Activity Recognition from Wrist IMU (PAMAP2)

Chouaib (github.com/ChouaibB)

January 21, 2026

Contents

1. Data source: PAMAP2 (UCI)

Dataset: *PAMAP2 Physical Activity Monitoring* (UCI Machine Learning Repository)

- **UCI dataset page (download + description):**
  - https://archive.ics.uci.edu/dataset/231/pamap2+physical+activity+monitoring

- **Local path (expected):** download and unzip the dataset into `./data/pamap2/` (the `data/` directory is not committed to git)

- **Dataset readme (format / notes):**
  - https://archive.ics.uci.edu/ml/machine-learning-databases/00231/readme.pdf

- **License:** Creative Commons Attribution 4.0 International (**CC BY 4.0**)

PAMAP2 is a public wearable-sensing dataset for **human activity recognition (HAR)**. It contains recordings from **9 subjects** performing **18 labeled activities** (plus label `0` for transient/other) while wearing three IMU sensors (**hand/wrist**, **chest**, **ankle**) and a heart-rate monitor.

File format (from the dataset readme): each row has **54 columns**: - `1` timestamp (s)
- `2` activityID
- `3` heart rate (bpm)
- `4-20` IMU hand (wrist)
- `21-37` IMU chest
- `38-54` IMU ankle

Each IMU block contains: - temperature (°C)
- 3D accelerometer (±16g) and 3D accelerometer (±6g)
- 3D gyroscope (rad/s)
- 3D magnetometer (T)
- orientation (not valid in this collection)

Activity IDs (18 activities + transient): 1 lying; 2 sitting; 3 standing; 4 walking; 5 running; 6 cycling; 7 Nordic walking;
9 watching TV; 10 computer work; 11 car driving; 12 ascending stairs; 13 descending stairs;
16 vacuum cleaning; 17 ironing; 18 folding laundry; 19 house cleaning; 20 playing soccer; 24 rope jumping;
0 other (transient activities)

What we use in this notebook: Protocol recordings only, with a **watch-like setup**: wrist IMU signals (ACC/GYRO/MAG) as inputs and the **activityID** as the multi-class label. We **exclude label `0`** ("transient/other") to keep the benchmark focused on the defined activities.

2. Load data & quick sanity checks

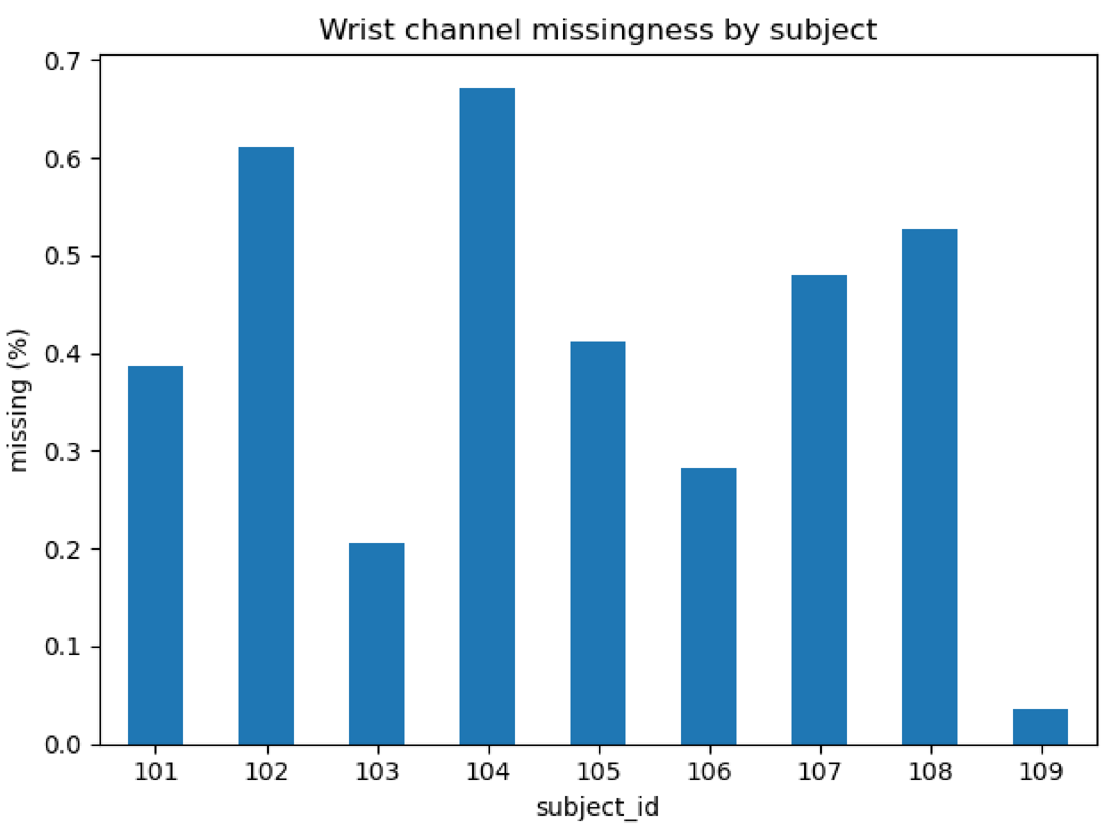
We load the **Protocol** recordings (one file per subject) from `./data/pamap2/`. The raw `.dat` files do not include column headers, so we explicitly assign the **54 columns** defined in the dataset readme. A `subject_id` is added based on the filename.

