Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica per il Management

Internet of Things collaborativo: progettazione ed analisi di una piattaforma di aggregazione di dati sensoristici

Relatore:
Chiar.mo Prof.
Marco Di Felice

Presentata da:
Silvia Perrino

Correlatore:
Dott. Federico Montori

Sessione I
Anno Accademico 2015/2016

"We are drowning in information but starved for knowledge."
(John Naisbitt)

Introduzione

Internet ha rivoluzionato il modo in cui le persone comunicano e lavorano insieme. Siamo stati testimoni della nascita e dello sviluppo di una nuova era caratterizzata dalla disponibilità di informazione libera e accessibile a tutti.
Negli anni recenti tramite la diffusione di smartphone e tablet e con l'aumentare di diverse tipologie di dispositivi connessi ad Internet, è cambiato il fulcro dell'innovazione spostandosi dalle persone alle "cose". Sempre più dispositivi sono connessi con la possibilità di interagire e scambiarsi informazioni nei diversi ambiti applicativi: un esempio rappresentativo è quello della domotica, ambito in cui i diversi sistemi presenti nelle case come gli elettrodomestici possono interagire e connettersi ad Internet, con l'obiettivo di creare ambienti intelligenti e capaci di gestirsi in autonomia.
È così che nasce l'Internet of Things, letteralmente Internet delle cose, termine coniato nel 1999 da un imprenditore inglese, Kevin Ashton [2] e usato per descrivere la rete di comunicazione creata tra i diversi devices connessi ad internet capaci di interagire tra loro in autonomia.

Come illustrato nel primo capitolo gli ambiti applicativi dell'Internet of Things variano dalla domotica, alla sanità, alla realizzazione di smart cities e così via; l'obiettivo principale di tale disciplina è quello di migliorare la vita delle persone grazie a sistemi che siano in grado di interagire tra loro senza bisogno dell'intervento dell'essere umano, addirittura cercando di anticipare i loro comportamenti.
Proprio per la natura eterogenea della disciplina e in relazione ai diversi ambiti applicativi, nell'IoT è facile sviluppare soluzioni e ambienti eterogenei sia in termini di tecnologie che di modalità con cui memorizzare i dati. Questo porta alla presenza di

network disconnessi, nei quali è possibile comunicare con sistemi omogenei ma risulta difficile integrare informazioni provenienti da fonti differenti. Nel primo capitolo viene introdotto il concetto di Internet of Things Collaborativo, termine che indica l'obiettivo di realizzare applicazioni che possano garantire interoperabilità tra i componenti nei vari ecosistemi e tra le diverse fonti da cui l'IoT attinge sfruttando la presenza di piattaforme di pubblicazione di open data.

L'obiettivo di questa tesi è stato quello di creare un sistema per l'aggregazione di dati da due piattaforme di open data sensoristici eterogenee, Thingspeak[1] e Sparkfun[2], unificandoli in un database comune come illustrato nel capitolo 2. Tali dati rappresentano le varie misurazioni fatte dagli utenti con i propri sensori in diverse parti del mondo, caricate online per essere condivise. Le rilevazioni fatte vanno da misurazioni di temperatura, pressione, umidità e così via. A tal proposito con i dati estratti sono state sperimentate due tecniche note di Data Mining spiegate nel dettaglio nel capitolo 3, con lo scopo di estrarre informazioni significative che potessero caratterizzare i suddetti dati. Infatti per come sono presentati sulle piattaforme, i dati non hanno un significato generale e condivisibile da tutti, risultando spesso di dubbia interpretabilità. Lo scopo del progetto è dunque quello di realizzare un sistema che sia in grado di dare un significato specifico ad ogni dato processato nel contesto dei dati sensoristici, utilizzando le informazioni associate ai dati stessi e accomunando i dati che hanno caratteristiche comuni.
Benchè il set di dati estratti sia relativamente limitato in confronto al numero totale di informazioni presenti su ThingSpeak e Sparkfun, con questo progetto è stata avanzata un'analisi che ha individuato le più frequenti misurazioni fatte dagli utenti; di conseguenza ci si immagina di poter applicare le tecniche implementate anche alla restante porzione di dati ottenendo dei risultati relativamente soddisfacenti.
Riguardo ai risultati ottenuti, sono mostrati nel capitolo 4: ciascuna metodologia è stata analizzata e sono state calcolate delle metriche di performance su cui basare le considerazioni finali.

[1] https://thingspeak.com/
[2] https://data.sparkfun.com/

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Internet of Things

L'Internet of Things (IoT) può essere concettualmente definito come una dinamica e globale infrastruttura di rete caratterizzata dalla capacità di auto-configurarsi, basato su protocolli di comunicazione standard ed interoperabili dove gli oggetti fisici e virtuali hanno una propria identità, dei propri attributi fisici e personalità virtuali e sono integrati nella rete informativa, senza soluzione di continuità.

Nell'ambito dell'IoT ci si aspetta dunque che gli "oggetti" siano sempre più coinvolti attivamente nei processi a cui si rivolgono, dal business ai processi informativi e sociali nei quali essi sono tenuti ad interagire e comunicare tra loro e con l'ambiente di riferimento, scambiandosi dati ed informazioni, reagendo autonomamente agli input del "mondo reale" con o senza l'intervento degli esseri umani [20].

La rilevanza di tale disciplina è sempre più evidente e aumenta man mano che si iniziano a vedere praticamente le opportunità che offre per la vita di tutti i giorni. Non a caso Gartner[1] ha inserito l'IoT tra le "Top Ten Strategic Technologies" negli anni dal 2012 al 2016, stimando che nel 2020 gli oggetti connessi nel mondo saranno 20.8 miliardi [1].

[1] http://www.gartner.com/technology/home.jsp

Pervasività e diffusione dei dispositivi IoT

Secondo la Cisco Internet Business Solutions Group (IBSG)[2] l'IoT si posiziona in un periodo di tempo in cui sono connessi ad Internet più oggetti e dispositivi che persone. A questo proposito si veda la figura 1.1 in cui vengono mostrate le previsioni, sempre fonte CISCO, del numero di dispositivi per persona che saranno connessi ad Internet entro il 2020, in relazione alla popolazione mondiale e al numero di dispositivi connessi totali [8].

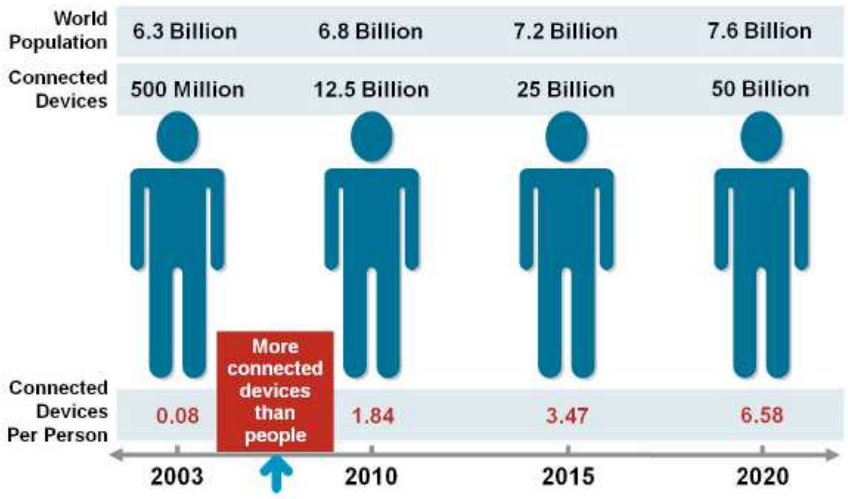


Figura 1.1: Dispositivi connessi ad Internet entro il 2020 [8]

Come si nota dalla figura 1.1 fino al 2003 i dispositivi connessi erano meno di uno per individuo, periodo in cui non esisteva ancora praticamente l'IoT. Con l'aumentare della diffusione di dispositivi mobili indicativamente a partire dal 2007 (lancio del primo iPhone) e in relazione all'aumento della popolazione mondiale, è aumentato notevolmente il numero di dispositivi connessi ad Internet e lo sviluppo dell'IoT si posiziona proprio in questo periodo temporale. Cisco IBSG stima la nascita dell'IoT indicativamente tra il 2008 e il 2009 [8].

[2]http://www.cisco.com/c/en/us/about.html

Tuttavia le stime riguardanti l'aumento dei dispositivi connessi in fututo non tiene conto della rapidità con cui le tecnologie si sviluppano; i numeri mostrati si basano sulle attuali conoscenze presenti. Si può dunque immaginare che tali dati possano risultare maggiori in stime successive future.

1.1 Applicazioni dell'IoT

La sfida dell'IoT consiste nel creare ambienti "intelligenti" e oggetti "self-aware", cioè capaci di reagire consapevolmente agli input percepiti, negli ambiti della domotica, del clima, dell'energia, della mobilità, della società digitale e ambientale [20].
Il trend da seguire per compiere tale impresa quindi è quello di estendere la diffusione della connessione ad Internet agli oggetti quotidiani, oltre che ai pc e ai dispositivi mobili.

Tra gli ambiti applicativi dell'IoT troviamo:

- Home automation o Smart Home
- Smart City
- Healthcare
- Business e Commercio
- Environmental monitoring
- Sport

Smart Home

L'IoT nell'ambito della domotica trova numerose possibilità di azione. Ci si immagina di poter costruire le "case del futuro", in cui ogni componente o elettrodomestico sarà in grado di auto gestirsi e comunicare indipendentemente con altri dispositivi.
Grazie alla diffusione dell'IoT questo scenario ha smesso di essere pura fantascienza, permettendo a chiunque disponga dei mezzi e delle capacità necessarie, di realizzare sistemi per l'*home automation*. Ad esempio avendo a disposizione dei sensori che rilevano la

temperatura o la luminosità all'interno di una stanza, si fa in modo che le finestre si chiudano o si aprano da sole senza l'intervento delle persone.

In Italia sono diverse le realtà presenti nell'ambito della domotica e dell'IoT, tra le quali Morpheos[3], una consolidata startup siciliana che prevede di lanciare sul mercato internazionale un robot domestico chiamato "Momo" entro il 2017[4]. Tale dispositivo sarà dotato di sensori per il controllo dell'ambiente e attuatori (Led, altoparlanti, connessioni wireless, ecc.) gestiti da un sistema di intelligenza artificiale.

La figura 1.2 mostra le previsioni riguardo al numero di installazioni smart home in Nord America e in Europa entro il 2020 [10].

