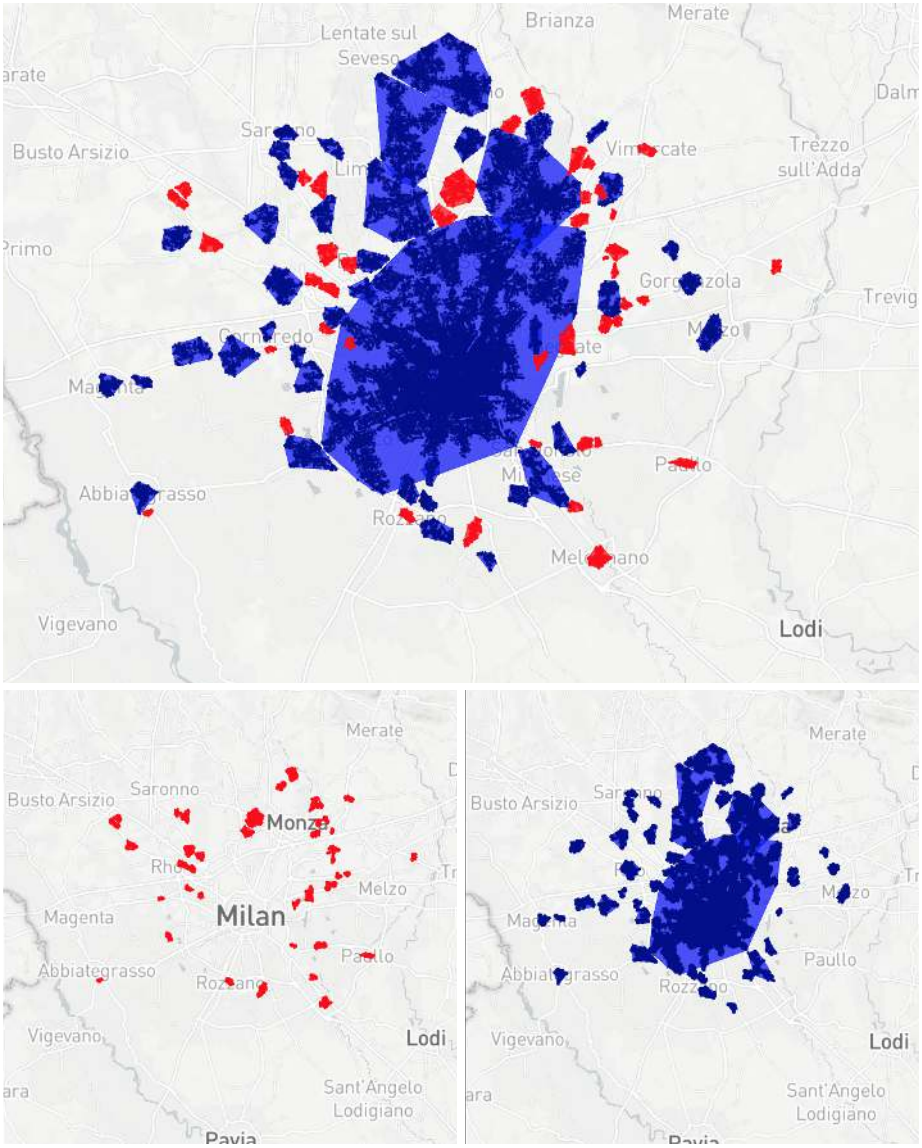


Figura 5.6: Visualizzazione del livello 2 della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

Livello 3

La figura 5.7 si riferisce al terzo livello della gerarchia che inizia a identificare Cluster di dimensione sempre inferiore, le uniche aree che necessitano di un ulteriore divisione sono le aree più centrali di Milano e alcuni centri urbani esterni alla città. Un aspetto interessante di questa visualizzazione è la necessità molto più elevata delle aree a nord di Milano rispetto alle aree a sud, di essere ulteriormente analizzate, questo è probabilmente dovuto ad una più forte differenza di urbanizzazione tra le due zone.

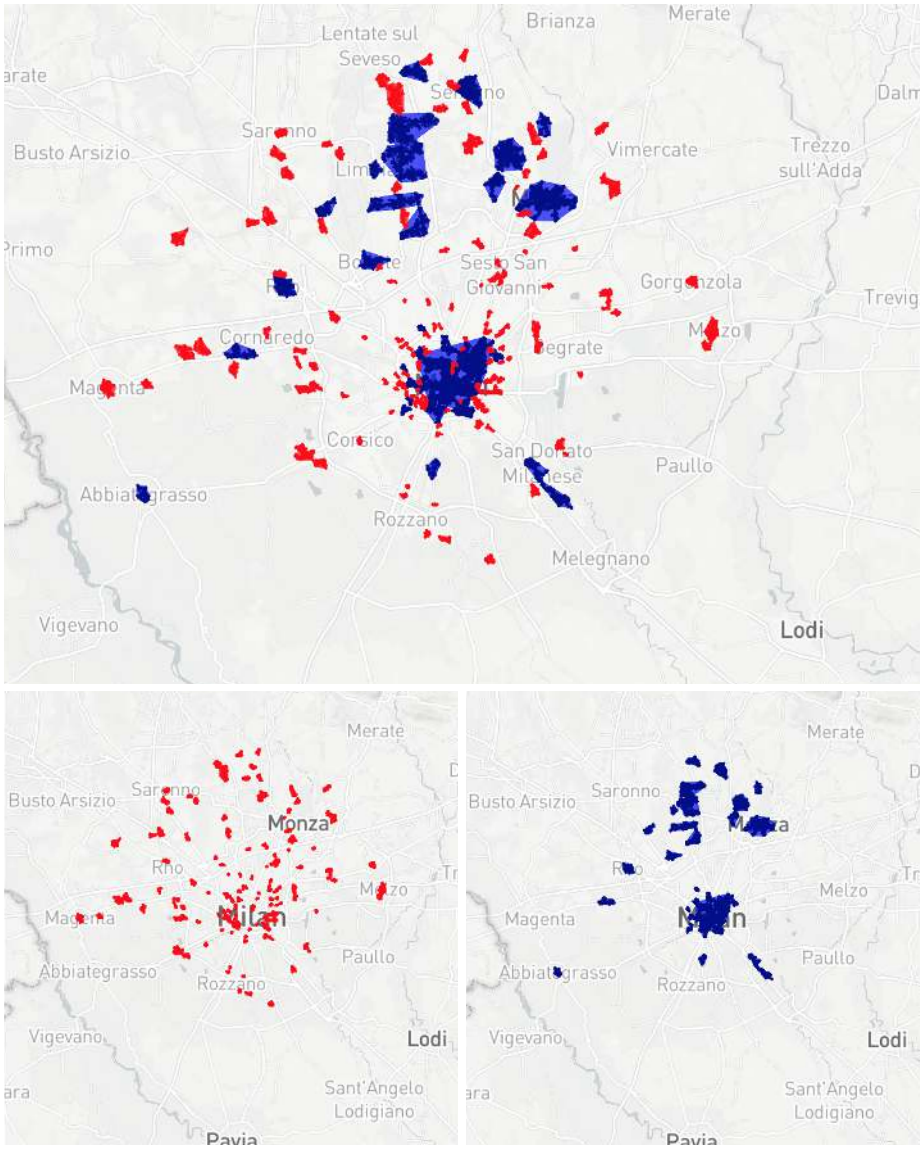


Figura 5.7: Visualizzazione del livello 3 della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

Livello 4

La figura 5.8 si riferisce al quarto livello della gerarchia che interessa una porzione del territorio sempre minore. In particolare, la maggior parte delle aree è ormai ritenuta sufficientemente rappresentata dai cluster estratti, compresa l'area centrale di Milano. Mentre, a nord della città rimangono alcune piccole aree che necessitano di ulteriori analisi.

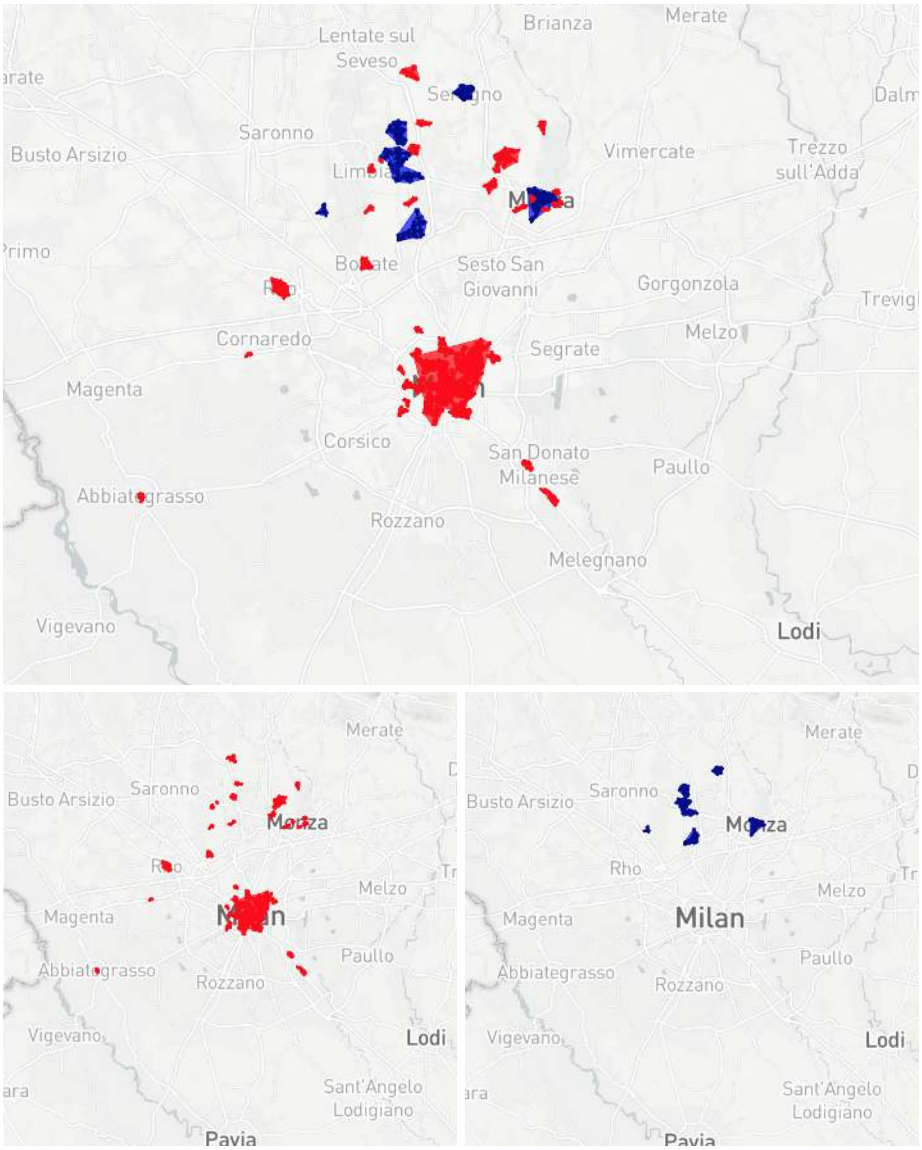


Figura 5.8: Visualizzazione del livello 4 della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

Livello 5

Infine, la figura 5.9 rappresenta l'ultimo livello della gerarchia che identifica gli ultimi Cluster-indivisibili e Sotto-dataset di piccolissima dimensione che, nel passaggio successivo, sono stati poi declassati a Cluster-indivisibili interropendo così il processo iterativo di clustering.