As a sanity check, we verify that the raw files match the expected **54-column** structure. After this check passes, we immediately **subset to the wrist/hand signals** plus the minimal metadata needed for HAR (`timestamp_s`, `activity_id`, `subject_id`). We also **drop the orientation channels** because they are stated to be **invalid** in this data collection.

Finally, we summarize **activity coverage per subject** and visualize **subject-wise missingness** for the wrist channels.

```
Raw files OK: 54 raw columns (expected 54)

            n_samples  n_activities (of 19 incl. 0: transient)
subject_id
101            376417                                       13
102            447000                                       13
103            252833                                        9
104            329576                                       12
105            374783                                       13
106            361817                                       13
107            313599                                       12
108            408031                                       13
109              8477                                        2
```



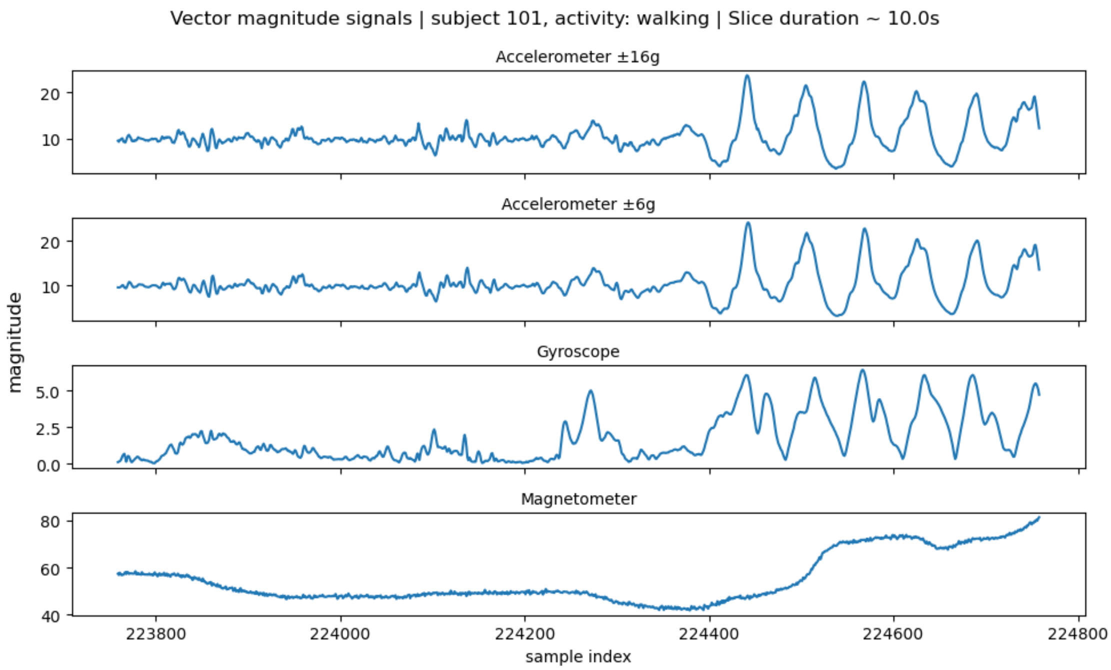
3. Preprocessing

In this section, we prepare the wrist IMU signals for windowing and feature extraction:

- We **remove transient activity samples** (`activity_id == 0`) to focus on well-defined activity classes.
- We handle **missing values** by interpolating short gaps **within each subject** and dropping any remaining missing samples.
- For each tri-axial wrist sensor, we compute the **vector magnitude** signal to provide an orientation-robust representation.

