Human Activity Recognition sul Dataset HARTH: EDA, Preprocessing e Classificazione con Gradient Boosted Decision Tree

Giulia Tartaglia - 2113526 Romeo D'Angelo - 2157226

02/02/2026

Abstract

In questo progetto analizziamo il dataset HARTH (Human Activity Recognition Trondheim), un dataset di accelerometria triassiale in un contesto free-living. Dopo un'analisi esplorativa (EDA) delle registrazioni multi-soggetto e della distribuzione delle attività, proponiamo una pipeline di preprocessing basata su segmentazione in finestre temporali e feature engineering statistico. Addestriamo e valutiamo un modello di classificazione multiclasse basato su Gradient Boosted Decision Trees (LightGBM), usando uno split per soggetto per stimare la capacità di generalizzazione a nuovi individui. La valutazione include report di classificazione (precision, recall, F1-score), confusion matrix e analisi dell'importanza delle feature.

Contents

1 Introduzione

La *Human Activity Recognition* (HAR) mira a riconoscere automaticamente le attività di un individuo a partire da segnali provenienti da sensori. Applicazioni tipiche includono monitoraggio della salute, sport tracking, riabilitazione, assistenza agli anziani e rilevamento di comportamenti sedentari.

L'accelerometria triassiale è una delle sorgenti informative più diffuse: segnali in 3 assi che descrivono l'accelerazione lungo direzioni ortogonali. La complessità del problema HAR aumenta quando i dati sono raccolti in scenari *free-living* (vita quotidiana), perché:

- esiste elevata variabilità inter-soggetto (stile di camminata, altezza, postura, ecc.);
- le classi sono spesso sbilanciate (alcune attività sono più frequenti di altre);
- i confini tra attività possono essere ambigui (transizioni, movimenti simili).

1.1 Obiettivo del progetto

La consegna richiede di selezionare un dataset e sviluppare un modello di **classificazione o regressione**. In questo lavoro affrontiamo un problema di **classificazione multiclasse**:

> *dato un segmento temporale di accelerazioni misurate su schiena e coscia, predire quale attività stava svolgendo il soggetto.*

1.2 Approccio proposto

Il notebook implementa una pipeline completa:

1. download ed estrazione del dataset;
2. EDA (dimensioni, durata, qualità dati, distribuzione classi, ispezione segnali);
3. preprocessing e feature engineering:
   - segmentazione in finestre temporali con overlap;
   - filtraggio finestre "pure" (una sola label nella finestra);
   - estrazione di feature statistiche e di energia;
4. split training/test **per soggetto** (valutazione realistica);
5. training con LightGBM (GBDT);
6. valutazione con metriche per classe, confusion matrix e feature importance.

2 Descrizione del Dataset

2.1 Panoramica

Il dataset **HARTH** (Human Activity Recognition Trondheim) contiene registrazioni di **22 soggetti** in contesto free-living. Ogni soggetto indossa due accelerometri triassiali posizionati su:

- **lower back** (schiena bassa),
- **right front thigh** (coscia destra anteriore).

La frequenza di campionamento è **50 Hz**, cioè 50 misure al secondo per asse.

2.2 Formato dei file e variabili

Ogni soggetto è rappresentato da un file CSV separato contenente tipicamente:

- `timestamp` (data e ora del campione),
- accelerazioni schiena: `back_x`, `back_y`, `back_z`,
- accelerazioni coscia: `thigh_x`, `thigh_y`, `thigh_z`,
- `label`: codice numerico dell'attività.

2.3 Classi di attività

Le attività annotate includono (codice → nome):

- 1: walking, 2: running, 3: shuffling,
- 4: stairs (ascending), 5: stairs (descending),
- 6: standing, 7: sitting, 8: lying,
- 13: cycling (sit), 14: cycling (stand),
- 130: cycling (sit, inactive), 140: cycling (stand, inactive).

Nel notebook è prevista (opzionale) la rimozione delle classi "inactive" (130 e 140) se si vuole focalizzarsi sulle attività principali.

3 Analisi Esplorativa dei Dati (EDA)

L'EDA serve a comprendere struttura, qualità e criticità del dataset prima della modellazione.

3.1 Dimensione del dataset e durata

Il codice:

- conta il numero di file CSV (soggetti),
- somma le righe totali,
- ricava la durata totale (campioni / 50 Hz).

Questa analisi fornisce una stima della quantità di dati disponibili e della copertura temporale.