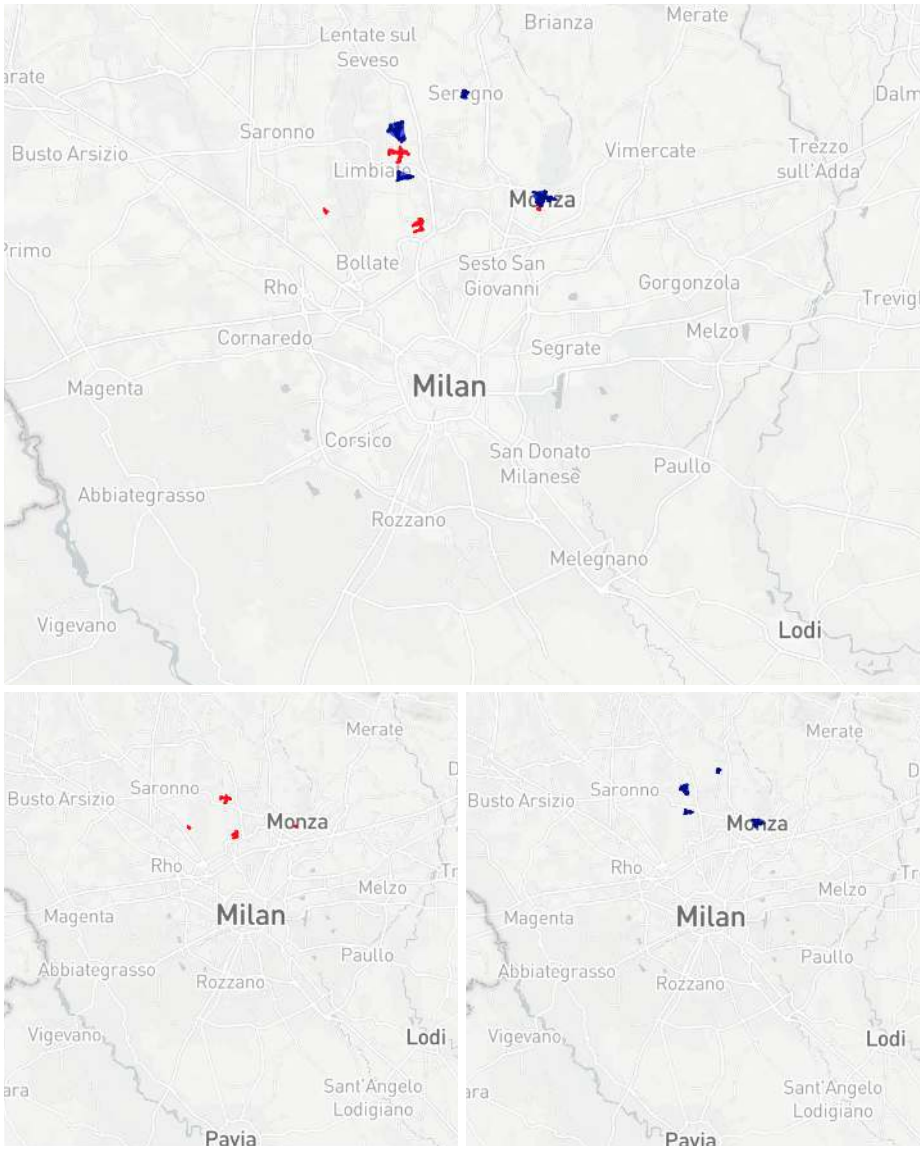


Figura 5.9: Visualizzazione del livello 5 della gerarchia ottenuta attraverso il processo di Clustering iterativo. In rosso i Cluster-Indivisibili, mentre in blu i Sotto-Dataset

5.1.3 Le aree estratte dal processo

Il processo di clustering gerarchico-iterativo applicato ai Place del territorio di Milano ha permesso la realizzazione di una gerarchia di Cluster di aree densamente servite da servizi con diversi livelli di densità, ben separate da zone a bassa densità di servizi. Estraendo, da tale gerarchia, i soli Cluster foglia è possibile identificare alcuni AOI con confini bottom-up delle aree più servite della città che nelle sezioni successive saranno descritti in modo più accurato.

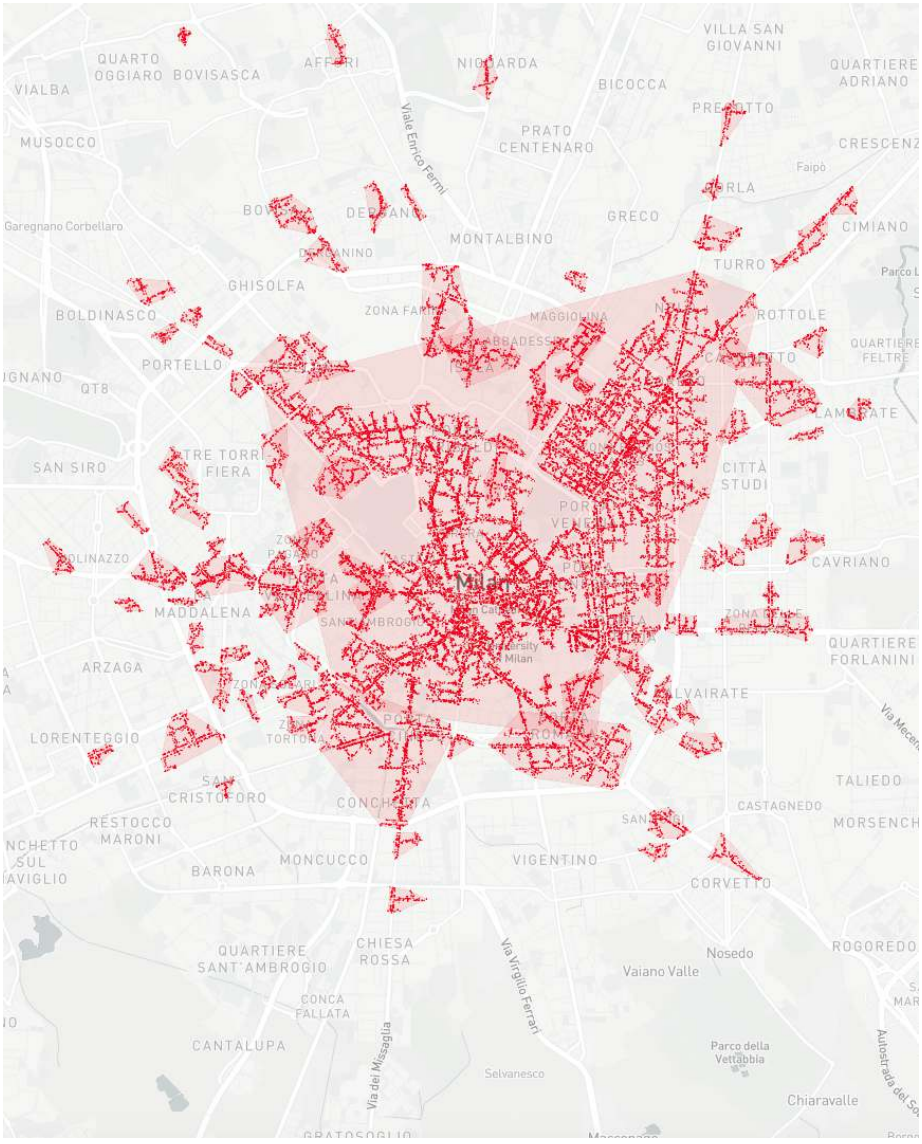
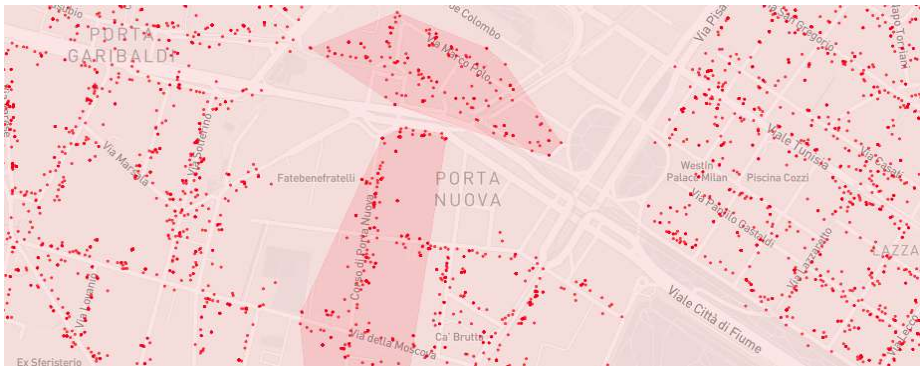


Figura 5.10: Le aree più densamente servite da servizi dell'area centrale di Milano secondo l'algoritmo di clustering gerarchico-iterativo

Aree di interesse

In figura 5.11 è possibile osservare alcune popolari aree di interesse (AOI) della città di Milano (Porta Nuova e zona Stazione Centrale). Tali aree sono particolarmente interessanti in quanto pur essendo immerse in aree abbastanza servite da servizi, esse emergono comunque come aree maggiormente servite a causa di una maggiore densità e di una certa separazione dalle aree circostanti.



(a) Zona Porta Nuova, Milano

(b) Zona Stazione Centrale, Milano

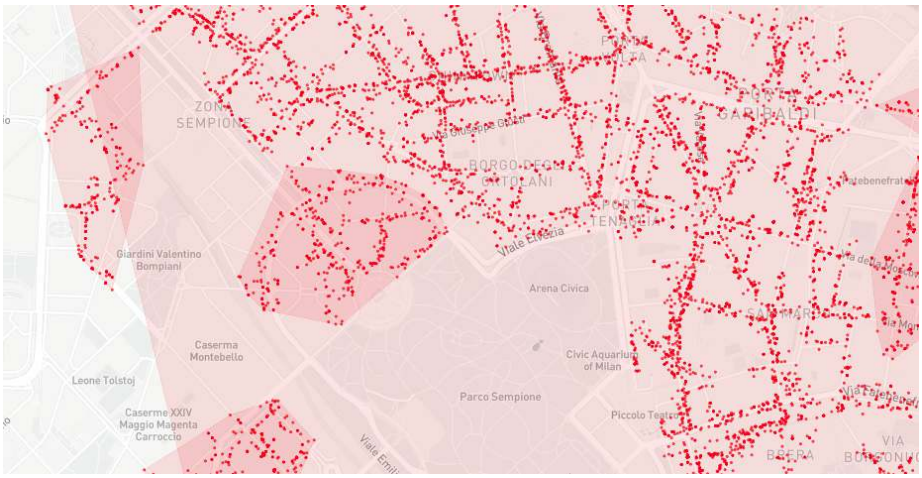
Figura 5.11: Alcune aree di interesse emerse dal processo di clustering iterativo rispetto alla densità dei servizi presenti

Strutture ed ostacoli urbani

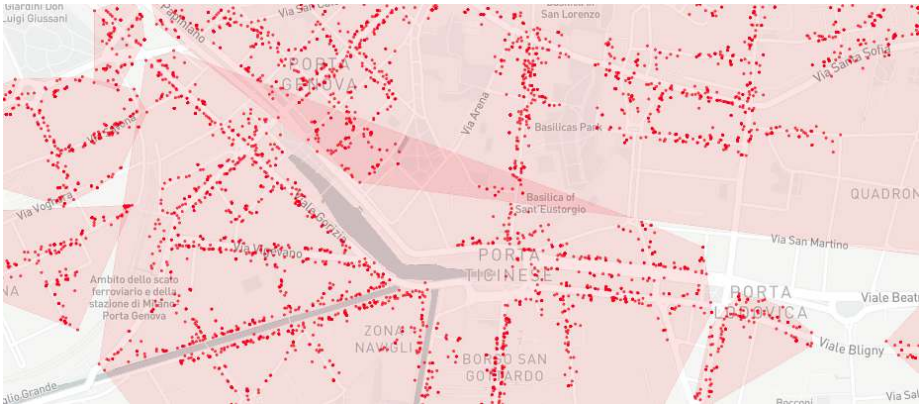
Le grandi strutture urbane rappresentano spesso un ostacolo alla presenza di servizi. Ad esempio, in figura 5.12a, l'area del Parco Sempione ha facilitato l'emergere della zona Arco della Pace come Cluster.