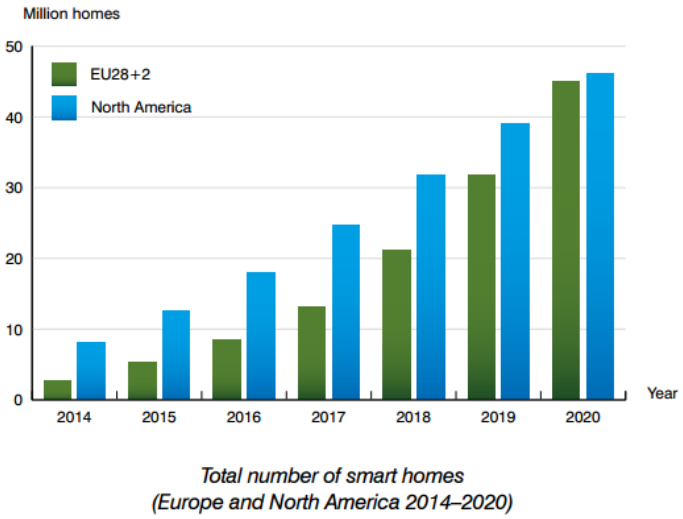


Figura 1.2: Smart Home entro il 2020 [10]

[3]http://www.morphemos.eu

[4]https://www.key4biz.it/smart-home-entro-un-anno-sul-mercato-il-robot-domestico-realizzato-in-sicilia/163517

Smart Cities

La visione di "città intelligenti" mira a sfruttare le più avanzate tecnologie di comunicazione presenti per sostenere e fornire servizi che abbiano un valore aggiunto negli ambiti amministrativi delle città e siano di supporto ai cittadini [21]. Per tale scopo risultano fondamentali dei dati che siano liberi e disponibili a chiunque volesse progettare sistemi per avvicinarsi alle "Smart Cities", tipicamente dati riguardanti le infrastrutture, dati ambientali, dati in relazione alla mobilità ecc.
A questo scopo alcune istituzioni forniscono tali dati in formato open online, come ad esempio il Comune di Bologna, il quale dispone di un dataset pubblico composto da 1493 file in formato open[5].

Healthcare

Grazie alle potenzialità nell'ambito dell'IoT si può significativamente migliorare la qualità della vita degli individui; questo è infatti uno degli scopi principali della disciplina.
È possibile estendere l'applicazione dell'Iot in ambito sanitario ad esempio con lo scopo di monitorare lo stato di salute dei pazienti in un determinato contesto, rilevarne le informazioni relative e adottare misure adeguate e migliorare l'assistenza.
Ad esempio si possono rilevare i dati di un elettrocardiogramma, monitorare la temperatura o i livelli di glucosio nel sangue con l'obiettivo di tracciare le informazioni per i diversi pazienti. Ovviamente benchè ne consegua un intervento da parte di personale qualificato, ciò fa si che dei dispositivi possano diffondere dati utili nello specifico contesto, comportando una riduzione dell'interazione diretta tra medico e paziente[6].

Business e Commercio

Uno degli aspetti su cui l'IoT avrà maggior impatto sarà sicuramente l'e-commerce. I dispositivi connessi, dalle macchine agli elettrodomestici, cambieranno le modalità con cui avviene lo shopping online trasformando gli oggetti dell'IoT in potenziali canali di

[5] http://dati.comune.bologna.it

[6] http://internetofthingsagenda.techtarget.com/feature/Can-we-expect-the-Internet-of-Things-in-healthcare

vendita per i commercianti; un esempio è il caso dello smart fridge, il quale può "comprare da solo" online un prodotto se questo è terminato.
D'altra parte i dispositivi IoT aiuterranno l'e-commerce nell'ottimizzazione delle operazioni relative al business. Ad esempio gestire l'inventario sarà più facile grazie alla possibilità di individuare i movimenti dei prodotti tracciabili in tempo reale. Le informazioni così prodotte possono essere usate per notificare i proprietari del business dei movimenti delle merci[7].
Un altro aspetto riguarda il marketing: grazie alla presenza di dispositivi connessi, le aziende hanno la possibilità di ottenere sempre più informazioni riguardo ai bisogni e alle abitudini dei consumatori; questi dati possono essere tracciati appunto dal possesso di devices. Dal lato dei consumatori invece, questo si traduce in offerte di prodotti e servizi sempre più personalizzati.

Environmental monitoring

Il monitoraggio dei dati ambientali quali livelli di umidità, di temperatura e così via risultano una soluzione utile in diversi contesti, uno dei quali è il trasporto di merci deperibili. Ad esempio la realizzazione di sistemi che permettono la misurazione e il controllo di tali informazioni dal distributore al consumatore consentono di evitare scarsi livelli di qualità delle merci, migliorando l'efficienza della catena di distribuzione [3].

Sport

L'IoT ha cambiato inoltre il modo in cui le persone fanno esercizio fisico. La diffusione di dispositivi capaci di effettuare misurazioni sul battito cardiaco o che ad esempio permettono di contare il numero di passi, fungono da personali istruttori capaci di monitorare e motivare lo sportivo. I cosiddetti "wearable" ovvero i dispositivi indossabili dotati di sensori hanno riscosso successo negli ultimi anni tanto da essere prodotti da diversi brand.

[7] http://www.itdonut.co.uk/blog/2016/04/internet-things-and-its-influence-ecommerce-0

1.2 IoT collaborativo

Sono presenti alcuni limiti nello sviluppo di applicazioni e tecnologie nell'ambito dell'IoT. Uno di questi limiti è la mancanza di possibilità da parte di un soggetto di produrre da sè i dati necessari per lo sviluppo di tali applicazioni, che in qualche modo rappresentano una delle caratteristiche fondamentali dell'IoT. In alcuni casi dunque potrebbe non essere facile procurasi da sè le tecnologie adeguate, ad esempio nel caso di sensori troppo costosi o che presentano limiti fisici. Per sopperire a tale mancanza l'IoT collaborativo è la soluzione ideale che permette di facilitare l'accesso ai dati necessari [11].
A questo proposito assumono rilevanza fondamentale le piattaforme online di pubblicazione e condivisione di dati "liberi", spiegate nella sezione seguente.

1.2.1 Open data platforms

Nell'ambito dell'IoT si ha a che fare con una sempre più vasta mole di dati, risultante appunto dalla varietà dei campi applicativi della disciplina. Il volume, la velocità e la volatilità dei dati prodotti rende l'elaborazione, l'integrazione e l'interpretazione degli stessi come una sfida significativa. Inoltre il volume di tali dati continua a crescere ad una velocità sorprendente [4].
Mentre le "passate" tecnologie in ambito IT venivano sviluppate principalmente per specifici scopi con flessibilità limitata, le iniziative dell'IoT necessitano di servizi e piattaforme che siano in grado di catturare, comunicare, storicizzare, accedere e condividere dati dal mondo fisico [4].
Si possono distinguere diverse tipologie di piattaforme di condivisione di open data in relazione al tipo di dati che intendono diffondere; tuttavia lo scopo di tali data-store resta in ogni caso quello di condividere informazioni utili a tutti gli utenti.
Due tra le varietà di piattaforme di pubblicazione di dati online possono essere le seguenti:

- piattaforme di condivisione di dati prodotti da utenti privati tramite i loro sensori;
- piattaforme di condivisione di dati istituzionali provenienti da enti pubblici;

Dati sensoristici

Due tra le piattaforme di condivisione di dati sensoristici sono nel dettaglio analizzate in questa tesi. Si tratta di ThingSpeak[8] e Sparkfun[9], entrambe permettono la storicizzazione e la condivisione di dati prodotti dagli utenti con i propri dispositivi e sensori in formato open. Sistemi come Arduino[10] o Raspberry Pi[11] grazie alla loro facilità di utilizzo fanno si che sempre più dati vengano prodotti e le suddette piattaforme facilitano la gestione di tali dati grazie anche alla possibilità di generare grafici e statistiche.

Nella figura 1.3 viene mostrato un grafico prodotto in ThingSpeak il quale mostra le statistiche relative alle misurazioni caricate da un utente per un certo valore, in questo caso riguardo la temperatura nel tempo all'interno di una stanza.

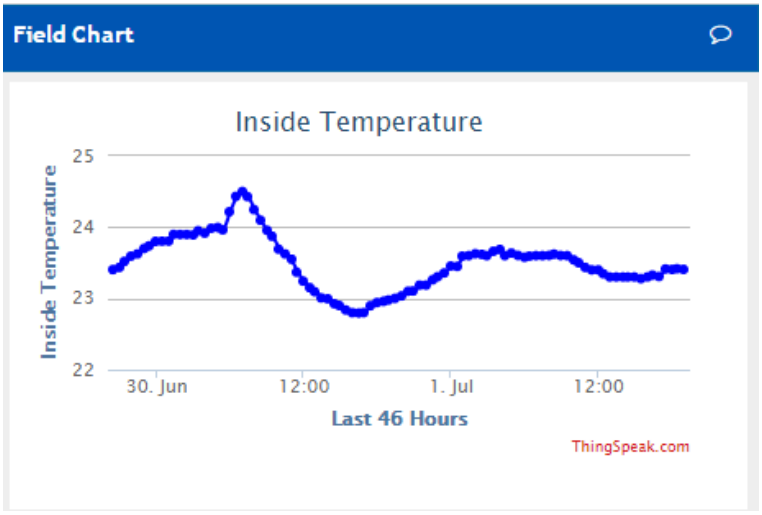


Figura 1.3: Esempio di statistica prodotta da ThingSpeak

[8]https://thingspeak.com
[9]https://data.sparkfun.com
[10]https://www.arduino.cc
[11]https://www.raspberrypi.org

Dati istituzionali

Come già accennato anche diversi enti pubblici forniscono in formato open i dati relativi alla mobilità, alle infrastrutture, alla sanità ecc. A questo proposito è necessario evidenziare l'importanza e il duplice scopo di tale diffusione: se da un lato i dati diffusi dagli enti pubblici possono rendersi utili ai fini della progettazione di applicazioni dell'IoT come ad esempio la realizzazione di Smart Cities e di sistemi intelligenti, tali dati rendono inoltre possibile la trasparenza amministrativa, caratteristica da non sottovalutare e di particolare interesse per i cittadini.
Si veda ad esempio DatiOpen.it[12], un portale nato ad agosto del 2012 che attualmente ha nel suo "catalogo" oltre 650 dei principali open data italiani scaricabili in formati liberi; o ancora l'Arpae Emilia-Romagna[13] che fornisce un dataset pubblico di informazioni concernenti l'organizzazione e l'attività delle pubbliche amministrazioni, tra cui informazioni ambientali, sulle strutture sanitarie, sugli immobili e la gestione del patrimonio ecc.

Nel dettaglio è utile specificare la distinzione tra *Linked Data* e *Open Data*, che convergendo formano i **Linked Open Data**.

- **Linked Data**: il termine si riferisce all'insieme di pratiche per la pubblicazione e la correlazione di dati strutturati sul Web. Si intende creare collegamenti ovvero *links* tra dati provenienti da fonti differenti [5]. Per la natura dei Linked Data, essi non devono necessariamente essere *open*, ovvero disponibili sotto licenze libere. Possono quindi anche essere disponibili in alcuni contesti interni; un importante uso dei Linked Data si ha appunto a livello personale o di gruppo[14].

- **Open Data**: ci si riferisce a quei dati pubblicati sottoforma di licenze libere, per cui è possibile rendere tali dati e le relative informazioni di pubblico dominio.

- **Linked Open Data**: sono quei Linked Data rilasciati sotto licenze open rendendone possibile il libero riuso.

[12] http://www.datiopen.it
[13] http://www.arpae.it
[14] https://www.w3.org/DesignIssues/LinkedData.html

È stato realizzato da Tim Berners-Lee[15] uno schema di valutazione per i Linked Open Data chiamato **Modello a 5 stelle**[16], il quale permette di assegnare la qualifica di Linked Open Data ai dati in questione, se sussistono le seguenti caratteristiche[17]:

- **1 stella**, se sono disponibili sul Web in qualsiasi formato, in modo da essere definiti come Open Data;
- **2 stelle**, se i dati sono strutturati in modo da essere interpretabili dagli elaboratori, cioè *machine-readable*;
- **3 stelle**, se sono machine-readable e disponibili sul Web in formati non proprietari;
- **4 stelle**, se sono presenti i punti precedenti e in più vengono usati gli standard del W3C[18] come RDF[19] e SPARQL[20] per identificare le risorse;
- **5 stelle**, se sono presenti i punti precedenti e in più i dati sono collegati ad altri già presenti sul Web.