We visualize as an example, the **vector magnitude signals computed** for a short segment of one subject and one activity.

```
Preprocessed data shape: (1942872, 20)
```



4. Windowing

We segment the continuous wrist IMU signals into **fixed-length sliding windows**, which are the basic samples used for feature extraction and modeling in wearable activity recognition.

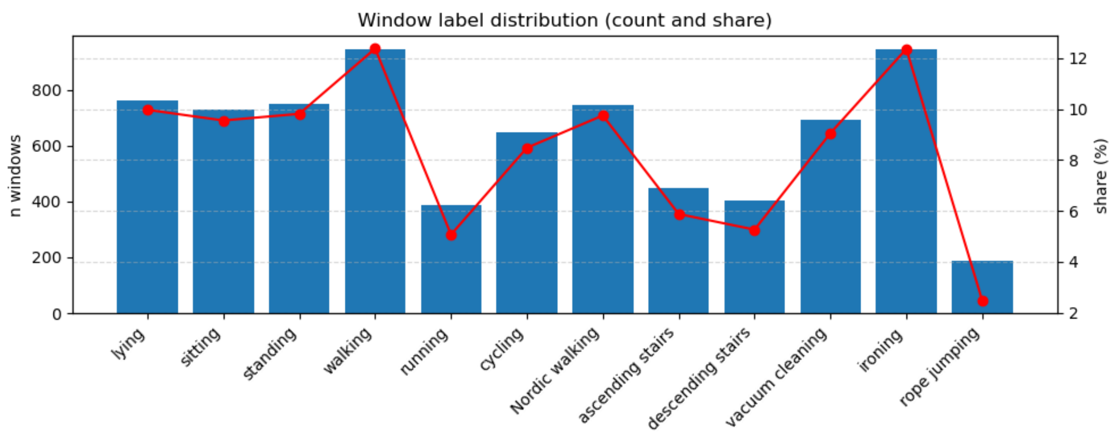
We use a **5-second window** with **50% overlap**, a common and widely accepted choice in HAR that balances temporal context with responsiveness. Windowing is performed **within each subject** to preserve temporal structure and avoid leakage. The effective **sampling rate is estimated from timestamps** to translate time-based window definitions (seconds) into sample indices in a data-driven way.

Each window is assigned a single activity label using a **majority vote** over the samples it contains. To reduce label noise near activity transitions, we apply a **label purity threshold of 0.8** (a common choice in HAR) and **discard windows below this threshold**, focusing the model on learning from **steady-state activities** rather than ambiguous transition regions.

```
Estimated fs: 100 Hz | window: 500 samples (5.0s) | step: 250 samples (50%
overlap)

Windows: (7645, 500, 17) | Labels: (7645,) | Groups: (7645,)

            n_windows  kept_fraction
subject_id
101               983       0.984970
102              1037       0.985741
103               685       0.984195
104               912       0.987013
105              1075       0.988051
106               984       0.984985
107               913       0.981720
108              1032       0.985673
109                24       1.000000
```



5. Feature extraction

In this section, we convert each windowed wrist IMU signal into a **fixed-length feature vector** suitable for classical machine learning models.