In modo analogo, anche il Naviglio Grande ha condizionato l'identificazione dei cluster, come è possibile osservare in figura 5.12b. Infatti il Naviglio Grande sembra rappresentare un ostacolo tale da spingere l'algoritmo a identificare due Cluster differenti, uno per la zona a nord ed uno per la zona a sud del Naviglio. É quindi probabile che in assenza di tale ostacolo le due aree non sarebbero state identificate come separate.

Questo esempio mostra come le grandi strutture urbane (naturali o artificiali) possono influenzare in modo significativo l'identificazione di aree ad alta densità di servizi.



(a) Zona Parco Sempion e Arco della Pace, Milano



(b) Zona Naviglio Grande, Milano

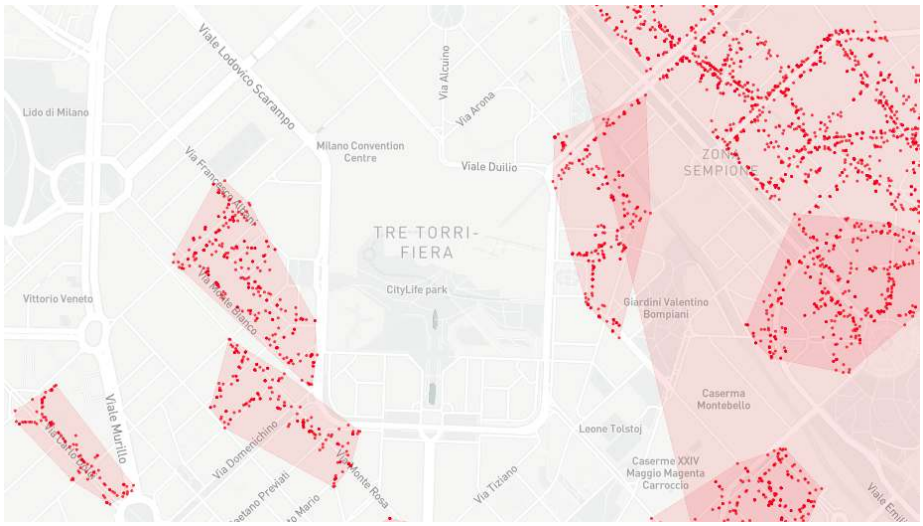
Figura 5.12: Aree di interesse ed ostacoli urbani a Milano

L'evoluzione delle aree urbane

Un altro interessante fenomeno catturato dall'algoritmo è l'evoluzione che il territorio urbano subisce giorno per giorno. Infatti spesso le aree della città mutano di importanza in base alla presenza di servizi o viceversa.

Ne è un esempio l'area Tre Torri Fiera di Milano che in figura 5.13a appare come area a bassa densità di servizi, invece è noto che attualmente in quest'area sorge il grande complesso commerciale City Life Shopping District[1] che al momento della raccolta dei dati era ancora in fase di realizzazione, inaugurato poi solo qualche mese più tardi (30 novembre 2017).

L'area analizzata, a differenza di altre (figura 5.13b), appare come completamente vuota. Mentre raccogliendo nuovamente i dati tale l'area potrebbe essere identificata come un cluster ad alta densità di servizi.



(a) Zona Tre Torri, Milano; Agosto 2017

(b) Centro Commerciali Vulcano, Sesto San Giovanni: Agosto 2017

Figura 5.13: Complessi commerciali a confronto rispetto ai servizi disponibili in agosto 2017

[1] https://citylifeshoppingdistrict.it/it/

5.2 Analisi dei Type

In questa sezione si è esplorata la possibilità di automatizzare la caratterizzazione dei Cluster estratti dall'algoritmo. L'idea di base consiste nel fatto che ogni Cluster è un insieme di Place caratterizzati da diverse informazioni, tra le quali i *Type* (vedi sezione 3.1.3). Dunque, un Cluster può essere descritto attraverso i *Type* che i propri Place possiedono.

Basandoci su questo concetto sono state esplorate alcune tecniche di analisi allo scopo di fornire informazioni sulla natura dei Cluster identificati attraverso un'analisi quantitativa, senza quindi considerare i significati semantici dei diversi *Type*, ovvero, presi due *Type* semanticamente correlati, sono considerati in ogni caso come differenti.

5.2.1 Cenni di base e preparazione dei dati

Prima di procedere con le analisi dei *Type* è stato necessario preparare i dati scegliendo una rappresentazione adeguata per le fasi successive.

Rappresentazione Vettoriale

La rappresentazione vettoriale consiste nel descrivere attraverso vettori numerici contenuti complessi quali immagini o testo. Tale rappresentazione è di cruciale importanza per un gran numero di analisi, per questo motivo si è voluto dedicare del tempo ad una buona formalizzazione dei vettori e del loro significato così da fornire una base di lavoro per analisi future.

Type dei Cluster Per ogni Cluster estratto disponiamo di un certo numero di Place. Per ognuno dei Place sia ha disposizione uno o più $Type$ presi da una lista, dunque, aggregando i *Type* contenuti nei Place è possibile caratterizzare i Cluster in due modalità:

- calcolando il numero o la percentuale dei Place in cui ogni $type_i$ appare;
- analizzando la presenza/assenza totale di ogni $type_i$ nei Cluster.

Vettore della presenza dei Type nei Place Data la lista dei *Type* indicata da Google (vedi sezione 3.1.3) è stato definito un vettore che per ogni *Place* identifica la presenza o l'assenza di uno specifico *Type*.

Ovvero, data la lista dei $Type = \{type_1, ..., type_n\}$, possiamo costruire un vettore $\vec{T_i}$ su un i-esimo Place indicando la

presenza/assenza dell'j-esimo *Type* in un determinato Place come segue:

$$\vec{T}_{place_j} = \{t_0, ..., t_n\}, t_i \in \{0, 1\},\ 1 = presente, 0 = assente \tag{5.1}$$

Vettore della presenza dei Type nei Cluster Dato un Cluster $C_c = \{place_1, ..., place_x\}$, il vettore della presenza/assenza di un *Type* nel c-esimo Cluster è

$$\vec{T}^{presence}_{Cluster_c} = \{t_0, ..., t_n\}, t_i \in \{0, 1\},\ 1 = presente, 0 = assente \tag{5.2}$$

Tale che t_i sarà :

- 0 se l'i-esimo *Type* non è presente in nessun Place del Cluster c-esimo
- 1 se l'i-esimo *Type* è presente in almeno un Place del Cluster c-esimo

Ottenendo cosi una matrice che rappresenta la presenza/assenza di ogni *Type* su tutti i Cluster considerati.

	$type_1$	$type_2$	...	$type_n$
C_1	0	1	...	0
C_2	1	0	...	0
⋮	⋮	⋮	...	⋮
C_c	t_1	t_2	...	t_n

Vettore delle occorrenze dei Type nei Cluster L'occorrenza di un *Type* rispetto al Cluster può essere rappresentata in due modalità:

- in termini assoluti: ovvero pari al numero di Place in cui compare
- in termini relativi: ovvero pari alla percentuale dei Place (rispetto al totale del Cluster) in cui compare;

É stata preferita la quella in termini relativi perché meno influenzata dalla differenza del numero di Place contenuti in ogni Cluster.

Occorrenze assolute Dato un Cluster $C_j = \{place_1, ..., place_x\}$, il vettore delle occorrenze assolute di un *Type* nel j-esimo è un vettore

$$\vec{T}^{occAbs}_{Cluster_j} = \{t_0, ..., t_n\}, t_n \in [0 \rightarrow |C_j|] \tag{5.3}$$

Tale che t_n sarà :

- 0 se n-esimo *Type* non è presente in nessun Place del Cluster j-esimo

- $|C_j|$ se n-esimo *Type* è presente in tutti i Place del j-esimo Cluster

	$type_1$	$type_2$	...	$type_n$
C_1	7	10	...	0
C_2	5	1	...	35
⋮	⋮	⋮	...	⋮
C_x	t_1	t_2	...	t_n

Occorrenze relative Mentre, la rappresentazione delle occorrenze relative t_n sarà :

- 0 se l'n-esimo *Type* non è presente in nessun Place del cluster j-esimo;
- $\frac{p}{|C_j|} * 100$ se l'n-esimo *Type* è presente p-volte rispetto alla dimensione del Cluster C_j;
- 100 nel caso è presente in tutti i Place del Cluster C_j.

	$type_1$	$type_2$	...	$type_n$
C_1	56	6	...	100
C_2	87	0	...	42
⋮	⋮	⋮	...	⋮
C_x	t_1	t_2	...	t_n

5.2.2 Analisi vettoriali

Presenza/assenza dei Type nei Cluster

Basandoci sulle rappresentazioni vettoriali è stato possibile realizzare alcune visualizzazioni che mostrano la presenza o l'assenza di un determinato *Type* nei vari Cluster.

Una modalità di rappresentare l'assenza/presenza dei *Type* è attraverso un *Heat Map* a due colori (assente, presente), come in figura 5.14, dove il colore chiaro rappresenta la presenza mentre il colore scuro indica l'assenza del *Type* nel Cluster.

La figura 5.14 raffigura sulle ordinate i Cluster ordinati per numero di Place, dove in alto troviamo il più popoloso, mentre in basso il meno popoloso. Invece l'asse delle ascisse raffigura i *Type* in ordine alfabetico, da sinistra a destra. Tale visualizzazione mette in luce tre fatti sui *Type* presenti nei Cluster:

- alcuni *Type* sono sempre presenti, enfatizzati dalle linee verticali chiare;

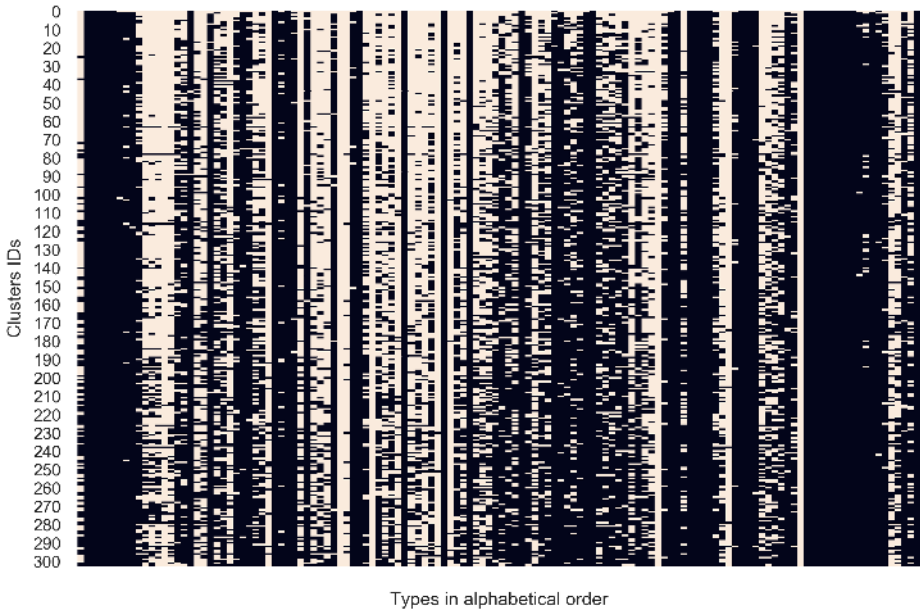


Figura 5.14: Assenza/Presenza di un *Type* in un Cluster foglia

- alcuni *Type* sono sempre assenti, raffigurati dalle linee verticali di colore scuro;

- i Cluster più popolosi (quelli più in alto) coprono una tipologia di *Type* lievemente maggiore, infatti alcune linee si affievoliscono mano a mano che si scende nel grafico.