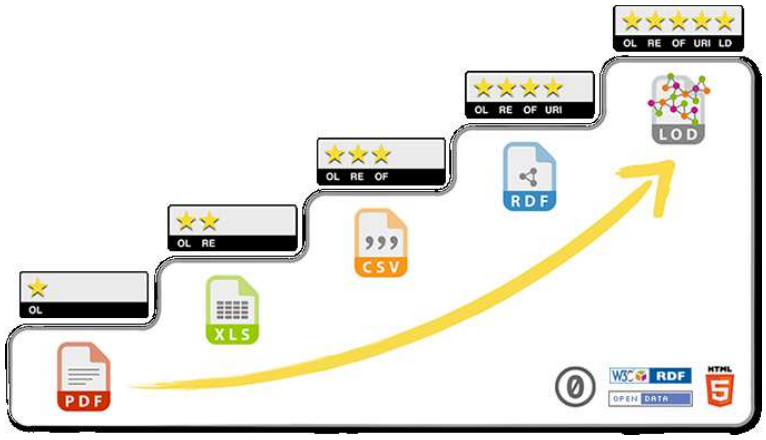


Figura 1.4: Modello 5 stelle per gli Open Data - 5stardata.info

[15]https://www.w3.org/People/Berners-Lee/
[16]http://5stardata.info/en/
[17]https://www.w3.org/DesignIssues/LinkedData.html
[18]https://www.w3.org
[19]https://www.w3.org/RDF
[20]https://www.w3.org/TR/rdf-sparql-query

1.3 Web of Things e Semantic Web nell'IoT

Affinchè si possano realizzare applicazioni che abbiano successo nell'ambito dell'Internet of Things è necessario riferirsi ad un protocollo unico ed affidabile per tutti i dispositivi eterogenei connessi. Tale protocollo deve rispondere a determinate caratteristiche:

- semplicità;
- overhead relativamente basso;
- scalabilità;
- flessibilità;

1.3.1 Web of Things

Si parla di Web of Things poichè le suddette caratteristiche sono attualmente presenti tra i vari protocolli e tecnologie disponibili e ampiamente diffuse nel web, tra cui HTTP, TCP, XML, JSON, RSS, ATOM, REST, URI ecc.. In particolare, si utilizzano gli URI per identificare univocamente gli oggetti, mentre XML e REST permettono agli oggetti di esporre le proprie caratteristiche e di comunicare con servizi esterni [14].

1.3.2 Semantic Web

Le tecnologie sviluppate nel Semantic Web, quali ontologie, annotazioni semantiche, Linked Data e Web services semantici, possono essere sfruttate come principale soluzione per gli scopi dell'IoT. Ad esempio definire un'ontologia e usare una descrizione semantica per i dati rende l'applicazione interoperabile per gli utenti o per chi altro avesse interesse ad utilizzare la stessa ontologia [4].

È possibile fondere le tecnologie del Semantic Web all'Internet of Things, trattando gli oggetti come risorse e sfruttando le ontologie per la gestione della conoscenza, in altre parole, dei dati prodotti dagli oggetti [14].

Le tecnologie del Semantic Web basate sulle rappresentazioni “machine-interpretable” hanno mostrato tecniche per la descrizione degli oggetti, la condivisione e l’integrazione di informazioni e la deduzione di nuova conoscenza insieme ad altre tecniche di processazione di intelligenza. La fusione con l’area semantica ha inoltre aiutato nella creazione di dati interpretabili dalle macchine e auto-descrittivi nell’ambito dell’IoT [4].

A questo proposito è necessario fornire “interoperabilità semantica”, permettendo ai soggetti interessati di accedere ed interpretare i dati in maniera non ambigua. Gli oggetti dell’IoT hanno bisogno di scambiare dati tra loro e con altri utenti in Internet. Fornire una descrizione dei dati non ambigua, in maniera da poter essere processata ed interpretata dalle macchine e dai software agent è la chiave per facilitare la comunicazione automatizzata di informazioni e l’interazione nell’IoT. L’annotazione semantica dei dati fornisce una descrizione interpretabile dalle macchine su ciò che il dato rappresenta, da dove proviene, come può essere correlato alle proprie circostanze, chi lo fornisce e quali sono gli attributi del dato [4].

Capitolo 2

Data retrieval

Con il termine "Data retrieval" si intende il processo di identificazione ed estrazione di dati da un database tramite query.
Ai fini di questo lavoro, come accennato in precedenza, sono state considerate due piattaforme di pubblicazione e condivisione di dati sensoristici, ThingSpeak e Sparkfun. L'obiettivo perseguito è stato quello di realizzare un data-store omogeneo nel quale raccogliere i dati provenienti da entrambe le piattaforme, uniformarli ed estrarne delle informazioni significative [11].

In particolare sono stati estratti 300 *stream/channel* da ThingSpeak e Sparkfun e sono stati analizzati manualmente per dedurre le 32 classi significative su cui basare la classificazione. Ciascuno stream è stato esaminato nel dettaglio e a seconda di tutti i campi misurati, a ciascuno è stata attribuita un'etichetta corrispondente alla classe di appartenenza. In alcuni casi sono state utili altre informazioni dello stream come nome e descrizione per attribuire significato ai valori misurati. Tramite le tecniche di classificazione e di clustering spiegate nel capitolo successivo, si intende assegnare ai dati presenti sulle due piattaforme delle informazioni riguardanti il tipo di misurazioni effettuate dagli utenti.
Grazie agli algoritmi implementati si intende automatizzare la deduzione di tali informazioni, con l'intento di estendere tale processo anche ai dati che non sono stati analizzati in questa sede.

2.1 Funzionalità dell'applicazione

È stato realizzato un sistema che permette di estrarre i dati caricati dagli utenti sulle due piattaforme, classificarli ed estrarre dei cluster da tali dati, implementando e/o adattando due diverse tecniche di Data Mining.
Da ThingSpeak sono stati estratti i dati necessari tramite richieste HTTP per ciascun insieme di dati caricato dagli utenti, mentre per Sparkfun è stato necessario effettuare in partenza uno scraping della pagina web in cui sono elencati i gruppi di dati caricati da ciascun utente, estraendone per ciascuno l'id e in seguito sono state effettuate tutte le richieste HTTP necessarie (sezione 2.2).
Tali informazioni vengono salvate tramite uno script Python in un database locale, nel quale vengono memorizzate le informazioni sui dati e i metadati delle misurazioni fatte dagli utenti (sezione 2.2.1).

2.2 Realizzazione di un data store omogeneo

Le piattaforme di open data analizzate permettono agli utenti da ogni parte del mondo di caricare e salvare i propri dati rendendoli disponibili a chiunque avesse interesse a farne qualsiasi tipo di utilizzo, ad esempio visualizzarli online, salvarli ed utilizzarli per i propri scopi. I due diversi data-store presentano caratteristiche comuni e alcune differenze.

ThingSpeak

ThingSpeak [1] è una piattaforma open source per l'IoT che permette agli utenti di collezionare, salvare e analizzare dati. Fornisce agli utenti una personale piattaforma cloud su cui poter caricare e visualizzare i dati prodotti da sensori e dispositivi vari grazie alle API messe a disposizione. I dati inseriti da ciascun utente sono suddivisi in gruppi chiamati *channel* caratterizzati da libertà di espressione, ovvero è possibile inserire qualsiasi dato senza nessun vincolo sul nome del channel o dei campi stessi. I channel possono essere aggiornati con nuove misurazioni ogni 15 secondi, aggiungendo

[1] https://thingspeak.com/

record ai dati già presenti. Infine i dati relativi ai channel sono accessibili e scaricabili in diversi formati: JSON, XML o CSV [11]. Tuttavia ai fini della nostra applicazione i dati sono stati estratti da un file JSON contenente sia i dati che i metadati relativi al channel, accessibile dall'URL del channel del tipo:

`https://thingspeak.com/channels/id_channel/feed.json.`

Sparkfun

Sparkfun Electronics, Inc.[2] è un rivenditore di microcontrollori il quale fornisce anche altri servizi, tra cui diversi tipi di tutorial e funge da piattaforma open source di open data. Come nel caso di ThingSpeak, anche in Sparkfun l'utente può caricare e salvare i dati prodotti da dispositivi e sensori con una certa libertà di espressione. A differenza di ThingSpeak tuttavia, i gruppi in cui sono divisi i dati dei diversi utenti sono chiamati *stream* e la localizzazione della provenienza dei dati non è rappresentata da coordinate GPS ma dal nome della città e/o della nazione, se presenti. Tali dati sono pubblici e disponibili a chiunque volesse usufruirne, potendoli scaricare in diversi formati: JSON, CSV, MySQL, PostgreSQL e Atom [11]. Anche per Sparkfun ai fini della nostra applicazione i dati e i metadati sono stati estratti da file di tipo JSON, in questo caso tuttavia gli URL a cui accedere e da cui prendere i dati sono differenti: per i metadati l'URL è di tipo:

`https://data.sparkfun.com/streams/streamID.json.`

Mentre per i dati si fa una richiesta all'URL:

`https://data.sparkfun.com/output/streamID.json.`

Nel file contenente i metadati è inoltre presente la localizzazione dello stream in termini di coordinate GPS.

La figura 2.1 mostra l'architettura del progetto implementato per l'estrazione il clustering e la classificazione dei dati di ThingSpeak e Sparkfun.

[2]https://www.sparkfun.com/

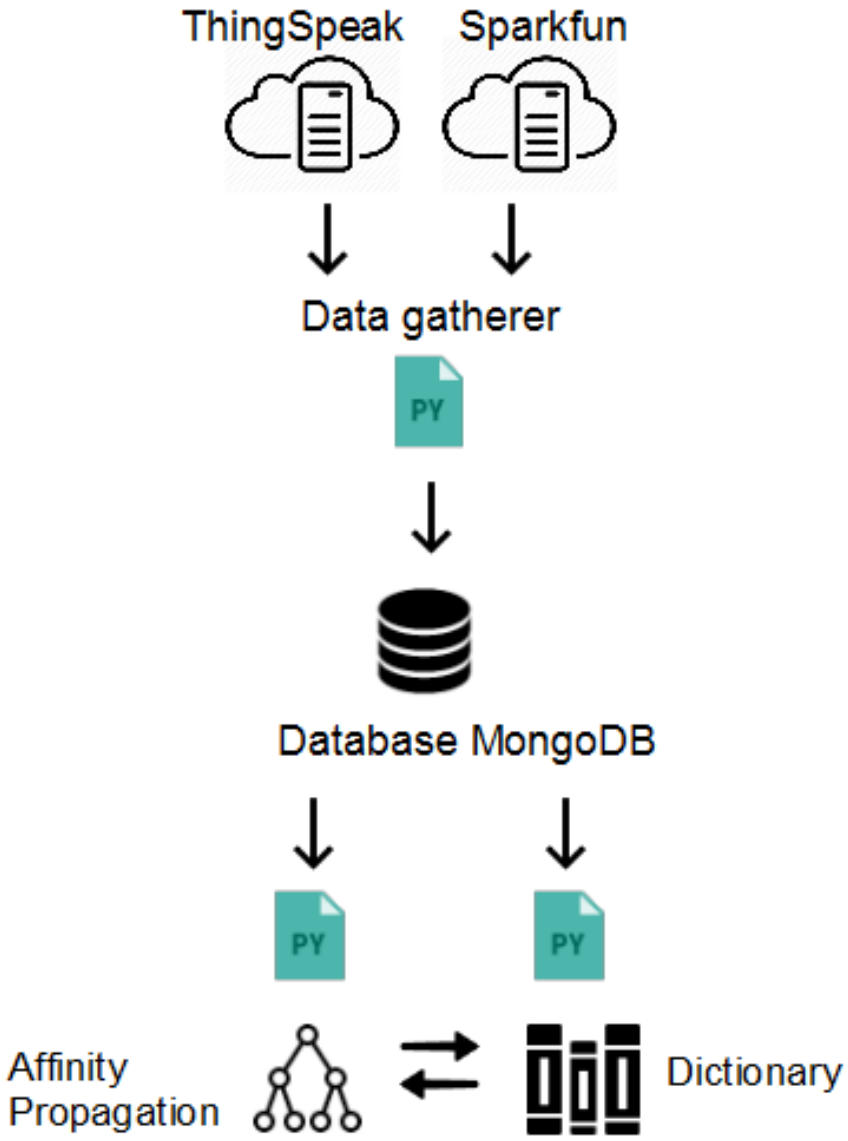


Figura 2.1: Architettura del progetto realizzato

2.2.1 Strutturazione dataset

Per la realizzazione dell'applicativo è stato utilizzato il linguaggio di programmazione dinamico Python[3], orientato agli oggetti e utilizzabile per molti tipi di sviluppo software. Offre un forte supporto all'integrazione con altri linguaggi e programmi ed è fornito di un'estesa libreria standard.

