Capitolo 1

Progettazione

Nella prima fase di questo progetto, abbiamo eseguito un'analisi approfondita dei requisiti specifici. Abbiamo identificato le seguenti esigenze per l'app mobile, il server e il frontend:

- **Applicazione mobile:** L'applicazione mobile deve essere in grado di riconoscere l'attività dell'utente, la variazione dell'attività di un utente (WALKING → DRIVING e viceversa), visualizzare una mappa con le varie aree poligonali di Bologna, chiedere al backend la disponibilità di parcheggi all'interno di una certa area poligonale, riconoscere la posizione dell'utente e inviare al server gli eventi di entrata e uscita di un parcheggio.

- **Server:** Il server deve essere in grado di ricevere e rispondere alle richieste di disponibilità di parcheggi da parte dell'app, ricevere le richieste di entrata e uscita di parcheggi da parte dell'app, inviare le aree poligonali di Bologna all'app e, in caso di zone prive di dati (ad esempio per mancanza di utilizzo dell'app da parte di utenti in quella zona) deve prevedere un meccanismo di interpolazione dei dati, per stimare il numero di parcheggi disponibili.

- **Frontend:** Il frontend deve essere in grado di visualizzare, per ogni zona della città, il numero totale di richieste di parcheggio provenienti da quella zona, utilizzare il clustering K-Means per visualizzare gli eventi di parcheggio e rappresentare gli eventi di parcheggio mediante una heatmap.

Per quanto riguarda la seconda fase, è stato richiesto di valutare l'accuratezza di tre classificatori di attività binari (solo WALKING e DRIVING). In particolare si è lavorato su un dataset fornito per allenare il modello di Human Activity Recognition (HAR). Sono stati testati tre algoritmi di classificazione: Random Forest, KNN, Guassian Bayes. Inoltre, è stata valutata l'accuratezza del sistema di HAR della libreria nell'app mobile mediante un piccolo testbed (esperimenti con pochi dati reali) e confrontata con l'accuratezza degli algoritmi testati, tenendo presente che il confronto ha una valenza scientifica

limitata in quanto si basa su dataset differenti. Infine, è stato integrato il Random Forest direttamente nell'app mobile prendendo i dati dai sensori e sono state sviluppate due feature aggiuntive a discrezione del gruppo.

1.1 Feature Aggiuntive

In aggiunta alle feature precedenti, la consegna prevedeva l'inclusione di due feature aggiuntive a discrezione del gruppo. Data la natura del progetto, abbiamo pensato a delle feature che potessero rendere più completo l'utilizzo del sistema da parte dell'utente e che rispecchiassero i contenuti del corso di studi. Abbiamo quindi deciso di implementare come prima feature aggiuntiva un secondo algoritmo di clustering già visto a lezione: l'algoritmo **DBSCAN**. L'algoritmo DBSCAN è un algoritmo di clustering che decide da solo il numero di cluster a partire da due parametri in input: l'*epsilon* cioè un treshold di distanza tra i punti ed il *minPoints*, cioè un treshold del numero di vicini per ogni punto. Come seconda feature aggiuntiva abbiamo trovato un dataset offerto dagli Open-Data del comune di Bologna che presenta la posizione di tutte le colonnine per la ricarica elettrica della città di Bologna. Abbiamo quindi integrato all'interno dell'app un sistema di crowdsensing che permette agli utenti di verificare lo stato di disponibilità di una cerca colonnina. Quando un utente parcheggerà il proprio veicolo nei pressi di una colonnina per la ricarica, apparirà un pop-up nell'applicazione che chiederà all'utente se sta utilizzando la colonnina. In caso di risposta affermativa il backend aggiornerà lo stato di disponibilità della colonnina che sarà quindi visibile agli altri utenti. La colonnina verrà liberata quando l'utente che l'ha occupata avrà lasciato il parcheggio. L'implementazione di queste feature aggiuntive ci ha permesso di estendere le funzionalità del nostro progetto con funzioni utili agli utenti pur rimanendo all'interno di implementazioni legate allo scopo del corso.