Numero di presenze dei Type nei Cluster

Ordinando i *Type* rispetto alla loro presenza/assenza nei Cluster possiamo produrre una ulteriore raffigurazione, dove le ordinate rappresentano il numero di Cluster in cui compare almeno una volta l'i-esimo *Type*, mentre le ascisse raffigurano i singoli *Type*, in questo caso ordinati dal più presente (a sinistra) al meno presente (a destra) rispetto ai Cluster (vedi figura 5.15).

Tale visualizzazione mostra che alcuni *Type* sono presenti in almeno un Place in tutti i Cluster, mentre altri sono completamente assenti in tutti i Cluster. Un numero cospicuo di *Type* nell'area centrale si distribuiscono in modo estremamente regolare con piccoli gradini.

L'ipotesi non confermata della presenza di gradini della figura 5.15 è che alcuni *Type vicini* con occorrenza simile possano avere una qualche correlazione semantica: vedi ad esempio: *hospital* e *healt*, oppure *university* e *education*.

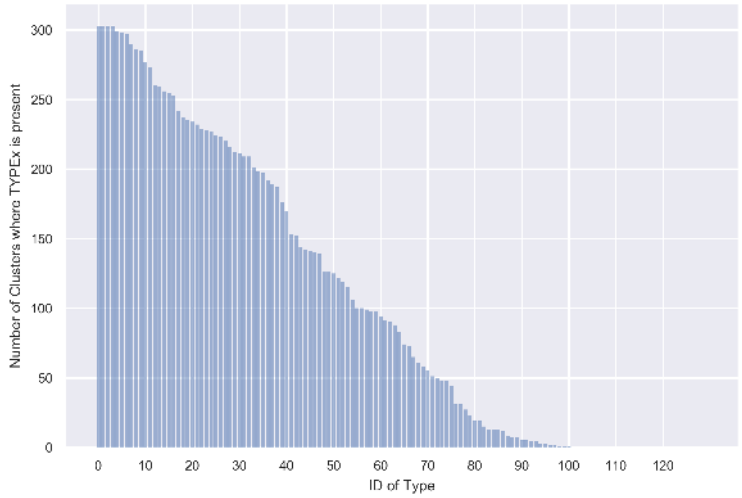


Figura 5.15: In quanti Cluster un determinato Type è presente in almeno un Place. Sulle ascisse è rappresento il numero di Cluster, sulle ordinate sono rappresentai i singoli Type, questi ultimi ordinati per presenza/assenza nei Cluster

Presenza/assenza in ordine di occorrenza nei Place

Per mettere in luce alcune ulteriori correlazioni tra Cluster e *Type* si è realizzato una visualizzazione di presenza/assenza dei *Type*, ordinando questi ultimi sulla base della loro presenza nei Place come in figura 5.16, ovvero il *Type* presente in più Place in assoluto sarà sulla sinistra, il meno presente apparirà sulla destra. Mentre i Cluster, anche in questa visualizzazione, sono ordinati in base al numero di Place che contengono.

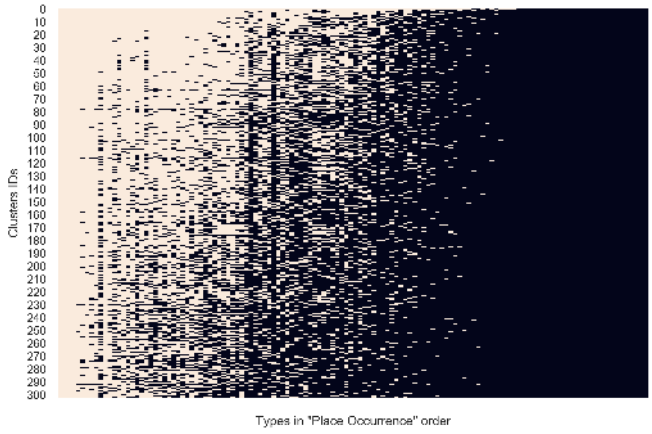


Figura 5.16: Assenza/Presenza di un Type in un Cluster, in ordine di presenza rispetto ai Cluster

Anche in questo caso la presenza di un *Type* in un Cluster è rappresentata dal colore chiaro, mentre l'assenza dal colore scuro.

Tale rappresentazione conferma in modo ancor più forte il fatto che alcuni *Type* sono sempre presenti, atri invece sempre assenti. Questi ultimi in numero molto maggiore. In oltre nell'area centrale del grafico in figura 5.16 si possono osservare un gran numero di *Type* presenti in misura via-via inferiore.

In altre parole, i Cluster identificati sono tutti molto simili tra loro in termini di presenza/assenza dei *Type*, se esistono degli out-layer sono rappresentati da un numero esiguo di Cluster che si differenziano per pochi *Type*

Infine in merito all'ordine verticale dei Cluster sembra esserci una leggera differenza tra i Cluster più popolosi in alto rispetto ai meno popolosi. Dove i primi sembrano possedere più *Type* rispetto ai meno popolosi in basso, ma la differenza è molto lieve.

Occorrenza assoluta dei Type rispetto ai Place

Una ulteriore domanda che possiamo formulare è "Quanto sono popolari determinati *Type* rispetto ai al numero assoluto di Place?"

Nella figura 5.17, il numero di Place in cui è presente un *Type* è rappresentato sull'asse delle ordinate, mentre i *Type* sono raffigurati sull'asse delle ascisse, questi ordinati per occorrenza assoluta ovvero il più presente è il più a sinistra, il meno presente è il più a destra.

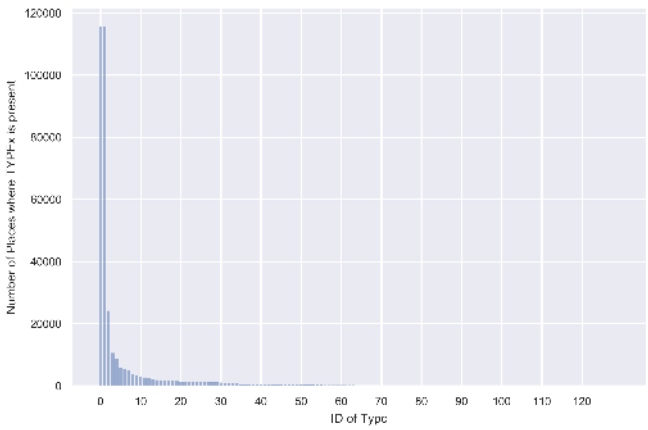


Figura 5.17: Occorrenze dei Type rispetto al numero assoluto di Place contenuti nei vari Cluster

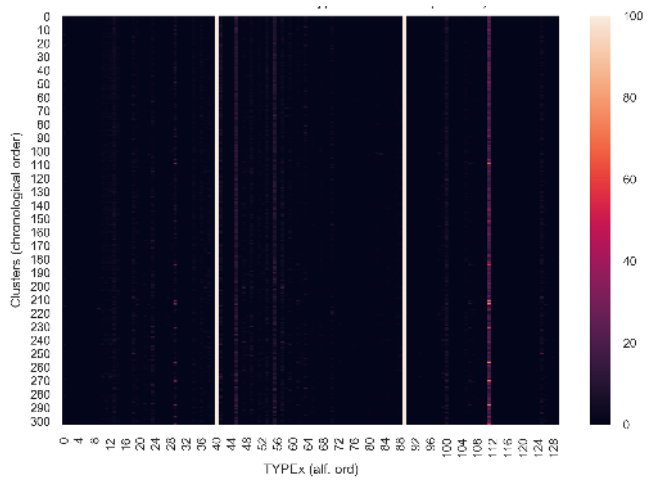
Risulta evidente che alcuni *Type* sono onnipresenti, mentre molti altrri hanno una occorrenza piuttosto bassa, in oltre la differenza in termini di occorrenza nei Place è estremamente alta tra i primi ed i successivi. Dunque pochi *Type* rappresentano i *Type* più presenti nei Place, questo rende i Cluster simili tra loro e poco differenziati, confermando che solo pochi *Type* diversificano in modo significativo i Cluster.

Occorrenza relativa dei Type

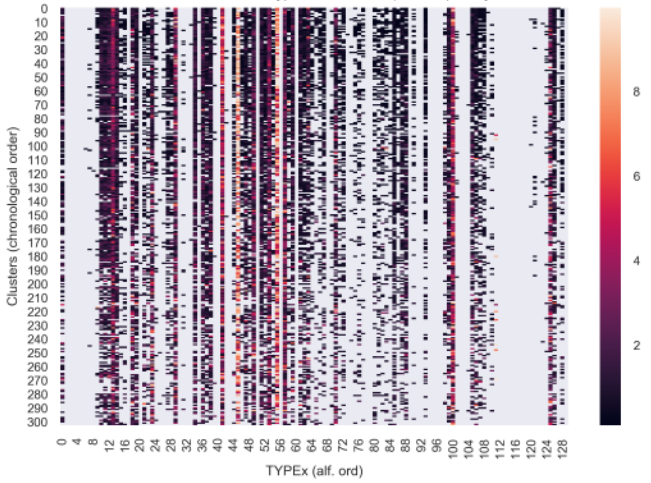
Un ulteriore possibile rappresentazione della presenza dei *Type* rispetto ai Cluster è quella normalizzata rispetto al numero di Place nei Cluster, ovvero per ogni *Type* se ne è calcolata la presenza in percentuale rispetto alla numerosità totale del Cluster

La figura 5.18a il colore nero indica lo 0% di presenza nel Cluster mentre il colore chiaro indica il 100% di presenza nel determinato Cluster.

Attraverso la figura 5.18a è chiaro che 2 *Type* rendono quasi inutilizzabile la rappresentazione, essi sono presenti nel 100% dei ogni Cluster, mentre il restante sembra essere presente in misura troppo bassa per essere individuabili con questa rappresentazione.



(a) Numerosità dei Type in percentuale sulla dimensione dei Cluster



(b) Numerosità in percentuale sulla dimensione dei Cluster escludendo i Type al di sopra del 10% ed al di sotto del 1%

Figura 5.18

Per ovviare a questo problema di visualizzazione è stata realizzata una versione alternativa della figura 5.18a, dove questa volta i *Type* raffigurati sono solo quelli compresi tra l'1% ed il 10% di presenza nei vari Cluster, escludendo sia quelli al di sotto che al di sopra del range attraverso il colore grigio (vedi figura 5.18b).

Tale visualizzazione dei dati (figura 5.18b) mostra con chiarezza ancora una volta come la maggior parte dei *Type* sono compresi tra l'1% ed il 3% (colore molto scuro), mentre pochissimi sono compresi tra il 4% ed il 10%. Questo conferma in modo inequivocabile che molti *Type* sono presenti pochissime volte anche all'interno di uno stesso Cluster.

5.2.3 Similarità tra Cluster

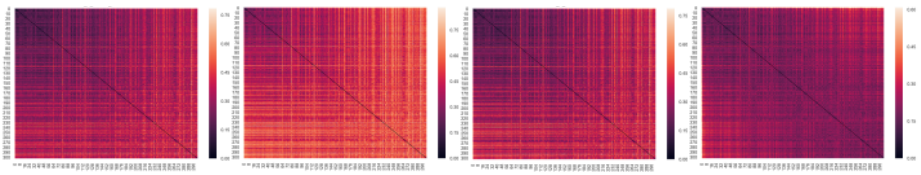
Un diverso modo di confrontare i Cluster può essere quello di utilizzare le formule di distanza attraverso le rappresentazioni vettoriali dei Cluster basati sulla presenza/assenza di un *Type* in almeno un Place.