L'applicazione non dispone di un client o di un'interfaccia web o desktop poichè ha il compito di estrarre i dati degli *stream/channel* da ThingSpeak e Sparkfun e in seguito processare tali dati.

MongoDB

MongoDB[4] è un DBMS di tipo non relazionale orientato ai documenti, di tipo No-SQL. Si differenzia dai classici database relazionali poichè i dati non sono collezionati in strutture tabellari, bensì in documenti di tipo JSON con schema dinamico, tipicamente in formato BSON[5] (Binary JSON). Un database MongoDB è suddiviso in *collection*, corrispondenti alle tabelle nei database relazionali. All'interno di una singola collection possono essere presenti documenti strutturati in maniera differente, senza quindi avere vincoli di schema o sui campi presenti. Tipicamente tutti i documenti all'interno di una specifica collection sono caratterizzati da uno scopo comune. Oltre alle collection, un altro elemento che caratterizza i database MongoDB sono i *documenti*. Un documento rappresenta un singolo record all'interno di una collection, quindi l'unità di base dei dati in MongoDB. Tipicamente è rappresentato da un oggetto di tipo BSON. Infine ogni documento è composto da campi, ovvero *fields*. Ogni campo corrisponde ad una coppia nome-valore, analoghi alle colonne delle tabelle nei database relazionali.

[3] http://www.python.it/
[4] https://www.mongodb.com
[5] http://bsonspec.org

Dataset realizzato

I dati e i metadati estratti dalle due piattaforme sono stati salvati in un database locale MongoDB, con documenti strutturati in maniera simile sia per ThingSpeak che per Sparkfun. Il tool utilizzato per gestire i documenti è Robomongo[6].

Il primo step è quello di estrarre i metadati degli *stream/channel* e salvarli in una collection unica, con documenti strutturati nel modo seguente:

```
{"stream_id": Integer,
 "nome": String,
 "descrizione": String,
 "tags": List,
 "campi": List,
 "geolocalization": {"longitude": Double,
                     "latitude": Double},
 "last_update": Timestamp,
 "last_entry_id": Double
 }
```

Tutti i valori salvati nella collection dei metadati rappresentano un'informazione significativa per lo stream o il channel in questione. I metadati estratti dalle due piattaforme per gli scopi della nostra applicazione sono i seguenti [11]:

- **Stream ID**: è l'identificativo univoco che viene assegnato allo stream al momento della creazione. In ThingSpeak corrisponde ad un numero incrementale mentre in Sparkfun è una stringa di 20 caratteri combinati in maniera casuale. Al momento dell'estrazione sono stati rilevati 28806 stream attivi per ThingSpeak e 3575 per Sparkfun.

- **Nome stream**: è il nome che l'utente attribuisce al proprio stream senza alcuna restrizione sul tipo di informazioni da inserire; infatti sono stati rilevati casi in cui tale metadato si è rivelato utile per dedurre le categorie di informazioni che andava a misurare e casi in cui questo non avveniva.

[6] https://robomongo.org/

- **Descrizione**: è un campo non obbligatorio in cui l'utente può fornire una breve descrizione relativa al proprio stream e come per il nome non ci sono restrizioni sulla tipologia di informazioni da dover inserire.

- **Tag**: è un campo non obbligatorio composto da una o più parole chiave che l'utente assegna allo stream; ai fini dell'applicazione implementata sono stati utili per classificare i dati nei casi in cui i nomi dei valori misurati non erano ritenuti sufficienti.

- **Nome Campo**: è il nome attribuito a ciascuna misurazione fatta dall'utente. Anche in questo caso non ci sono restrizioni su ciò che l'utente può inserire per questo valore, ovvero se attribuire alla misurazione un'etichetta significativa o meno. All'interno del database realizzato questo valore è stato associato al nome della classe che lo rappresenta; in questo modo alla voce "campi" ogni stream avrà associata una lista, in cui ogni elemento è composto da una coppia **nome_campo - classe**.

- **Geolocalization**: in ThingSpeak vengono fornite le coordinate GPS *latitudine-longitudine*, mentre in Sparkfun sono presenti in aggiunta le informazioni relative alla città e/o allo stato; tuttavia tali informazioni non sono sempre presenti.

- **Last update**: è un timestamp corrispondente all'ultima modifica fatta sullo stream, ovvero l'ultima volta che l'utente ha caricato dei nuovi dati.

- **Last entry id**: è un'informazione presente solo in ThingSpeak e corrisponde all'ultimo update fatto dall'utente sullo stream, a cui viene attribuito un ID incrementale per associarlo ai dati relativi.

Tuttavia nei data store online sono presenti anche altre tipologie di metadati che non sono stati salvati nel database locale dell'applicazione, poichè non sono stati ritenuti fondamentali per gli scopi del progetto. Tali informazioni sono [11]:

- **Creation timestamp**: è presente nei metadati di entrambe le piattaforme. Tuttavia come si potrebbe pensare dal nome, non è detto che rappresenti il timestamp relativo al primo caricamento dei dati dello stream poichè sia in ThingSpeak che Sparkfun il numero di update per stream da poter memorizzare è limitato, infatti man mano che le registrazioni aumentano, quelle passate vengono eliminate.

- **Elevation**: è un metadato di ThingSpeak non sempre presente che rappresenta la distanza della locazione dello stream dal livello del mare.
- **Metadata**: è un metadato di ThingSpeak non sempre presente contenente informazioni addizionali in testo libero riguardo lo stream.
- **Url**: presente in ThingSpeak ma non obbligatorio, indica l'indirizzo in cui trovare la pagina web dello stream.

L'obiettivo principale in questo step è stato quello di costruire un dataset omogeneo, nel quale i dati delle due piattaforme siano uniformati e memorizzati allo stesso modo. Effettivamente le due strutture sono analoghe e si differenziano solo per due caratteristiche:

- **l'id dello stream/channel**, in Sparkfun è del tipo */streams/id_stream* mentre in ThingSpeak è rappresentato da un numero incrementale;
- **last_entry_id** campo presente sono nei metadati di ThingSpeak.

Come secondo step, è stato implementato uno script che permette di prendere dai due data store tutti i dati degli *stream/channel* salvati nella collection dei metadati e memorizzarli a loro volta in un'altra collection. In ogni documento vengono quindi salvate le seguenti informazioni sia per ThingSpeak che per Sparkfun:

```
{
"stream_id" : String,
"update" : List
}
```

dove nel campo *update* viene salvata la lista di misurazioni effettuate dal sensore o dal dispositivo in questione e i relativi valori. In particolare per ogni stream in ogni elemento della lista sono presenti il nome del campo misurato e la classe attribuita dall'analisi effettuata.

Gestione del database con Robomongo

Nella figura 2.2 viene mostrato uno screenshot del tool utilizzato per la gestione del database locale in questa applicazione, Robomongo.
Nel menù a sinistra viene mostrata la lista di collection creata per il database, in questo caso **metadati** e **dati**, mentre nella parte centrale sono elencati tutti i documenti della collection selezionata, nel caso della figura 2.2 si tratta appunto dei documenti JSON contenenti i metadati di tutti gli stream/channel estratti dai data store.

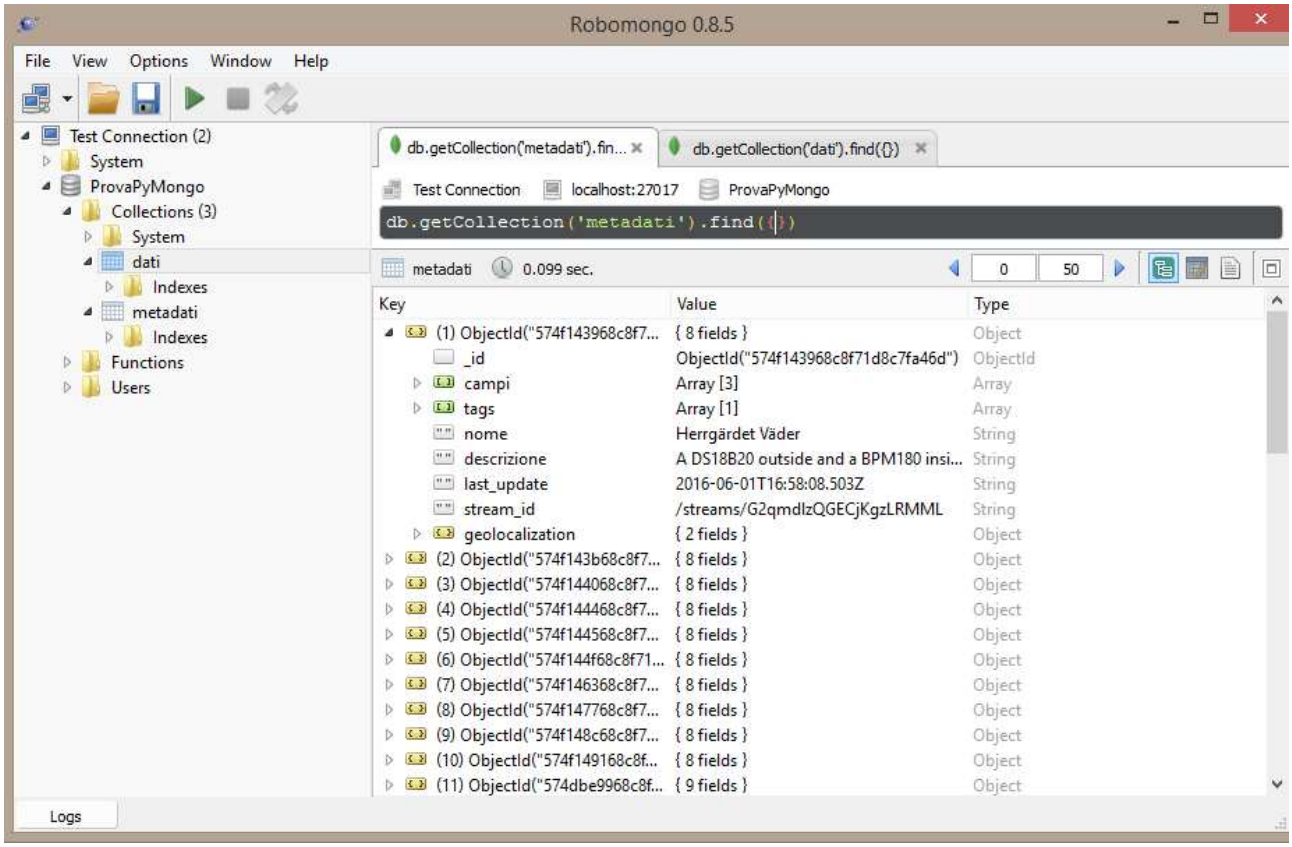


Figura 2.2: Screenshot del dataset realizzato con Robomongo

2.3 Estrazione classi significative

Un ulteriore step consiste nell'aver trovato un set di classi significative che potessero rappresentare le più comuni e frequenti rilevazioni fatte dagli utenti utilizzatori delle piattaforme. A questo scopo è stato analizzato un insieme di 300 *stream/channel* e sono state individuate 32 classi di valori misurati con più frequenza.