3.2 Qualità del dato: valori mancanti

Viene verificata l'assenza di *missing values* su un sottoinsieme di file (o su tutti i file, se abilitato). L'assenza di missing riduce la necessità di imputazione e semplifica il preprocessing.

3.3 Distribuzione delle classi

Il notebook calcola la frequenza delle label su tutto il dataset e visualizza un istogramma. In un contesto free-living è normale osservare:

- classi dominanti (es. standing/sitting),
- classi rare (es. running o stairs),
- conseguente **class imbalance**.

Questo è importante perché l'accuracy da sola può essere fuorviante: il modello potrebbe favorire le classi frequenti.

3.4 Durata per soggetto

Viene stimata la durata di ciascun file/soggetto. Differenze tra soggetti sono normali e possono influenzare:

- distribuzione delle attività,
- varietà di pattern disponibili in training,
- robustezza della generalizzazione.

3.5 Statistiche descrittive dei segnali

Il notebook calcola `describe()` sui canali accelerometrici per un soggetto, ottenendo media, deviazione standard, min/max e quantili. Questo aiuta a:

- identificare eventuali outlier,
- intuire differenze tra sensori e assi.

3.6 Visualizzazione del segnale grezzo

Viene tracciato un segmento breve (es. 5 secondi) del segnale per i tre assi della schiena. La visualizzazione qualitativa permette di vedere:

- cambiamenti rapidi in transizioni.

3.7 Feature EDA: energia per classe

Una feature semplice (energia su `back_x`) viene calcolata su finestre non sovrapposte e mostrata con un boxplot per classe. Scopo:

- verificare se la feature discrimina tra attività,
- identificare sovrapposizioni tra classi,
- motivare l'uso di feature engineering e modelli non lineari.

4 Preprocessing e Feature Engineering

4.1 Motivazione

I dati originali sono **time-series** ad alta frequenza; per usare modelli tabellari come GBDT è necessario trasformare il segnale in un insieme di feature per segmento.

4.2 Pulizia minima

Nel notebook:

- si rimuove `timestamp` (non necessario per classificazione in questa baseline),
- si converte `label` in intero,
- (opzionale) si filtrano le label "inactive" {130, 140}.

4.3 Segmentazione in finestre (windowing)

Ogni file viene segmentato in finestre temporali di:

- **WINDOW = 100 campioni** ($\approx$ 2 secondi a 50 Hz),
- **STEP = 50 campioni** ($\approx$ 1 secondo, overlap 50%).

Questo compromesso permette di catturare dinamiche motorie (cicli di passo, ecc.) mantenendo un numero di campioni sufficiente per feature stabili.

4.4 Finestre "pure"

Per ogni finestra si mantiene solo se `label` è costante all'interno della finestra (`nunique()==1`). Motivazione:

- riduce rumore da transizioni;
- garantisce che l'etichetta sia coerente con le feature estratte;
- rende la supervisione più affidabile.

4.5 Feature estratte

Per ciascun asse e sensore, il codice calcola un set di feature statistiche:

- media (μ),
- deviazione standard (σ),
- minimo, massimo,
- energia: $\frac{1}{N}\sum_{i=1}^{N} x_i^2$.

Inoltre aggiunge due feature globali:

- **SMA back**: media della somma dei valori assoluti sui tre assi della schiena,
- **SMA thigh**: analogo per la coscia.

Con 6 canali $\times$ 5 feature = 30, più 2 SMA = **32 feature** totali.

4.6 Costruzione del dataset finale

Per ogni finestra valida vengono salvati:

- X: vettore di feature (dimensione 32),
- y: label della finestra,
- `group`: identificativo del soggetto (nome file), usato per split.

5 Scelta del Modello

5.1 Perché un modello GBDT (LightGBM)

Il modello scelto è un **Gradient Boosted Decision Trees** (GBDT) implementato tramite LightGBM. Le motivazioni principali sono:

- **Non linearità**: le relazioni tra feature e attività sono spesso non lineari (es. combinazioni di energia e variabilità).
- **Robustezza**: buona resistenza al rumore tipico del free-living.
- **Prestazioni su tabellare**: GBDT è uno standard de-facto su feature engineered tabellari.
- **Efficienza**: training rapido anche su decine di migliaia di finestre.
- **Interpretabilità**: disponibilità di feature importance.