Dati i vettori $\vec{v}_{C_x} = \{t_0, ..., t_n\} \in \{1, 0\}$ realizzati in precedenza che indicano per ogni Cluster quali *Type* sono presenti e la rispettiva matrice di tutti i Cluster:

	$type_1$	$type_2$	...	$type_n$
C_1	0	1	...	0
C_2	1	0	...	0
⋮	⋮	⋮	...	⋮
C_x	t_1	t_2	...	t_n

Data una distanza $d_i()$ è possibile calcolare la distanza per ogni coppia di Cluster, rappresentando il risultato attraverso una *Heat Map* le cui ascisse e ordinate rappresentano a specchio i Cluster in ordine di numerosità di Place. Ottenendo così una raffigurazione a matrice triangolare dove la diagonale rappresenta la similarità di ogni Cluster con se stesso.

Le formule di similarità sperimentate con tale rappresentazione sono la Cosine Distance, Jaccard Distance, Dice Distance e la Matching Distance; tutte descritte in dettaglio nelle sezioni successive.



(a) Cosine D. (b) Jaccard D. (c) Dice D. (d) Matching D.

Figura 5.19: Confronto tra le visualizzazioni delle diverse formule di similarità applicate alle rappresentazioni vettoriali dei Cluster

Cosine Distance - presenza/assenza Type

Per ogni coppia di Cluster-indivisibili è stata calcolata la similarità attraverso la Cosine Distance:

$CosineDistance(\vec{u}_{C_x}, \vec{v}_{C_y})$ dove $\vec{u}_{C_x}, \vec{v}_{C_y} \in 1, 0$

dove lo 0 indica l'assenza del *Type* in tutti i Place del Cluster, mentre l'1 indica la presenza in almeno un Place del cluster

La similarità è stata calcolata attraverso la libreria Python Scipy[2] utilizzando il seguente metodo che calcola la formula 5.4, raffigurata in figura 5.20.

Metodo Python

```
d = scipy.spatial.distance.cosine(u,v)
```

Formula

$$CosineDistance(\vec{u}, \vec{v}) = 1 - \frac{u \cdot v}{\|u\|_2 \cdot \|v\|_2} \tag{5.4}$$

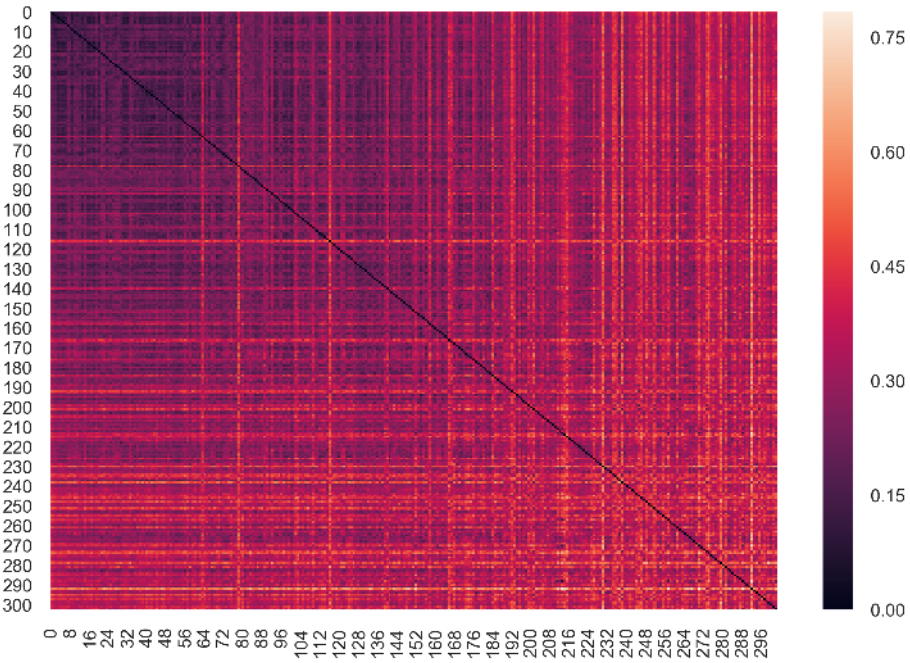


Figura 5.20: Matrice della similarità tra Cluster utilizzando la formula di similarità Cosine Distance

[2] https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.distance.cosine.html

Jaccard Distance - presenza/assenza Type

Per ogni coppia di Cluster-indivisibili è stata calcolata la similarità attraverso la Jaccard Distance:

$JaccardDistance(\vec{u}_{C_x}, \vec{v}_{C_y})$ dove $\vec{u}_{C_x}, \vec{v}_{C_y} \in 1, 0$

dove lo 0 indica l'assenza del *Type* in tutti i Place del Cluster, mentre l'1 indica la presenza in almeno un Place del cluster

La similarità Jaccard Distance è stata calcolata attraverso la libreria Python Scipy[3] utilizzando il seguente metodo che calcola la formula 5.5, raffigurata in figura 5.21.

Metodo Python

```
d = scipy.spatial.distance.jaccard(u,v)
```

Formula

$$JaccardDistance(\vec{u}, \vec{v}) = \frac{C_{FT} + C_{TF}}{C_{TT} + C_{TF} + C_{FT}} \tag{5.5}$$

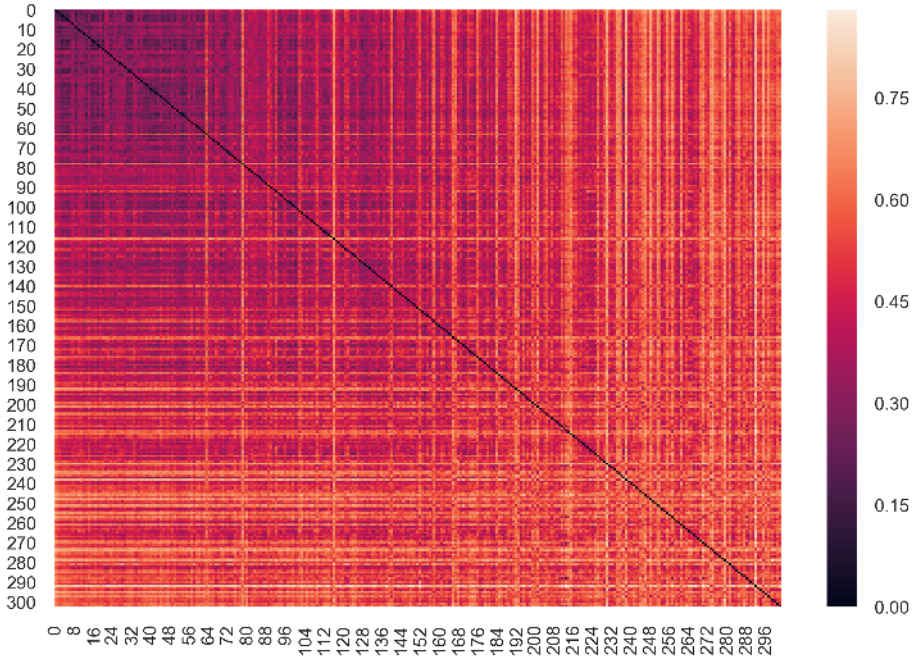


Figura 5.21: Matrice della similarità tra Cluster utilizzando la formula di similarità Jaccard Distance

[3] https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.jaccard.html

Dice Distance - presenza/assenza Type

Per ogni coppia di Cluster-indivisibili è stata calcolata la similarità attraverso la Dice Distance:

$DiceDistance(\vec{u}_{C_x}, \vec{v}_{C_y})$ dove $\vec{u}_{C_x}, \vec{v}_{C_y} \in 1, 0$

dove lo 0 indica l'assenza del *Type* in tutti i Place del Cluster, mentre l'1 indica la presenza in almeno un Place del cluster

La similarità Dice Distance è stata calcolata attraverso la libreria Python Scipy[4] utilizzando il seguente metodo che calcola la formula 5.6, raffigurata in figura 5.22.

Metodo Python

```
d = scipy.spatial.distance.dice(u,v)
```

Formula

$$DiceDistance(\vec{u}, \vec{v}) = \frac{C_{FT} + C_{TF}}{2C_{TT} + C_{TF} + C_{FT}} \tag{5.6}$$

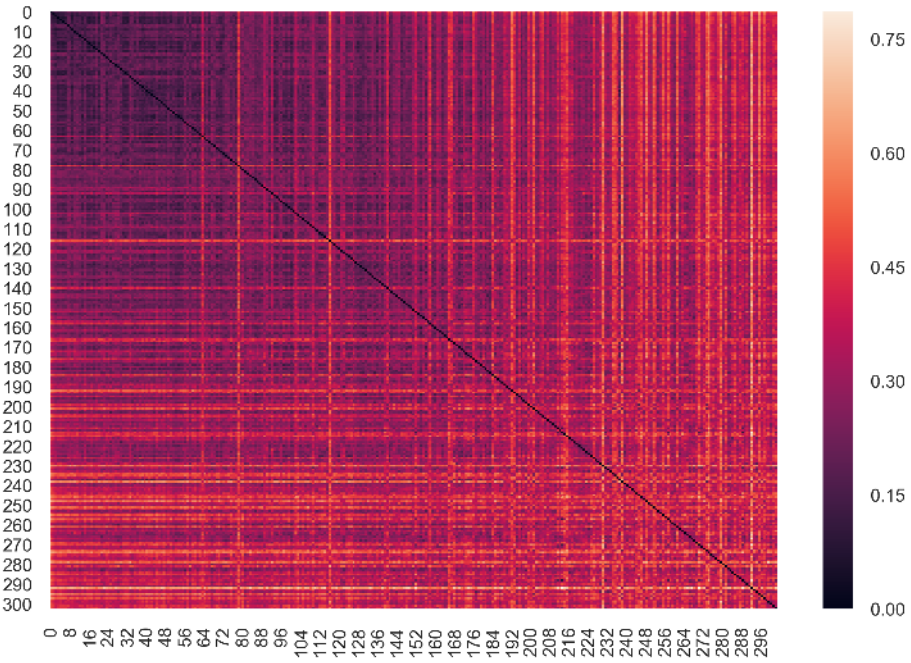


Figura 5.22: Matrice della similarità tra Cluster utilizzando la formula di similarità Dice Distance

[4] https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.dice.html

Matching Distance - presenza/assenza Type

Per ogni coppia di Cluster-indivisibili è stata calcolata la similarità attraverso la Matching Distance:

$MatchingDistance(\vec{u}_{C_x}, \vec{v}_{C_y})$ dove $\vec{u}_{C_x}, \vec{v}_{C_y} \in 1, 0$

dove lo 0 indica l'assenza del *Type* in tutti i Place del Cluster, mentre l'1 indica la presenza in almeno un Place del Cluster

La similarità Matching Distance è stata calcolata attraverso la libreria Python Scipy [5] che rimanda alla formula di Hamming[6] definendo deprecata la Matching Distance. Dunque la formula in 5.7 è la Hamming Distance, raffigurata in figura 5.23.

Metodo Python

```
d = scipy.spatial.distance.matching(u,v)
```

Formula

$$HammingDistance(\vec{u}, \vec{v}) = \frac{c_{01} + c_{10}}{n} \tag{5.7}$$

dove c_{ij} è il numero di occorrenze di $\vec{u}[k] = i$e $\vec{v}[k] = j$per $k < n$.