Nella tabella 2.1 sono elencate le 32 classi estratte dall'analisi di ThingSpeak e Sparkfun, le quali rispecchiano i tipi di campi più comuni misurati dagli utenti che usufruiscono dei due data store per salvare le proprie misurazioni.

Classe	Descrizione	Occorrenze
Dust Level	livello di polvere nei vari contesti	1 stream
CPU Usage	utilizzo della CPU	5 stream
Distance	distanza tra due punti/elementi	1 stream
Energy	capacita di un corpo di compiere un lavoro	1 stream
Capacity	misurazioni di una quantita di liquidi	2 stream
Gas Level	livello di gas nei diversi ambienti	5 stream
Price	prezzo nelle diverse monete	7 stream
Geolocalization	latitudine e longitudine	1 stream
Brightness	luminosità di un ambiente	50 stream
Memory	capacità di memoria nelle diverse unità	5 stream
Motion	quantità di moto di un sistema	3 stream
Time	espressione del tempo	36 stream
Power	potenza espressa in watt	6 stream
Pressure	rappresenta la pressione di un sistema	31 stream
PH	acidità o della basicità di una soluzione acquosa	9 stream
Rain Index	indice di piovosità	19 stream
Radiation	misurazione radiazioni	1 stream
Temperature	temperatura in gradi Celsius o Fahrenheit	191 stream
Voltage	misurazioni della differenza di potenziale elettrico	12 stream
Current	differenti valori per la corrente elettrica	8 stream
Humidity	grado di umidità in un ambiente	87 stream
UV	livello dei raggi uv	1 stream
Wind	direzione del vento	20 stream
Speed	velocità di un sistema, ad esempio del vento	22 stream
Height	altezza di un qualsiasi elemento	1 stream
Rate	indica un certo tipo di tasso o percentuale	3 stream
Battery Level	livello di un qualsiasi tipo di batteria	18 stream
Heat Index	combinazione di umidità e temperatura	6 stream
Count	numero di volte in cui si è verificato un evento	4 stream
RSSI	usato per misurare la potenza di un segnale	2 stream
LQI	misura la qualità di un link di comunicazione	1 stream
Colour	rappresenta i colori	3 stream

Tabella 2.1: Occorrenze classi

Capitolo 3

Analisi dei dati

L'obiettivo principale del sistema implementato è quello di ricavare delle informazioni significative dai dati estratti da ThingSpeak e Sparkfun.
In particolare si intende dedurre quali sono le principali tipologie di misurazioni fatte dagli utenti che usufruiscono delle due piattaforme per la pubblicazione e la condivisione dei dati da loro prodotti.

Per estrarre tali informazioni significative sono stati utilizzati due tra gli algoritmi di Data Mining presentati successivamente, ovvero l'Affinity Propagation e il Dictionary Learning, opportunamente adattati alle esigenze dei dati in questione.
Tali metodologie di Data Mining rientrano rispettivamente nell'ambito degli algoritmi di **clustering** e di **classificazione**.

3.1 Data Mining

Il Data mining è un'area dell'informatica con un'ampia prospettiva di sviluppo e presenta numerose potenzialità. Lo scopo di tale disciplina è quello di processare dati per scoprire ed estrarre informazioni utili da database di dimensioni significative.
Esistono diverse aree sotto il data mining, tra cui la classificazione, che spiegheremo meglio nel dettaglio poichè si tratta di un punto fondamentale di questa tesi.

La significativa quantità di dati generati dall'IoT presenta un'elevata utilità e di conseguenza rappresentano informazioni di ampio valore. Indubbiamente il Data Mining gioca un ruolo primario nel rendere questo sistema (le applicazioni dell'IoT) abbastanza intelligente da fornire servizi ed ambienti sempre più utili [19].

In generale le tecniche di data mining prescindono dalla natura dei dati stessi che nel caso dell'IoT, come già visto, è eterogenea. Tali tecniche in particolare presentano vantaggi significativi per l'elaborazione di dati:

- non è necessario essere a conoscenza in partenza di ipotesi sui dati;
- non sono richiste in partenza ipotesi sulla forma distributiva delle variabili;
- possibilità di elaborare una grande quantità osservazioni;

Di conseguenza date le caratteristiche delle suddette tecniche, esse possono essere ritenute valide per elaborare la vasta mole di informazioni prodotte dall'IoT.

Tuttavia è importante scegliere quale tra gli algoritmi di data mining sottoporre al proprio sistema [19]. Tra queste si trovano:

- Clustering, se l'obiettivo è quello di estrarre dei pattern non etichettati;
- Classificazione, nel caso in cui si volesse estrarre dei pattern parzialmente etichettati;
- Association Rules, se si volessero cercare eventi dai pattern in input che non si presentano in un ordine particolare;
- Sequential Patterns, come per l'Association Rules ma nel caso in cui gli eventi si presentano in un determinato ordine;

La figura 3.1 mostra le quattro tipologie di algoritmi di data mining precedentemente descritti, tra i quali sono stati selezionati il Clustering e la Classificazione nello sviluppo di questa tesi.

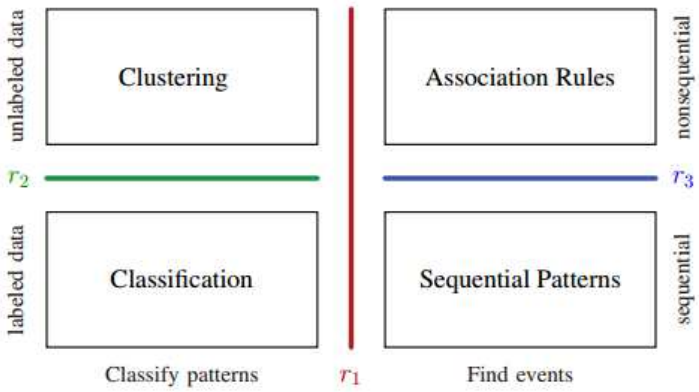


Figura 3.1: Algoritmi di Data Mining [19]

3.2 Clustering

Lo scopo degli algoritmi di clustering è quello di estrarre da un set di dati dei gruppi concettualmente significativi che siano relazionati tra loro, appunto i cluster. I gruppi di dati vengono estratti sulla base delle informazioni presenti nei dati stessi. L'obiettivo è quello di avere gruppi in cui gli oggetti siano simili tra loro ovvero **related**, e diversi da quelli di altri gruppi cioè **unrelated**.

La definizione dei cluster non è precisa e dipende dalla natura e dalle caratteristiche dei dati presi in esame; tali gruppi significativi condividono caratteristiche comuni tra loro. In generale in un algoritmo di clustering non si hanno a priori informazioni riguardo ai cluster da estrarre, quindi non si conoscono le caratteristiche dei dati che vengono estratti. Tra i vari algoritmi tuttavia ce ne sono alcuni in cui è noto il numero di cluster da estrarre, come ad esempio il K-means.
La **Cluster Analysis** è lo studio delle tecniche per la ricerca automatizzata di cluster. I campi di applicazione dell'analisi dei cluster sono vari, tra cui pattern recognition, information retrieval, machine learning e data mining [17].

3.3 Classificazione

A differenza delle tecniche di clustering le metodologie di classificazione intendono ricercare automaticamente da un set di dati la classe di appartenenza, partendo da informazioni relative al numero e/o alla tipologia delle classi risultanti. Vengono applicate delle tecniche statistiche ai dati in questione con lo scopo di assegnare un'*etichetta* alle informazioni estratte.
In questo ambito rientrano gli **alberi di decisione**, una struttura dati composta da *nodi decisionali* e *nodi foglia* in cui i nodi foglia rappresentano le classificazioni e i nodi decisionali l'insieme delle proprietà che portano a quelle classificazioni. Gli ambiti applicativi degli algoritmi di classificazione variano dal data mining al machine learning [16]. Gli algoritmi di classificazione si fondano su alcuni concetti di base, come ad esempio la distanza tra parole che verrà analizzata in seguito.

Idealmente, si può pensare che gli algoritmi di clustering e di classificazione debbano svolgere funzioni tipiche degli esseri umani:

- Clustering: l'abilità degli esseri umani di suddividere gli oggetti in gruppi.
- Classificazione: la capacità di assegnare agli oggetti un'etichetta ovvero una classe di appartenenza.

Supervised e Unsupervised Classification

In particolare è possibile fare un'ulteriore distinzione tra le due tecniche usate nella nostra applicazione.
L'Affinity Propagation infatti rientra nella categoria delle **Unsupervised Classification** mentre il Dictionary fa parte delle **Supervised Classification**.

- **Supervised Classification**: si conoscono in partenza il numero e/o il tipo di classi da dedurre.
- **Unsupervised Classification**: non è noto nè il numero nè il tipo di informazioni da ricavare.

Di seguito vengono spiegate nel dettaglio le implementazioni dei due algoritmi nell'ambito della nostra applicazione per l'estrazione di gruppi di informazioni significative.

3.4 Supervised Classification

Tra gli algoritmi "supervisionati" rientra il *Dictionary Learning*. Nei metodi di dictionary learning un dizionario è costituito con elementi di un *training set* ed è utilizzato per classificare elementi di un *test set* [6].

In questa applicazione è stata adattata l'implementazione dell'algoritmo per rispondere alle caratteristiche dei dati e del sistema in questione.
Innanzitutto sono stati selezionati due insiemi di dati dal database, i quali costituiscono rispettivamente il *training set* e il *test set*.

- **Training set**: composto da 601 record, è stato utilizzato per popolare il dizionario sul quale fondare la classificazione, formato dalle 32 classi trovate e per ogni classe da un insieme di termini rappresentanti il nome dei campi che rientrano nell'area di tale classe.

- **Test set**: composto da 240 record, è stato utilizzato per testare l'algoritmo del *Dizionario* e verificarne la bontà e la precisione.