5.2 Perché non Logistic Regression o kNN come modello principale

- **Logistic Regression**: è un classificatore lineare; spesso insufficiente per separare classi molto simili (es. sitting vs standing) quando la separazione richiede frontiere non lineari.
- **kNN**: tende a essere più sensibile al rumore, richiede molta memoria e può risultare lento in predizione su dataset grandi; inoltre soffre in spazi a dimensionalità medio-alta.

Questi modelli restano utili come *baseline*, ma GBDT è più adatto come modello finale in questa pipeline.

5.3 Iperparametri usati

Nel notebook, un set ragionevole di iperparametri:

- `n_estimators=500`, `learning_rate=0.05`
- `num_leaves=63`
- `subsample=0.9`, `colsample_bytree=0.9`

Questi controllano capacità del modello e regolarizzazione (riducendo overfitting).

6 Strategia di Split e Valutazione (Setup sperimentale)

6.1 Split per soggetto (GroupShuffleSplit)

Per stimare la generalizzazione a nuovi individui, il dataset viene diviso in training e test con **GroupShuffleSplit** usando come gruppo il soggetto (file). Motivazione:

- evita che finestre dello stesso soggetto finiscano sia in train che test;
- riduce il rischio di *data leakage* (il modello imparerebbe caratteristiche del soggetto);
- è una valutazione più realistica per un sistema HAR utilizzabile su persone non viste.

6.2 Training

Il modello LightGBM viene addestrato su X_{train} e valutato su X_{test}.

7 Metriche di Valutazione e Motivazioni

In un problema di classificazione multiclasse con class imbalance è importante usare metriche per classe.

7.1 Precision, Recall, F1-score

Per ogni classe c:

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c}, \qquad \text{Recall}_c = \frac{TP_c}{TP_c + FN_c}$$

$$\text{F1}_c = 2 \cdot \frac{\text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}$$

Motivazione:

- **Precision**: quanto sono affidabili le predizioni della classe.
- **Recall**: quanto la classe viene effettivamente riconosciuta.
- **F1**: bilancia i due aspetti, utile quando le classi sono sbilanciate.

7.2 Confusion Matrix

La confusion matrix mostra, per ogni classe reale (righe), quante volte viene predetta ogni classe (colonne). È fondamentale per capire **quali attività si confondono** (es. sitting vs standing; stairs up vs down; cycling sit vs stand).

7.3 F1-score per classe (grafico a barre)

Il grafico del F1 per classe rende immediato individuare:

- classi più difficili,
- eventuale bias verso classi frequenti,
- necessità di bilanciamento o nuove feature.

7.4 Feature Importance

LightGBM fornisce un punteggio di importanza per feature. Questo permette:

- interpretabilità (quali statistiche contano di più),
- diagnosi (feature inutili o ridondanti),
- possibili miglioramenti futuri (aggiunta di feature nel dominio della frequenza).

8 Risultati e Discussione

8.1 Output prodotto dal notebook

Il notebook produce:

- grafici EDA: distribuzione label, durata per soggetto, segnale grezzo, boxplot energia;
- dimensione finale del dataset finestrato (X e y);

- classificazione: `classification_report` (precision/recall/F1);
- confusion matrix;
- F1 per classe;
- feature importance.

8.2 Interpretazione tipica dei risultati

Sebbene i valori numerici dipendano dallo split casuale e dalla rimozione delle classi inattive, in HAR è comune osservare:

- buone performance sulle attività dinamiche con pattern netti (walking vs running);
- maggiore confusione tra posture statiche (standing vs sitting) perché l'accelerazione può essere simile;
- confusione tra attività correlate (stairs up vs stairs down) se le feature sono solo statistiche e non frequenziali.

8.3 Nota sulle label "mancanti" (9–12)

È normale che le label saltino alcuni numeri (es. non esistono 9,10,11,12): la numerazione è una scelta degli autori del dataset. Il modello usa solo le classi effettivamente presenti in y.

8.4 Nota sui log di LightGBM

Messaggi come `Auto-choosing row-wise` o `No further splits with positive gain` indicano che, in alcuni alberi, non è utile aggiungere ulteriori ramificazioni con guadagno positivo.

9 Conclusioni

Il progetto implementa una pipeline completa per HAR su dataset free-living:

- EDA per comprendere qualità e distribuzioni del dataset;
- preprocessing con segmentazione e feature engineering;
- training di un classificatore GBDT (LightGBM) adatto a dati tabellari;
- valutazione robusta con split per soggetto e metriche per classe.

L'approccio è una baseline solida e replicabile, adatta a confronti futuri e a estensioni.