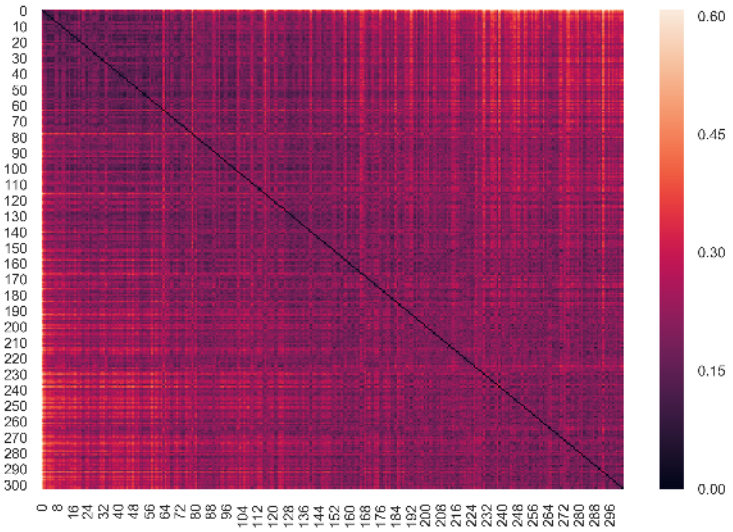


Figura 5.23: Matrice della similarità tra Cluster utilizzando la formula di similarità Matching Distance

[5] https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/generated/scipy.spatial.distance.matching.html

[6] https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/scipy.spatial.distance.hamming.html

Cosine Distance - occorrenza Type nomralizzata

Per ogni coppia di Cluster-indivisibili è stata calcolata la similarità attraverso la Cosine Distance:

$CosineDistance(\vec{u}_{C_x}, \vec{v}_{C_y})$ dove $\vec{u}_{C_x}, \vec{v}_{C_y} \in 0, 100$

dove lo 0 indica l'assenza del *Type* in tutti i Place del Cluster, mentre l'100 indica la presenza in tutti i Place del cluster. Sempre calcolata attraverso la libreria Python Scipy[7] utilizzando la formula 5.4 raffigurata in figura 5.24.

Metodo

```
d = scipy.spatial.distance.matching(u,v)
```

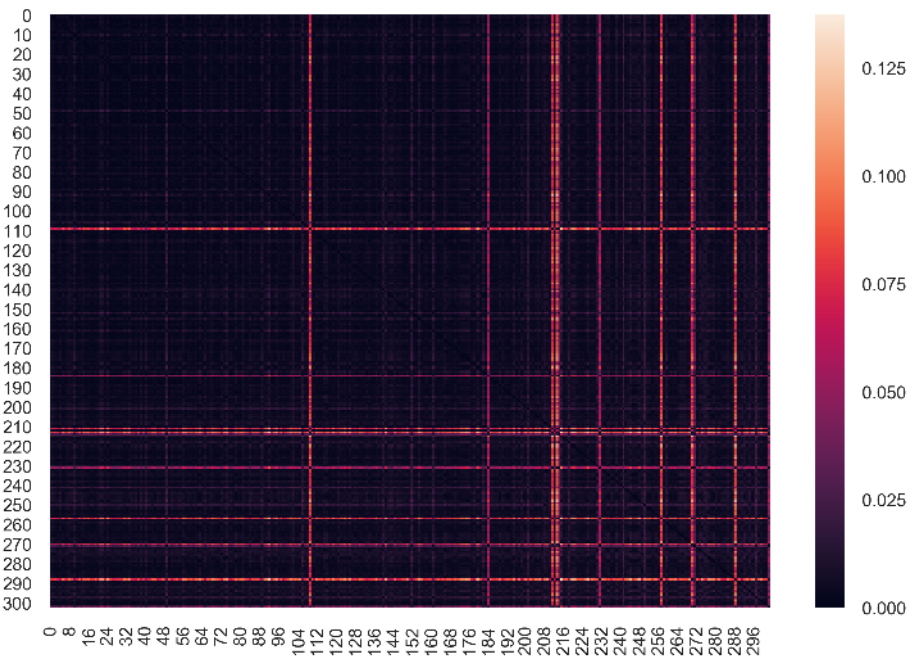


Figura 5.24: Matrice della similarità tra Cluster rispetto la numerosità della presenza dei Type normalizzata sul numero totale di Place nei Cluster utilizzando la formula di similarità Cosine Distance

[7] https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.distance.cosine.html

Capitolo 6

Conclusioni e sviluppi futuri

6.1 Conclusioni

Questo lavoro realizza un'analisi di dati geospaziali per applicazioni di Urban Informatics sul territorio di Milano attraverso l'acquisizione di dati geolocalizzati da Google Place e la successiva caratterizzazione delle aree urbane.

Tra le motivazioni del lavoro troviamo la continua evoluzione e diffusione di tecnologie informatiche che hanno portato alla digitalizzazione della vita quotidiana, la quale, sommandosi alla crescente urbanizzazione dei grandi centri urbani ed al fenomeno del trasferimento di persone dalle aree rurali verso i centri abitati, hanno fatto sì che si parli ormai sempre più di una Città Digitale fatta di flussi di informazioni sempre più eterogenei fra loro ed in quantità sempre più elevate.

Questo lavoro si posiziona all'interno di un emergente campo di ricerca che prende il nome di Urban Informatics. Tale campo di ricerca nasce dal connubio tra Scienze Informatiche e Scienze Urbane ed affonda le radici nelle tecnologie informatiche che sono sempre più pervasive nell'ambiente urbano e nella vita delle persone.

Tale contesto tecnologico fornisce un nuovo modo di vedere la città. Da una parte si ottiene una visione ottimistica della città che viene vista come una rete di persone, luoghi e tecnologie intelligenti che riescono ad interagire come un unico super-organismo. Dall'altra abbiamo il concetto dei Big Data nel contesto urbano e delle Smart City che mirano a risolvere i problemi della progettazione e della gestione della città attraverso processi e metodologie basate sui dati, quali: acquisizione, integrazione ed analisi per la comprensione e l'esplorazione della città.

In base a queste premesse si è scelto di utilizzare come sorgente dati i contenuti georeferenziati generati dagli utenti realizzati attraverso la piattaforma Google Place. Tali dati, frutto di un insieme di integrazioni tra dati degli utenti e dati generati dalla compagnia Google, offrono un quadro interessante dei servizi disponibili nelle diverse aree della città, rappresentando un potenziale modo di caratterizzare le diverse aree della città.

Acquisizione dei dati

In totale sono stati collezionati circa 290.000 place su una circonferenza di raggio circa $26km$ centrata nelle vicinanze del Parco Sempione di Milano. Tali place sono stati acquisiti attraverso un processo adattivo basato su un movimento a spirale, in grado di soffermarsi ed analizzare in modo più dettagliato alcune aree della mappa in base al numero di place identificati. La procedura adattiva ha permesso un'ottimizzazione del numero delle richieste verso la Web API di Google ed un'acquisizione più accurata dei place su tutta l'area analizzata, adeguandosi alle limitazioni imposte dal servizio di Google.

Di questi 290.000 place, circa 50.000 sono stati esclusi dalle analisi successive perché, rappresentando i nomi delle strade e delle vie, non erano utili a caratterizzare le aree della città in termini di servizi offerti. Ogni place acquisito è dotato di diverse informazioni, tra le quali alcune categorie che lo caratterizzano. Delle 129 categorie imposte da Google, circa 120 sono state effettivamente identificate nei restanti 240.000 place.

Tra le cinque più numerose categorie presenti nei place troviamo quelle relative al commercio, al cibo e alla salute. Mentre tra quelle meglio organizzate troviamo quelle relative ai trasporti, che comprende taxi, tram, bus, metropolitana e ferrovie; questo è probabilmente dovuto all'integrazione, da parte di sistemi automatici o personale di Google, di dati provenienti da banche dati dell'amministrazione comunale o dalle aziende dei trasporti.

Infine, tenendo conto delle combinazioni delle categorie identificate nei place, la maggior parte dei place possiede solo 2 o 3 categorie, mentre quelli che possiedono un numero maggiore di categorie è la parte meno importante, ma non per questo meno rappresentativa o interessante. Emerge quindi che, seppur Google offra un gran numero di place nell'area urbana di Milano e zone limitrofe, molti di tali place sono poco rappresentativi delle varie aree a causa del basso numero di categorie associate. Questo può essere causato da una difficoltà nel caratterizzare un place, la mancata selezione di una categoria in fase di inserimento del place oppure un basso interesse nell'inserire categorie nei place di minore interesse a scapito di quelli con un maggiore interesse.

Data Mining dei dati

La seconda fase del lavoro riguarda l'analisi dei dati attuata attraverso un processo di clustering gerarchico-iterativo basato sul DBSCAN, che ha come scopo quello di suddividere la città di Milano in aree attraverso un processo suddivisione bottom-up, ovvero raggruppando i place generati dagli utenti in cluster basati sulla densità.

Il processo di clustering gerarchico-iterativo basa il suo funzionamento su due idee di base:

- non conoscendo i valori di input del DBSCAN ($Eps, MinPts$) con cui ottenere il giusto clustering dello spazio, è possibile calcolare più clustering attraverso diversi input e quindi scegliere quello che più si avvicina alla suddivisione voluta;
- ogni cluster presente nella suddivisione scelta può, potenzialmente, essere ulteriormente suddiviso attraverso un nuovo processo di clustering;

Per far sì che queste due idee potessero funzionare sono stati inseriti due concetti:

- il clustering obiettivo, ovvero quel clustering che si vuole ottenere, formalizzato attraverso alcuni indicatori come il numero di cluster, la loro numerosità e la media dei place contenuti;
- un sistema di promozione dei cluster della suddivisione, da cluster a dataset, per essere ulteriormente suddivisi;

Il processo così realizzato, a partire dai 240.000 place iniziali, ha permesso di costruire un albero di cluster su 5 cinque livelli, dal quale circa 300 cluster sono stati selezionati tra quelli presenti sulle foglie dell'albero, ovvero quelli che l'algoritmo ha identificato come non ulteriormente divisibili, ma che al contempo contenessero almeno 25 place (vedi figura 6.1).

Analisi e caratterizzazione

L'ultima e terza fase del lavoro riguarda l'analisi e la caratterizzazione dei circa 300 cluster identificati dal processo precedente. La caratterizzazione è stata fatta in prima istanza sulle categorie associate ai place; in seconda istanza, ed in modo più esplorativo, sulle recensioni dei vari place.

Calcolando la presenza o l'assenza delle varie categorie in termini assoluti nei vari cluster identificati si è constatato che la distanza vettoriale (Cosine, Jaccard, Dice, Matching) tra un cluster e l'altro è

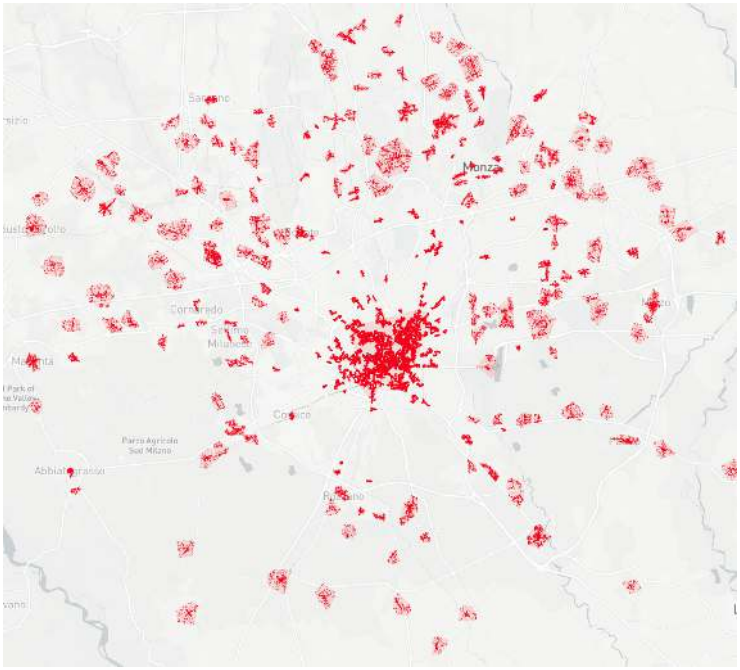
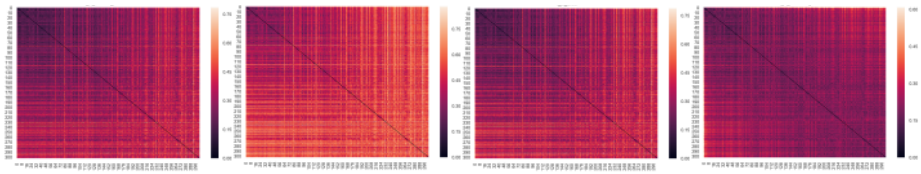


Figura 6.1: Cluster finali del processo di clustering gerarchico-iterativo basato sul DBSCAN applicato ai place della città di Milano e parte dei comuni limitrofi



(a) Cosine D. (b) Jaccard D. (c) Dice D. (d) Matching D.