Popolazione dizionario

La struttura dati su cui è stata basata la classificazione è di tipo dict[1], composta da 32 chiavi corripondenti alle classi estratte manualmente dai data store, e a ciascuna classe è associata una lista di parole. Le liste sono popolate a partire dagli elementi del trainig set, in cui ogni record è composto dal nome del campo misurato associato alla classe che lo caratterizza. La porzione di script Python seguente mostra il modo in cui le suddette stringhe sono estratte per popolare la struttura del dizionario:

```
with open('training_set.csv', 'rb') as csvfile:
 spamreader = csv.reader(csvfile, delimiter=',', quotechar='|')
  for row in spamreader:
   campo = row[0].strip().lower().replace(' ', '_')
   classe = row[1].strip()
   dictionary[classe].append(campo)
```

3.4.1 Implementazione

Dopo aver riempito la struttura del dizionario con i dati del training set e aver selezionato il test set, viene classificata ogni parola presente nel test set confrontandola con la lista di termini presenti nel dizionario per ogni classe. Tale confronto avviene calcolando la *distanza tra parole*, in particolare la **distanza di Damerau-Levenshtein** [7] tra due termini. La distanza tra parole rappresenta la similarità tra due stringhe ed è pari a 0 se i due termini sono uguali. In particolare quella chiamata *Edit distance* corrisponde al numero minimo di inserzioni, rimozioni e sostituzioni richieste per trasformare una stringa in un'altra [15]. La distanza di Damerau-Levenshtein inoltre, tiene conto anche dei passaggi richiesti per la trasposizione di due caratteri adiacenti all'interno della stringa [9].

[1] https://docs.python.org/2/tutorial/datastructures.html#dictionaries

Preprocessing

Prima di procedere con la classificazione viene effettuato un passaggio intermedio in cui i termini da classificare vengono ridotti in lettere minuscole e vengono sostituiti gli spazi con un altro carattere, poichè la stessa cosa viene fatta con gli esemplari che popolano il dizionario affinchè la distanza di Damerau-Levenshtein sia il più precisa possibile.

Di seguito viene mostrata una porzione di codice che dimostra come la stringa da classificare viene confrontata con tutti gli elementi presenti nel dizionario calcolandone la distanza relativa; tra tutte le distanze ottenute per ogni classe viene scelta quella che risulta inferiore a tutte.

```
res = {}
# Per ogni classe
# calcolo la distanza tra le parole di cui è composta
# e la parola da testare

stringa_da_classificare = 'airpressure'

for classe in dictionary.keys():
    dam = [(damerau_levenshtein_distance(stringa_da_classificare, campo))
              for campo in dictionary[classe]]

    res[classe] = min(dam)
```

Con l'esecuzione di questo passaggio si ottiene una struttura di tipo dict[2] le cui chiavi sono le 32 classi significative e per ciascuna classe è indicata la distanza minima trovata tra la stringa da classificare e l'insieme di esemplari della classe in questione.

[2]https://docs.python.org/2/tutorial/datastructures.html#dictionaries

Ad esempio:

```
# Stringa da classificare: airpressure.
# Distanze minime trovate per classe calcolando la distanza di
# Damerau-Levenshtein:

{"Battery_Level": 9, "Capacity": 12, "Temperature": 6,
"Time": 8, "Radiation": 10,  "Rain_Index": 9,  "Current": 8,
"LQI": 11, "Memory": 8, "Power": 10, "Price": 8, "Pressure": 0,
"RSSI": 8, "PH": 10, "Heat_Index": 10, "Count": 11, "Distance": 11,
"CPU_Usage": 10, "Gas_Level": 10, "Brightness": 8, "Colour": 9,
"Humidity": 7, "Motion": 11, "Geolocalization": 9, "Wind": 9,
"Energy": 11, "UV": 10, "Height": 10, "Rate": 9, "Voltage": 10,
"Dust_Level": 10, "Speed": 9}
```

In questo caso la distanza minima trovata è pari a 0 e corrisponde alla classe "Pressure", permettendo immediatamente l'attribuzione della classe corrispondente alla stringa.

A questo punto è possiblie stabilire la classe adatta a rappresentare la stringa classificata se si presentano le seguenti condizioni:

- se la distanza minima trovata è pari a 0;
- se la distanza minima trovata è inferiore ad un valore calcolato come percentuale δ sulla lunghezza della stringa da classificare;
- se il valore calcolato come percentuale ϵ rappresentante la differenza tra la distanza minima e tutte le altre distanze trovate, è maggiore del valore calcolato con δ.

Se si verifica una tra le prime due condizioni, si attribuisce immediatamente la classe corrispondente; altrimenti si procede con un'ulteriore classificazione, introducendo oltre alla stringa da classificare anche i **tag** presenti nei metadati per tale campo. In questo caso viene preso in considerazione un ulteriore valore calcolato con ϵ che misura la differenza in percentuale tra la distanza minima trovata per la stringa e tutte le altre distanze: tutte le classi per cui il valore calcolato con ϵ è minore di quello calcolato con δ vengono riclassificate con il supporto dei tag.

In seguito a questo step, si decide quale classe attribuire alla stringa scegliendo sempre la distanza minima trovata.

3.4.2 Librerie utilizzate

Per l'implementazione di tale algoritmo sono state utilizzate diverse librerie Python, tra cui: pyxdameraulevenshtein[3] e PyMongo[4].

pyxDamerauLevenshtein

Tale libreria implementa l'algoritmo per trovare la distanza di Damerau-Levenshtein in Python.
Come importare la libreria:

```
from pyxdameraulevenshtein import damerau_levenshtein_distance
```

Come ottenere la distanza di Damerau-Levenshtein tra due stringhe:

```
distanza = damerau_levenshtein_distance(stringa_1, stringa_2)
```

PyMongo

PyMongo è una distribuzione Python contenente tool per lavorare con MongoDB[5]. Per utilizzare la libreria è necessario eseguire i seguenti passaggi:

Importare la libreria:

```
from pymongo import MongoClient
```

Creare un'istanza MongoClient specificando host e porta

```
client = MongoClient(host, port)
```

Creare un'istanza del database utilizzato

```
db = client.nome_database
```

[3]https://pypi.python.org/pypi/pyxDamerauLevenshtein
[4]https://api.mongodb.com/python/current/
[5]https://api.mongodb.com/python/current/

Accedere alle collection esistenti

```
collection = db.nome_collection
```

3.5 Unsupervised Classification

Come accennato in precedenza le tecniche di clustering hanno lo scopo di estrarre dei gruppi significativi, appunto i *cluster*, da un insieme di dati.
Sono *unsupervised* quelle che tecniche che per essere eseguite non richiedono in partenza informazioni sul tipo di cluster da determinare o stimare. Nella nostra applicazione è stato sfruttato il metodo dell'Affinity Propagation.
L'Affinity Propagation è un algoritmo di tipo message-passing, in cui ogni elemento da clusterizzare comunica con gli altri tramite l'invio di messaggi. Si immagini che ogni elemento informi i restanti della propria attrattività ad associarsi con gli altri, e che ciascuno risponda con la disponibilità ad essere associato a tale elemento, dato il messaggio di attrattività ricevuto. Il passaggio di messaggi procede finchè non è raggiunto un consenso sulle associazioni migliori, considerando l'attrattività relativa e la disponibilità. Per ogni elemento, quello associato meglio ne rappresenta l'esemplare e tutti quelli che condividono l'esemplare sono riuniti nello stesso cluster [18]. In altre parole il messaggio inviato tra le coppie rappresenta l'idoneità di un elemento ad essere l'esemplare per gli altri, valore aggiornato dopo aver ricevuto le risposte. Tale aggiornamento avviene iterativamente finchè non si ottiene convergenza e si stabiliscono gli esemplari finali, quindi i cluster risultanti[6].

3.5.1 Implementazione

Il gruppo di dati di cui effettuare il clustering con l'Affinity Propagation coincide con il *test set* utilizzato nell'algoritmo del Dizionario, quindi si tratta di stringhe corrispondenti ai nomi delle varie misurazioni effettuate dagli utenti utilizzatori dei data store ThingSpeak e Sparkfun.
Per prima cosa viene costruita una lista contenente tutte queste stringhe. Tale lista verrà poi iterata per poter calcolare la distanza di Damerau-Levenshtein sulle coppie di parole al proprio interno.

```
# Costruzione della lista usufruendo della libreria NumPy
words = np.asarray(parole_da_clusterizzare)
```

[6] http://scikit-learn.org/stable/modules/clustering.html#affinity-propagation

```
# Calcolo della distanza tra tutte le parole presenti nella lista
dam = np.array([[(damerau_levenshtein_distance(w1, w2)) for w1 in words]
                for w2 in words])
```

Avendo ottenuto tutte le distanze tra le parole, è possibile procedere con l'Affinity Propagation e invocare un metodo della libreria utilizzata per questo scopo, trasformando la lista di distanze in forma matriciale:

```
# Applica l'affinity propagation sulla matrice di affinità
affprop.fit(matrice_distanze)
```

Il metodo fit(X[, y]) crea una matrice di affinità dalle distanze e successivamente applica l'affinity propagation clustering[7].

Come risultato si ottengono dei gruppi di parole cioè i cluster, composti da stringhe simili e rappresentanti ciascuno un gruppo omogeneo.

3.5.2 Librerie utilizzate

Sono state utilizzate alcune librerie Python per l'implementazione di tale algoritmo: NumPy[8] e Sklearn[9].

NumPy

NumPy è un pacchetto che fornisce funzionalità per elaborare dati scientifici in Python. Inoltre NumPy può essere usato come un efficiente container multi-dimensionale di dati generici.

[7] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html
[8] http://www.numpy.org
[9] https://pypi.python.org/pypi/scikit-learn/0.15.2

Sklearn

Sklearn consiste in un insieme di moduli Python per il machine learning e il data mining. Ai fini dell'applicazione è stato utilizzato il meotodo di tale libreria per realizzare l'Affinity Propagation:

```
ap = sklearn.cluster.AffinityPropagation(affinity="precomputed", damping=0.5)
```

Capitolo 4

Risultati

In questo capitolo vengono illustrati i risultati ottenuti dall'applicazione dei due differenti algoritmi di Data Mining, il Dictionary e l'Affinity Propagation.
In particolare di seguito vengono illustrate le differenti metodologie di misurazione delle performance, provenienti dall'ambito della statistica.
Vengono presi in considerazione i risultati numerici calcolati nei due approcci e infine i dati risultanti vengono confrontati tra loro.

Per quanto riguarda l'approccio del Dictionary, avendo a disposizione a priori informazioni riguardo ai dati elaborati (numero e tipologia delle classi da dedurre) è stato possibile calcolare l'occorrenza e la precisione di tali informazioni nei risultati ottenuti; per quanto riguarda l'Affinity Propagation invece, non si conoscono in partenza il numero e la tipologia dei cluster da ottenere. Tuttavia nel progetto sono stati combinati i due approcci in modo da poter avere anche per l'Affinity Propagation delle metriche analoghe a quelle del Dictionary. In particolare, una volta estratti i cluster questi sono stati classificati tramite l'algoritmo del Dictionary implementato per l'altro approccio, apportando alcune modifiche: gli elementi presenti all'interno di ciascun cluster risultante sono stati confrontati con il nome delle classi stabilite per il Dictionary, in modo da poter verificare l'occorrenza di tali classi all'interno di ogni cluster. Quindi sono state calcolare le relative metriche di riuscita per questi risultati.

4.1 Metriche di riuscita

Per valutare la bontà e l'efficienza dei diversi algoritmi sono state calcolate tre diverse metriche correlate tra loro: **Precision**, **Recall** e **F-measure**.
Nel contesto della nostra applicazione, le variabili utilizzate per il calcolo delle metriche sono le seguenti: numero di termini classificati in maniera corretta, numero di termini classificati in una certa classe, numero di termini realmente appartenenti ad una certa classe. Questi tre valori cambiano in base all'algoritmo di riferimento e verrano spiegati nel dettaglio nelle sezioni 4.2.1 e 4.2.2.

4.1.1 Precision

La Precisione o come viene chiamata in Data Mining, *Confidence*, denota la proporzione di casi previsti positivi che sono realmente positivi [13]. In altre parole rappresenta la porzione di istanze considerate che sono effettivamente rilevanti.
Per ciascuna classe trovata nella fase di Data Retrieval, è calcolata nel seguente modo:

$$\frac{\text{n° parole classificate in maniera corretta}}{\text{n° parole classificate con una certa classe}}$$

4.1.2 Recall

Il Recall è la proporzione dei casi veri positivi che sono correttamente predetti positivamente [13]. In altre parole denota la porzione di istanze rilevanti tra quelle considerate.
Per ciascuna classe trovata nella fase di Data Retrieval, è calcolata nel seguente modo:

$$\frac{\text{n° parole classificate in maniera corretta}}{\text{n° parole appartenenti realmente ad una certa classe}}$$

4.1.3 F-measure

Rappresenta una misura che combina Precision e Recall ovvero la loro media armonica, anche detta F-score [13].
Per ciascuna classe trovata nella fase di Data Retrieval, è calcolata nel seguente modo:

$$2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Tali metriche sono prima calcolate per ogni classe stabilita e successivamente i risultati di ogni classe vengono sommati tra loro con l'obiettivo di ottenere un risultato unico per l'algoritmo corrispondente.

4.2 Metodologie

Le suddette metriche sono state calcolate per valutare la riuscita di entrambi gli algoritmi implementati. In relazione ai risultati ottenuti per ciascuna metrica si possono trarre delle conclusioni riguardo allo specifico algoritmo e alla sua riuscita.

4.2.1 Risultati Dictionary

Riguardo all'implementazione del Dictionary Learning, sono stati confrontati i risultati ottenuti combinando due parametri per l'esecuzione di tale algoritmo. I due valori rappresentano due indici:

- il primo rappresenta una percentuale da calcolare sulla lunghezza della stringa da classificare, il δ mostrato nella sezione 3.4.1;
- il secondo invece rappresenta una percentuale da calcolare sulla differenza tra due distanze computate, l' ϵ mostrato nella sezione 3.4.1;

L'invocazione di tale algoritmo quindi si presenta nel modo seguente:

```
dictionary_learning(delta, epsilon)
```

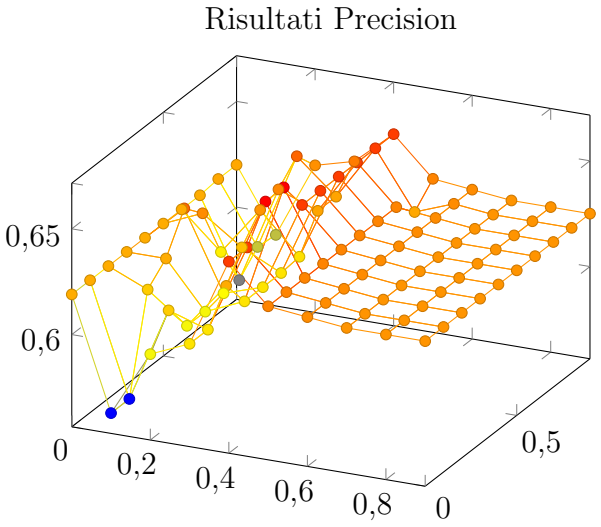
dove δ è l'indice da calcolare in percentuale sulla lunghezza della stringa da classificare e ϵ è l'indice da applicare sulla differenza tra due distanze.

Come accennato nelle sezioni riguardanti Precision Recall e F-Measure, per ottenere tali metriche di riuscita sono stati calcolati per ciascuna classe tre valori numerici, ovvero:

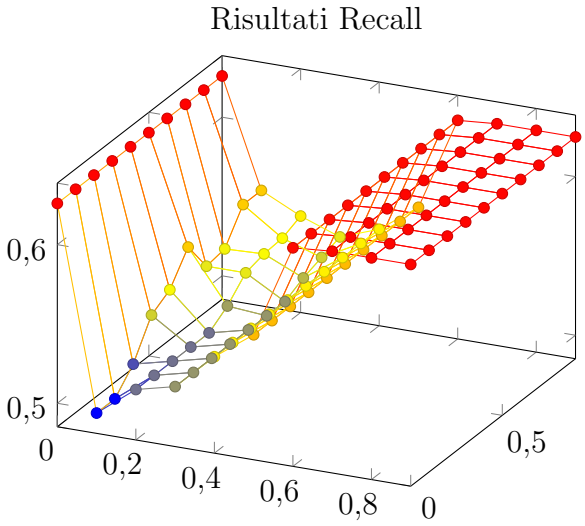
- **n° parole classificate per tale classe**, corrispondente al numero di parole attribuite a tale classe tra tutte quelle del test set, che siano classificate in maniera esatta o meno;
- **n° parole classificate esatte per tale classe**, ovvero il numero di parole del test set attribuite in maniera esatta per la classe in questione, cioè tutte le parole per cui la classificazione è risultata esatta;

- **n° parole effettive corrispondenti a tale classe**, che rappresenta il numero di parole realmente attribuibili a tale classe, senza tener conto dei risultati della classificazione.

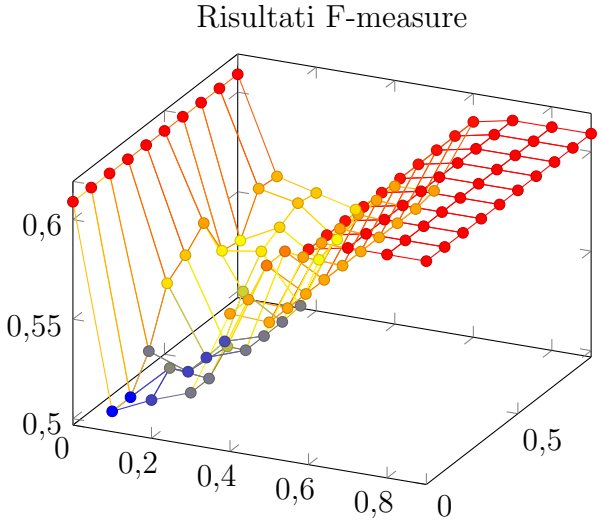
Sono stati realizzati dei grafici in tre dimensioni per mostrare i risultati delle tre metriche per l'approccio del Dictionary. Sugli assi x e y sono posizionate le percentuali combinate per l'esecuzione dell'algoritmo, tali δ e ϵ di cui si è discusso in precedenza. Sull'asse z invece sono posizionati i valori di Precision per le relative combinazioni.



Come si può notare dal grafico la maggior parte delle combinazioni portano a un risultato di precisione che si aggira attorno al valore 0.6; considerando che una precisione perfetta corrisponde al valore 1, si può affermare che tali risultati possono essere considerati come una buona base di partenza per lo sviluppo di tali metodologie. Inoltre precisioni migliori si ottengono nei punti in cui δ è fissato al 40%; ciò vuol dire che per tutte le parole considerate è opportuno confrontare la distanza di Damerau-Levenshtein con un valore sulla lunghezza della parola che si aggira attorno al 40% affinchè la classificazione risulti maggiormente precisa.



Per quanto riguarda il Recall invece, dal grafico emerge quanto segue. Innanzitutto come per la precisione i valori non superano lo 0.6 e si stabiliscono al di sotto di tale valore, inoltre risultano inferiori per δ compresi tra il 10% e il 50%. Ciò sta a indicare che in questo caso che la classificazione risulta migliore se il δ è superiore al 50%, mentre ϵ non influisce in maniera significativa sui risultati.



I risultati dell'F-measure invece rappresentano la media armonica tra precision e recall. Volendo riferirsi ad un unico valore per valutare la riuscita dell'algoritmo, si può osservare appunto l'F-measure. Dal grafico si deduce che i risultati migliori si aggirano attorno allo 0.6 visto che entrambe le metriche precedenti avevano valori simili. Anche

qui si può dunque sostenere che tali risultati sono una buona base di partenza per gli sviluppi futuri; dal momento che non ci sono lavori simili per gli stessi dati l'obiettivo di tale metodologia è stato appunto quello di verificare l'esistenza di risultati di base su cui fondare la prima versione dell'algoritmo. Di conseguenza si intende affinare la tecnica del Dictionary per ottenere risultati migliori, indicativamente con lo scopo di ottenere risultati per l'f-measure superiori allo 0.8.

Nella tabella 4.1 vengono elencati i risultati numerici relativi all'F-measure: rappresentano i valori ottenuti tramite le combinazioni delle due percentuali, δ sull'asse verticale e ϵ su quello orizzontale.

%	0	10	20	30	40	50	60	70	80	90
0	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609
10	0.507	0.507	0.523	0.550	0.557	0.566	0.545	0.543	0.562	0.561
20	0.516	0.525	0.516	0.516	0.517	0.535	0.548	0.553	0.558	0.556
30	0.523	0.523	0.532	0.523	0.523	0.523	0.524	0.540	0.543	0.551
40	0.566	0.566	0.576	0.576	0.566	0.566	0.566	0.566	0.566	0.566
50	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.567
60	0.605	0.605	0.605	0.605	0.605	0.605	0.605	0.605	0.605	0.605
70	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609
80	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609
90	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.609

Tabella 4.1: Valori di F-measure per il Dictionary

4.2.2 Risultati Affinity Propagation

Per verificare la riuscita dell'Affinity Propagation, i cluster ottenuti sono stati classificati con l'algoritmo del Dizionario esaminato in precedenza a cui sono state apportate delle modifiche per adattarlo a questa situazione.
Ogni termine presente all'interno di ciascun cluster viene confrontato con il nome delle classi presenti nel dizionario e tra queste due stringhe viene calcolata la distanza di Damerau-Levenshtein. Come risultato viene fornito il nome della classe con più occorrenze all'interno del cluster.
Infine per ottenere delle metriche simili a quelle del Dictionary Learning viene fatta una valutazione analoga. Anche in questo caso dunque ogni classe presente nel dizionario avrà tre valori numerici:

1. **n° parole classificate per tale classe**, sono le parole presenti nei cluster che vengono attribuite alla classe in questione;

2. **n° parole classificate esatte per tale classe**, è il numero di parole nei cluster che vengono classificate in maniera esatta;

3. **n° parole effettive corrispondenti a tale classe**, è il numero di parole realmente attribuibili a tale classe, senza tener conto dei risultati della classificazione.