Figura 6.2: Confronto tra le visualizzazioni delle diverse formule di similarità applicate alle rappresentazioni vettoriali dei Cluster

abbastanza regolare. In altri termini, i cluster contengono almeno una volta più o meno le stesse categorie, ovvero, non ci sono cluster che si differenziano in modo netto dagli altri, tali da contenere molte più categorie oppure molte meno. Riassumendo: i cluster, spesso, contengono un po' di tutto in termini di servizi (vedi figura 6.2).

Mentre, analizzando la presenza delle categorie dei place in percentuale rispetto al numero totale dei place in ogni cluster, emerge che 8 cluster appaiono molto diversi dagli altri (vedi figura 6.3).

Selezionando tali cluster, si è cercato di identificarli senza l'ausilio della mappa, ma utilizzando le parole chiavi utilizzate nelle recensioni realizzate dagli utenti nei vari place, quindi estraendo ed analizzando le recensioni attraverso un sistema di Name Entity Recognition emerge in

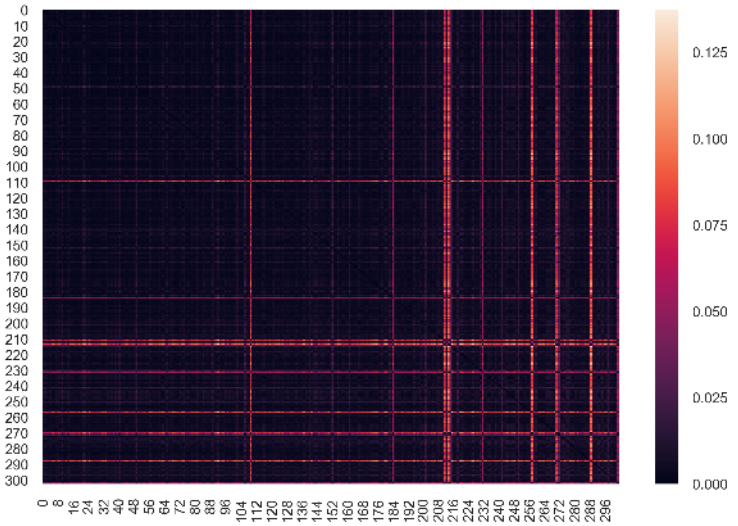


Figura 6.3: Matrice della similarità tra Cluster rispetto alla numerosità della presenza dei Type normalizzata sul numero totale di Place nei Cluster utilizzando la formula di similarità Cosine Distance

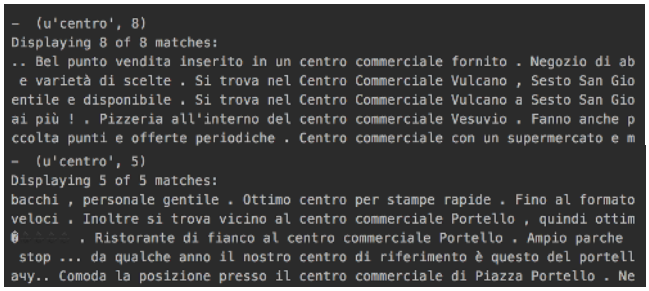


Figura 6.4: Alcuni estratti del termine "Centro" nei cluster identificati

modo significativo che in quasi tutti e 8 i cluster è presente il termine "Centro Commerciale", spesso associato alla zona di appartenenza (i.e. Portello, Arese) oppure al nome del centro (i.e. Vulcano, Il Centro, Centro Sarca, Metropoli), come è possibile osservare in figura 6.4.

Tale analisi non è stata ulteriormente approfondita a causa dei tempi e della mole di lavoro necessari ad avere dati e analisi più accurate. Ad ogni modo questa breve analisi mostra in via preliminare le potenzialità di utilizzare le recensioni dei place come fonte di informazioni per caratterizzare ulteriormente i cluster identificati.

In oltre l'emergere del termine "Centro Commerciale" nella maggior parte dei 8 cluster estratti spinge a pensare che tali cluster siano tutti dei centri commerciali i quali, in termini di copertura di servizi, si differenziano in modo molto più evidente rispetto agli altri cluster che di contro potrebbero essere semplicemente aree urbane con un alto numero di servizi.

6.2 Applicazioni e sviluppi futuri

Il presente lavoro di ricerca si presta a diversi sviluppi futuri che hanno lo scopo di verificare robustezza delle metodologie proposte e confrontare i risultati ottenibili da contesti diversi, ad esempio applicando l'analisi ad altre città come Firenze, Genova, Roma, Torino o Napoli e molte altre, facendo emergere possibili tratti distintivi della distribuzione dei servizi sul territorio.

Una ingegnerizzazione delle componenti software realizzate migliorerebbe la riusabilità, la robustezza, le performance e la modularità, al fine di essere utilizzate in contesti differenti.

A partire dal processo adattivo di acquisizione dei dati, descritto nel capitolo tre, che può essere utilizzato per acquisire Place da Web API di altri Social Media basati su mappe. La stessa cosa accade per il processo di clustering gerarchico-iterativo basato sul DBSCAN che, a seguito di una ingegnerizzazione, può essere applicato a dati spaziali differenti da quelli ottenibili con Google Map.

In oltre, tale processo di clustering gerarchico-iterativo, seppur basato sul DBSCAN, rappresenta un approccio generale che mira a risolvere un popolare problema di alcuni algoritmi di clustering: l'intrinseca difficoltà, o indecidibilità, di selezione dei parametri di input. Infatti, in alcuni algoritmi di clustering, il risultato è fortemente influenzato dai parametri di input e spesso non è possibile conoscere con certezza quale sia la giusta parametrizzazione che permette di ottenere il risultato voluto. Da questo punto di vista, l'approccio sviluppato, oltre che a generare una gerarchia di cluster, permette di identificare il clustering migliore senza conoscerne la parametrizzazione. Dunque sarebbe interessante sperimentare tale approccio generale ad algoritmi diversi dal DBSCAN.

Infine, per poter realizzare tale processo di clustering gerarchico-iterativo basato sul DBSCAN, sono stati introdotti alcuni metodi che guidano le scelte interne dell'algoritmo modificandone il comportamento. Seppur tali metodi siano modificabili, essi sono stati scelti e fissati al fine di rendere il comportamento dell'algoritmo omogeneo rispetto agli obiettivi prefissati. Dunque, uno degli sviluppi futuri è quello di modificare tali metodi e sperimentarne il comportamento del processo sia sulla stessa base di dati che su basi di dati diverse.

Passando a considerare i risultati di questo lavoro, una delle domande che è possibile fare è chiedersi se i dati in merito alle aree con un alta presenza di servizi possono essere utilizzati per determinare la fruibilità di un luogo o, in modo più preciso, il grado di pedonabilità di un luogo.

Considerando il fatto che i place maggiormente presenti sono relativi al commercio ed al cibo, c'è da chiedersi se tale fenomeno sia influenzato

maggiormente dal comportamento degli utenti rispetto alla piattaforma di Google, piuttosto che dall'effettiva caratterizzazione del luogo.

Mentre considerando i cluster emersi in maniera particolarmente evidente nell'ultima analisi che sembrano rappresentare centri commerciali (vedi figura 6.3), c'è da chiedersi se il grado di camminabilità non si colleghi in qualche modo ad essi ed al loro modo di essere progettati. Un'altra interessante domanda è legata al fatto che, tanto i centri commerciali quanto i centri urbani sono stati identificati come cluster, dunque è possibile chiedersi quanto hanno in comune i maggiori centri urbani dai moderni centri commerciali.

In fine, gli sviluppi futuri più immediati che possono essere realizzati sono quelli relativi miglioramento dei dati e dei risultati ottenuti, in particolare a partire dalle criticità emerse nei dati, ovvero che:

- un numero elevato di place non sono associati ad un numero sufficiente di categorie per essere caratterizzati in modo corretto;
- molte tipologie di attività commerciali non sono rappresentate dalle categorie di Google;
- le categorie considerate nel progetto non sono state analizzate in termini semantici.

La criticità che riguarda il fatto che molti place non siano associati a categorie sufficienti per essere caratterizzati, esso può essere risolto con un processo di integrazione dei dati al fine di aggiungere categorie ai place che lo necessitano, ad esempio analizzando il nome, le recensioni o il sito web associato.

Mentre la criticità relativa alla scarsa completezza delle categorie usate da Google può essere risolta effettuando un lavoro tassonomico delle tipologie di attività commerciali nel contesto urbano partendo, ad esempio, da classificazioni internazionali sul commercio.

Infine, l'ultima criticità, quella relativa al significato semantico delle categorie, è risolvibile attraverso un lavoro di modellazione dei dati basata sui linked data, basi di conoscenza aperte ed inferenze sui concetti.

Dunque, sviluppi futuri ed applicazioni si intersecano in diverse domande di ricerca che possono essere risolte solo attraverso del lavoro aggiuntivo che sarebbe interessante affrontare in attività di ricerca future al fine non solo di migliorare i risultati ottenuti, ma fornire un potenziale contributo in contesti differenti da quello sperimentato.

Bibliography

[1] Tariq Ali, Sohail Asghar, and Naseer Ahmed Sajid. Critical analysis of dbscan variations. In *Information and Emerging Technologies (ICIET), 2010 International Conference on*, pages 1–6. IEEE, 2010.

[2] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. In *ACM Sigmod record*, volume 28, pages 49–60. ACM, 1999.

[3] Michael Batty. Big data, smart cities and city planning. *Dialogues in Human Geography*, 3(3):274–279, 2013.

[4] Andrea e Vizzari Giuseppe Berzi, Christian e Gorrini. Mining the social media data for a bottom-up evaluation of walkability. *arXiv preprint arXiv:1712.04309*, 2017.

[5] Nicola Bicocchi, Damiano Fontana, Marco Mamei, and Franco Zambonelli. Collective awareness and action in urban superorganisms. In *Communications Workshops (ICC), 2013 IEEE International Conference on*, pages 194–198. IEEE, 2013.

[6] Paul Brindley, James Goulding, and Max L Wilson. Generating vague neighbourhoods through data mining of passive web data. *International Journal of Geographical Information Science*, 32(3): 498–523, 2018.

[7] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37456-2.

[8] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data*, 10(1):5:1–5:51, July 2015. ISSN 1556-4681. doi: 10.1145/2733381. URL `http://doi.acm.org/10.1145/2733381`.

[9] Michel Marie Deza and Elena Deza. Encyclopedia of distances. In *Encyclopedia of Distances*, pages 1–583. Springer, 2009.

[10] Lian Duan, Lida Xu, Feng Guo, Jun Lee, and Baopin Yan. A local-density based spatial clustering algorithm with noise. *Information systems*, 32(7):978–986, 2007.

[11] Richard C Dubes. How many clusters are best?-an experiment. *Pattern Recognition*, 20(6):645–663, 1987.

[12] Sarah Elwood, Michael F Goodchild, and Daniel Z Sui. Researching volunteered geographic information: Spatial data, geographic research, and new social practice. *Annals of the association of American geographers*, 102(3):571–590, 2012.

[13] Martin Ester. *Density-based Clustering*, pages 795–799. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_605. URL https://doi.org/10.1007/978-0-387-39940-9_605.

[14] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[15] Vladimir Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75, 2002.