Per quanto riguarda l'Affinity Propagation, il parametro dato in input all'algoritmo che influenza la riuscita dello stesso è il *Damping Factor*, compreso tra 0.5 e 1.

```
ap = sklearn.cluster.AffinityPropagation(affinity="precomputed", damping=0.5)
```

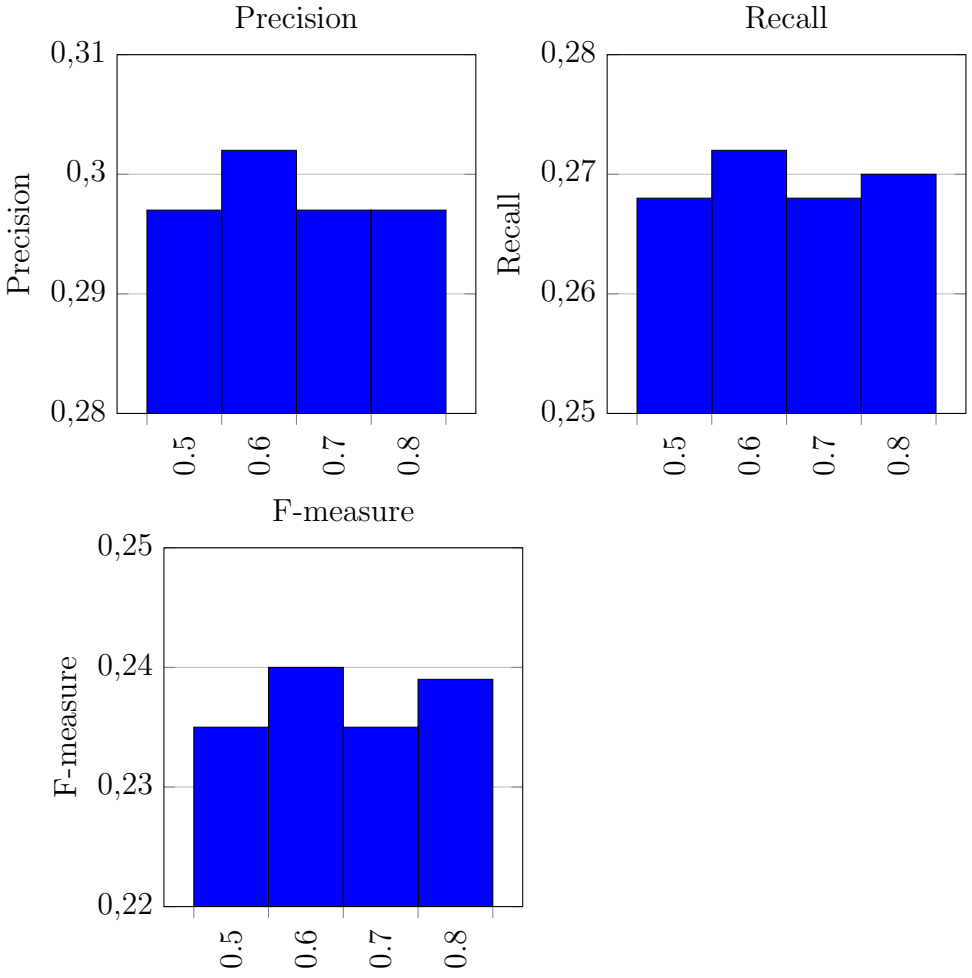
Comunemente il damping è necessario nei metodi di sovrarilassamento (over-relaxation methods) e in questo caso previene la presenza di fallimenti e di oscillazioni nella soluzione. Affinchè ci sia convergenza nell'Affinity Propagation, il damping adeguato non deve avere effetti significativi sul risultato, quindi damping elevati portano ad una lenta convergenza [12].

Dall'applicazione dell'Affinity Propagation con diversi valori per il damping risultano le metriche mostrate in tabella 4.2.

Damping	*Precision*	*Recall*	$F - measure$
0.5	0.297	0.268	0.235
0.6	0.302	0.272	0.240
0.7	0.297	0.268	0.235
0.8	0.297	0.270	0.239
0.9	0.297	0.268	0.235

Tabella 4.2: Risultati Affinity Propagation

Questi risultati sono stati rappresentati tramite tre istogrammi, uno per ogni metrica.



Per quanto riguarda gli istogrammi sull'asse delle x sono rappresentati i valori di damping e sull'asse delle y i relativi valori di precision, recall e f-measure.
Come si evince dai dati nella tabella e graficamente dagli istogrammi, il calcolo delle metriche per l'Affinity Propagation non ha ottenuto risultati da poter ritenere soddisfacenti. Comunque, sia la precisione che il recall presentano i valori maggiori con damping pari a 0.6. L'analisi di tali risultati va fatta tenendo conto che tali metriche si riferiscono alla classificazione fatta successivamente alla fase di clustering; per quanto riguarda tale fase infatti, non è possibile calcolare le suddette metriche di performance non essendo appunto note informazioni riguardo al test set. Tuttavia nel nostro caso il numero di classi da estrarre è noto e dunque si può confrontare con il numero di cluster risultanti dall'Affinity Propagation, come prima valutazione.

Conclusioni

In questo lavoro è stato presentato il concetto di Internet of Things e i diversi ambiti applicativi della disciplina, concentrandosi sull'importanza della disponibilità degli open data per la realizzazione di applicazioni separate ma interoperabili tra loro.
Sono state analizzate nello specifico due piattaforme di pubblicazione e condivisione di dati sensoristici, quali ThingSpeak e Sparkfun che sono state il punto di partenza per la realizzazione del progetto di tesi.
Vengono illustrate nel dettaglio le modalità di estrazione di dati dalle due piattaforme, i quali sono stati successivamente unificati e raggruppati in un database comune.
Tali dati sono stati processati tramite due tecniche di Data Mining: il Dictionary per effettuare la classificazione e l'Affinity Propagation per il clustering; lo scopo di tali implementazioni è stato quello di dedurre informazioni significative dai dati estratti. In particolare nell'algoritmo del Dictionary si era a conoscenza in partenza del numero e del tipo di informazioni da dedurre poichè è stata effettuata un'analisi manuale sui 300 stream selezionati; tutti i dati estratti dalle piattaforme vengono dunque classificati a partire da queste informazioni note a priori, con la possibilità di poter confrontare le informazioni di partenza con i risultati.
Per l'Affinity Propagation invece, nessuna informazione era nota in partenza; i cluster ottenuti dunque sono stati accomunati soltanto grazie alle informazioni degli elementi che li compongono. Tuttavia nel nostro approccio è stato introdotto uno step di classificazione all'Affinity Propagation: dopo che sono stati estratti i cluster in modo *unsupervised*, ciascun gruppo omogeneo di elementi è stato classificato tramite l'algoritmo del Dizionario. In questo modo è stato possibile attribuire dei risultati numerici analoghi ai due approcci.

Alla luce dei risultati ottenuti si può sostenere che le due implementazioni generano dei risultati che possono essere considerati una buona base partendo dal presupposto che non sono state effettuate altrove analisi simili per gli stessi dati; risultano quindi una base di partenza per sviluppi futuri: in particolare l'approccio Dictionary presenta metriche di riuscita migliori rispetto all'Affinity Propagation, grazie al fatto di avere a priori una conoscenza delle caratteristiche dei dati processati.

Sviluppi futuri

Considerando quindi i risultati ottenuti si può sostenere che entrambe le metodologie necessitano di percentuali di riuscita migliori per potersi considerare affidabili. Una possibile soluzione sarebbe quella di ampliare il training set, di modo che la struttura del dizionario sia più precisa e consenta una classificazione migliore. Un altro modo sarebbe quello di estendere la classificazione anche ad altre informazioni presenti negli stream, oltre che unicamente al nome del campo misurato e dei tag; tuttavia tale approccio è stato sperimentato e si è notato che in alcuni contesti non sempre le suddette informazioni sono servite a migliorare i risultati: infatti se tali dati (nome dello stream, descrizione dello stream) non sono significativi, essi possono addirittura peggiorare la riuscita dell'algoritmo.
Infine un ulteriore approccio da poter implementare è la tecnica del K-means, che rientra tra gli algoritmi unsupervised benchè sia noto a priori il numero di cluster da estrarre. Tale tecnica di clustering permette di prendere in considerazione altre informazioni come ad esempio la media e la varianza delle misurazioni presenti tra i dati estratti. L'obiettivo è quello di estrarre dei gruppi di dati con la possibilità di discriminare ulteriormente i dati raccolti; ad esempio nel caso dei valori di temperatura, le implementazioni realizzate permettono di distiguere i valori solo in base al nome del campo, mentre utilizzando i dati numerici è possibile fare un'ulteriore distinzione per esempio tra gradi Celsius e Fahrenheit.

Bibliografia

[1] Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015. `http://www.gartner.com/newsroom/id/3165317`.

[2] Kevin Ashton. That "internet of things" thing. *RFiD Journal*, 22(7):97–114, 2009.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[4] Payam Barnaghi, Wei Wang, Cory Henson, and Kerry Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[5] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pages 205–227, 2009.

[6] Teresa Bufford, Yuxin Chen, Mitchell Horning, and Liberty Shee. When dictionary learning meets classification. 2013.

[7] Fred J Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[8] Dave Evans. The internet of things, how the next evolution of the internet is changing everything. *Cisco Internet Business Solutions Group (IBSG)*, 2011.

[9] Wael H Gomaa and Aly A Fahmy. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13), 2013.

[10] Berg Insight. Smart homes and home automation. 2015.

[11] Federico Montori, Luca Bedogni, and Luciano Bononi. On the integration of heterogeneous data sources for the collaborative internet of things. Accepted for publication at RTSI Bologna 2016.

[12] University of Toronto Probabilistic and Statistical Inference Group. Affinity propagation faq. `http://www.psi.toronto.edu/affinitypropagation/faq.html`.

[13] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[14] Pasquale Puzio. Internet of things and its applications. `http://www.slideshare.net/PasqualePuzio/internet-of-things-and-its-applications`, 2011.

[15] Eric Sven Ristad and Peter N Yianilos. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532, 1998.

[16] S. Ruggieri. Efficient c4.5 [classification algorithm]. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):438–444, Mar 2002.

[17] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education India, 2006.

[18] Precha Thavikulwat. Affinity propagation: A clustering algorithm for computer-assisted business simulations and experiential exercises. *Developments in Business Simulation and Experiential Learning*, 35, 2014.

[19] Chun-Wei Tsai, Chin-Feng Lai, Ming-Chao Chiang, and Laurence T Yang. Data mining for internet of things: a survey. *IEEE Communications Surveys & Tutorials*, 16(1):77–97, 2014.

[20] Ovidiu Vermesan, Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Alessandro Bassi, Ignacio Soler Jubert, Margaretha Mazura, Mark Harrison, M Eisenhauer, et al. Internet of things strategic research roadmap. *Internet of Things: Global Technological and Societal Trends*, 1:9–52, 2011.

[21] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32, 2014.

Ringraziamenti

Ringrazio il professor Di Felice per avermi dato l'opportunità di svolgere questo progetto che mi ha permesso di scoprire aspetti della disciplina interessanti che non mi era capitato di intraprendere nel percorso di studi. Ringrazio inoltre il Dottor Montori per l'aiuto prezioso che si è rivelato in questo lavoro.

Un ringraziamento speciale va alla mia famiglia, senza la quale non avrei raggiunto questo traguardo. Spero di riuscire a ripagare tutti i sacrifici e il supporto che mi hanno sempre dimostrato.

Ringrazio di cuore le mie coinquiline, compagne di mille emozioni in questi anni, con le quali sarò sicura di poter condividere molti altri successi in futuro nonostante la lontananza. So che la distanza sarà solo una questione di kilometri.
Un grazie immenso va alle mie compagne dh'avventura, Silvia e Alice. Grazie per essere sempre le migliori. Di voi continuerò ad ammirare la caparbietà e la sensibilità che vi caratterizza.

Voglio ringraziare anche Andrea, che mi ha sopportato e supportato in questi ultimi mesi, nonostante molte volte non sia stato facile. Spero di riuscire a ricambiare almeno in parte tutto l'affetto che mi hai dimostrato.

Infine vorrei ringraziare gli amici di una vita, che nonostante la lontananza mi riempiono sempre di gioia e di serenità. È un piacere condividere con voi anche quest'altro traguardo.