[16] BS Everitt, S Landau, M Leese, and D Stahl. *Cluster analysis: Wiley series in probability and statistics*. Wiley Chichester, 2011.

[17] Damiano Fontana and Franco Zambonelli. Towards an infrastructure for urban superorganisms: challenges and architecture. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 390–393. IEEE, 2012.

[18] Marcus Foth, Jaz Hee-jeong Choi, and Christine Satchell. Urban informatics. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 1–8. ACM, 2011.

[19] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*, volume 20. Siam, 2007.

[20] Michael F. Goodchild. A geographer looks at spatial information theory. In Daniel R. Montello, editor, *Spatial Information Theory*, pages 1–13, Berlin, Heidelberg, 2001. Springer, Springer Berlin Heidelberg. ISBN 978-3-540-45424-3.

[21] Michael F Goodchild. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4):211–221, 2007.

[22] Michael F. Goodchild. *Geographic Information System*, pages 1231–1236. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_178. URL https://doi.org/10.1007/978-0-387-39940-9_178.

[23] Allan D Gordon. A review of hierarchical classification. *Journal of the Royal Statistical Society. Series A (General)*, pages 119–137, 1987.

[24] Viola Gorrini, Andrea e Bertini. Walkability assessment and tourism cities: the case of venice. *International Journal of Tourism Cities*, 2018.

[25] Ralf Hartmut Güting. An introduction to spatial database systems. *The VLDB Journal—The International Journal on Very Large Data Bases*, 3(4):357–399, 1994.

[26] Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4): 12–18, 2008.

[27] Thomas C Hales. The honeycomb conjecture. *Discrete & Computational Geometry*, 25(1):1–22, 2001.

[28] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[29] Jordan T. Hastings and Linda L. Hill. *Georeferencing*, pages 1246–1249. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_181. URL https://doi.org/10.1007/978-0-387-39940-9_181.

[30] Linda L Hastings, Jordan T e Hill. Georeferencing. In *Encyclopedia of Database Systems*, pages 1246–1249. Springer, 2009.

[31] Alexander Hinneburg and Daniel A Keim. A general approach to clustering in large databases with noise. *Knowledge and Information Systems*, 5(4):387–415, 2003.

[32] Song Hu, Yingjie e Gao, Bailang Janowicz, Krzysztof e Yu, Wenwen Li, and Sathya Prasad. Extracting and understanding urban areas of interest using geotagged photos. *Computers, Environment and Urban Systems*, 54:240–254, 2015.

[33] Jane Jacobs. *The death and life of American cities*. 1961.

[34] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3): 264–323, 1999.

[35] Mark Graham Joe Shaw. Our digital rights to the city. 2017.

[36] Bays Jonathan and Callanan Laura. 'urban informatics' can help cities run more efficiently. 2012.

[37] Christopher B. Jones and Ross S. Purves. Geographical information retrieval. *International Journal of Geographical Information Science*, 22(3):219–228, 2008. doi: 10.1080/13658810701626343. URL https://doi.org/10.1080/13658810701626343.

[38] Christopher B Jones and Ross S Purves. Geographical information retrieval. In *Encyclopedia of Database Systems*, pages 1227–1231. Springer, 2009.

[39] Christopher B Jones, Ross S Purves, Paul D Clough, and Hideo Joho. Modelling vague places with knowledge from the web. *International Journal of Geographical Information Science*, 22(10): 1045–1065, 2008.

[40] Amin Karami and Ronnie Johansson. Choosing dbscan parameters automatically using differential evolution. *International Journal of Computer Applications*, 91(7), 2014.

[41] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[42] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and Sababady Sarasvady. Dbscan: Past, present and future. In *Applications of Digital Information and Web Technologies (ICADIWT), 2014 Fifth International Conference on the*, pages 232–238. IEEE, 2014.

[43] Tim Kindberg, Matthew Chalmers, and Eric Paulos. Guest editors' introduction: Urban computing. *IEEE Pervasive Computing*, 6(3): 18–20, 2007.

[44] Slava Kisilevich, Florian Mansmann, and Daniel Keim. P-dbscan: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In *Proceedings of the 1st international conference and exhibition on computing for geospatial research & application*, page 38. ACM, 2010.

[45] Erica Kolatch et al. Clustering algorithms for spatial databases: A survey. *PDF is available on the Web*, pages 1–22, 2001.

[46] Vassilis Kostakos, Tom Nicolai, Eiko Yoneki, Eamonn O'Neill, Holger Kenn, and Jon Crowcroft. Understanding and measuring the urban pervasive infrastructure. *Personal and Ubiquitous Computing*, 13(5):355–364, 2009.

[47] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011. doi: 10.1002/widm.30. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.30`.

[48] John Krumm, Nigel Davies, and Chandra Narayanaswami. User-generated content. *IEEE Pervasive Computing*, 7(4):10–11, 2008.

[49] Vipin Kumar. *Data Mining and Knowledge Discovery Series*. Chapman & Hall/CRC, 2014.

[50] Pierre Legendre and Loic FJ Legendre. *Numerical ecology*, volume 24. Elsevier, 2012.

[51] Peng Liu, Dong Zhou, and Naijun Wu. Vdbscan: varied density based spatial clustering of applications with noise. In *Service Systems and Service Management, 2007 International Conference on*, pages 1–4. IEEE, 2007.

[52] Jeremy Mennis and Diansheng Guo. Spatial data mining and geographic knowledge discovery—an introduction. *Computers, Environment and Urban Systems*, 33(6):403–408, 2009.

[53] Boris Mirkin. *Clustering for data mining: a data recovery approach*. Chapman and Hall/CRC, 2005.

[54] Daniel R Montello, Michael F Goodchild, Jonathon Gottsegen, and Peter Fohl. Where's downtown?: Behavioral methods for determining referents of vague spatial queries. *Spatial Cognition & Computation*, 3(2-3):185–204, 2003.

[55] Fionn Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983.

[56] Eamonn O'Neill, Vassilis Kostakos, Tim Kindberg, Alan Penn, Danaë Stanton Fraser, Tim Jones, et al. Instrumenting the city: Developing methods for observing and understanding the digital cityscape. In *International Conference on Ubiquitous Computing*, pages 315–332. Springer, 2006.

[57] Luca Maria e Schifanella Rossano e Davies Adam Quercia, Daniele e Aiello. The digital life of walkable streets. In *Proceedings of the 24th International Conference on World Wide Web*, pages 875–884. International World Wide Web Conferences Steering Committee, 2015.

[58] Egidio Riva and Mario Lucchini. La natalità delle imprese straniere a milano: un'analisi spaziale. *IMPRESE & CITTÀ*, 5, 2014.

[59] Alessandro Rodriguez, Alex e Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.

[60] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[61] Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data mining and knowledge discovery*, 2(2):169–194, 1998.

[62] Shashi Shekhar, Pusheng Zhang, Yan Huang, and Ranga Raju Vatsavai. Spatial data mining. 2003.

[63] Abir Smiti and Zied Elouedi. Dbscan-gm: An improved clustering method based on gaussian means and dbscan techniques. In *Intelligent Engineering Systems (INES), 2012 IEEE 16th International Conference on*, pages 573–578. IEEE, 2012.

[64] Jeff Speck. *Walkable city: How downtown can save America, one step at a time*. Macmillan, 2013.

[65] Michael Steinbach, Levent Ertöz, and Vipin Kumar. *The challenges of clustering high dimensional data*, pages 273–309. Springer, 2004.

[66] Patrizia Sulis, Ed Manley, Chen Zhong, and Michael Batty. Using mobility data as proxy for measuring urban vitality. *Journal of Spatial Information Science*, 16:137–162, 2018.

[67] Piyushimita Vonu Thakuriah, Nebiyou Y Tilahun, and Moira Zellner. Big data and urban informatics: innovations and challenges to urban planning and knowledge discovery. pages 11–45, 2017.

[68] Thanh N Tran, Klaudia Drab, and Michal Daszykowski. Revised dbscan algorithm to cluster data with dense adjacent clusters. *Chemometrics and Intelligent Laboratory Systems*, 120:92–96, 2013.

[69] Anthony KH Tung, Jean Hou, and Jiawei Han. Spatial clustering in the presence of obstacles. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 359–367. IEEE, 2001.

[70] Ozge Uncu, William A Gruver, Dilip B Kotak, Dorian Sabaz, Zafeer Alibhai, and Colin Ng. Gridbscan: Grid density-based spatial clustering of applications with noise. In *Systems, Man and Cybernetics, 2006. SMC'06. IEEE International Conference on*, volume 4, pages 2976–2981. IEEE, 2006.

[71] Shuliang Wang and Hanning Yuan. Spatial Data Mining. *International Journal of Data Warehousing and Mining*, 2014. ISSN 1548-3924. doi: 10.4018/ijdwm.2014100103.

[72] D. Wishert. Mode analysis : a generalization of nearest neighbour which reduces chaining effects (with discussion). *Numerical Taxonomy*, pages 282–311, 1969. URL https://ci.nii.ac.jp/naid/10012395375/en/.

[73] Chen Xiaoyun, Min Yufang, Zhao Yan, and Wang Ping. Gmdbscan: multi-density dbscan cluster based on grid. In *e-Business Engineering, 2008. ICEBE'08. IEEE International Conference on*, pages 780–783. IEEE, 2008.

[74] Fatma Günseli Yaşar and Gözde Ulutagay. Challenges and possible solutions to density based clustering. In *Intelligent Systems (IS), 2016 IEEE 8th International Conference on*, pages 492–498. IEEE.

[75] Osmar R Zaïane and Chi-Hoon Lee. Clustering spatial data when facing physical constraints. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 737–740. IEEE, 2002.

[76] Franco Zambonelli. Toward sociotechnical urban superorganisms. *Computer*, 45(8):76–78, 2012.

[77] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intell. Syst. Technol.*, 5(3):38:1–38:55, September 2014. ISSN 2157-6904. doi: 10.1145/2629592. URL http://doi.acm.org/10.1145/2629592.

Acronimi

AOI Area of interest . 37, 38, 156, 157, *Glossary:* area di interesse

API Application Program Interface. 77, 78, 80, 84, 88, 94–96, 102

DBSCAN Density-Based Spatial Clustering of Applications with Noise. 7, 38, 41, 51, 55–58, 67–70, 109–113, 117, 119, 122, 124, 129, 130, 138, 140, 189

DENCLUE DENsity-based CLUstEring. 59

EOM Eccesso di Massa, in inglese Excess Of Mass. 8, 70

GIS Geographic Information System. 31–33, 77

GMDBSCAN Multy Density DBSCAN Cluster Based on Grid. 69

GPS Global Positioning System. 28, 81

GRIDBSCAN GRIdDensity-Based Spatial Clustering of Applications with Noise. 68

HDBSCAN Hierarchical DBSCAN. 69, 70

HDBSCAN-EOM Hierarchical DBSCAN with Excess Of Mass. 69, 70

HTTP HyperText Transfer Protocol. 80, 84

JSON JavaScript Object Notation. 80–82, 84, 86, 99

KDE Kernel Density Estimation. 7, 39, 59, 60

KPI Key Performance Indicator. 119, 123, 140

NIL Nuclei d'Identità Locale. 7, 36, 37

OPTICS Ordering Points To Identify the Cluster Structure. 8, 58, 59, 69

PGT Piano di Governo del Territorio. 7, 36, 37

POI Point of Interest. 106

SIT Sistemi Informativi Territoriali. 32

UGC User Generated Content . 24, 27, 28, 30, 37–40, *Glossary:* contenuto generati dall'utente

UPI Urban Pervasive Infrastructure. 18

URL Uniform Resource Locator. 80, 81, 84

VGI Volunteered geographic information. 29–32, 77

WebGIS Web Geographic Information System. 31–33

WMS Web Map Service. 32

XML eXtensible Markap Language. 80