For each window and each signal, we compute the following **simple time-domain features**:

- mean
- standard deviation
- median
- minimum
- maximum

- interquartile range (IQR)

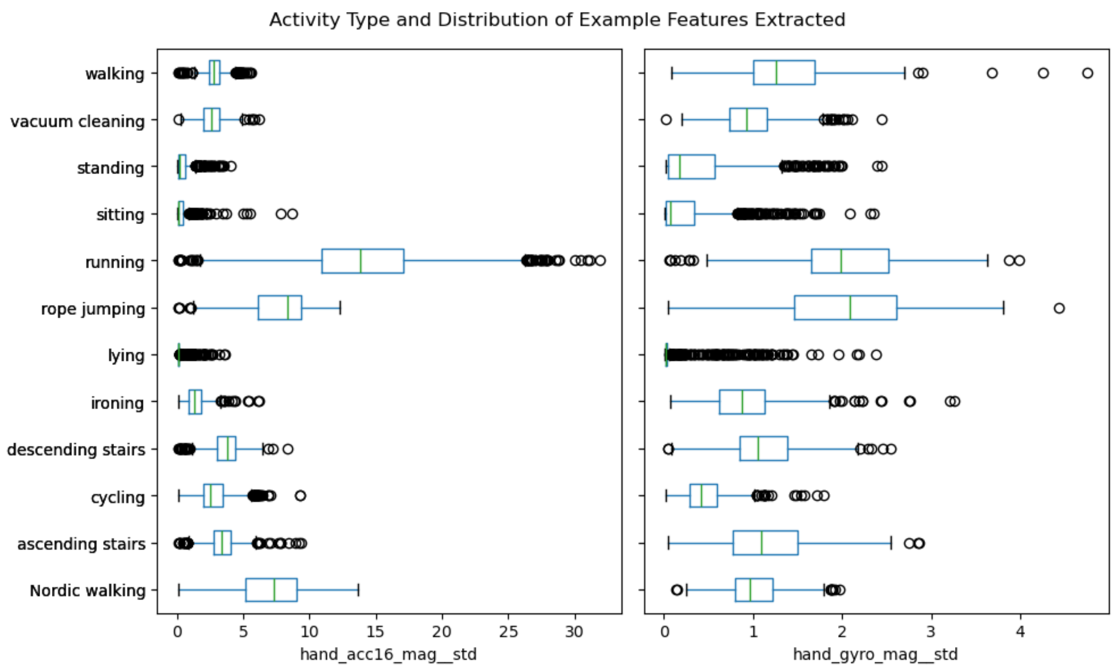
- root mean square (RMS)

These statistics capture signal level, variability, range, and overall energy within the window.

This feature-based representation is a **standard approach in wearable activity recognition** for classical models such as logistic regression and gradient boosting. More complex energy-style or frequency-domain features are intentionally omitted here to keep the notebook compact and focused.

```
Feature matrix: (7645, 119)
n_features: 119
```

As a lightweight sanity check, we inspect the **univariate distributions** of a small number of representative features. This helps verify that the extracted features have sensible ranges and capture differences across activities, without performing extensive exploratory analysis.



6. Modeling & subject-wise evaluation

In this section, we train and evaluate classical activity recognition models using the extracted window-level features, with a validation protocol that reflects **generalization to unseen users**.

We use **GroupKFold cross-validation**, grouping by `subject_id`, to ensure that windows from the same subject never appear in both training and test folds. This avoids subject-level leakage and mirrors real wearable deployment scenarios.

We evaluate two model families:
- **Logistic Regression (LR)** as a simple, interpretable baseline (with feature standardization).

- **Histogram-based Gradient Boosting (HGB)** as a stronger classical model for tabular data.

For each model, we compare performance using:
- the **original feature space**, and
- a **PCA-compressed feature space**, where PCA retains **90% of the variance** and is fit **within each training fold** via a pipeline to avoid leakage.

The PCA-based evaluation is included **to relate model performance to the upcoming decision boundary visualizations in PCA space**, rather than to mimic a real deployment setup.

We report **macro-F1** (primary metric) and **balanced accuracy** (secondary metric), summarized as mean ± standard deviation across folds.

The results of this section provide a quantitative basis for comparing models and feature representations, and set the stage for the interpretability visualizations in the final section.

```
CV: LR: 100%|
                                        | 5/5
[00:00<00:00,  5.92it/s]
CV: LR+PCA(90%): 100%|
                                        | 5/5
[00:00<00:00,  6.82it/s]
CV: HGB: 100%|
                                        | 5/5
[00:39<00:00,  7.83s/it]
CV: HGB+PCA(90%): 100%|
                                        | 5/5
[00:42<00:00,  8.60s/it]
```

```
Model performance (GroupKFold, mean ± std)
```

	model	macro_f1_mean	macro_f1_std	bal_acc_mean	bal_acc_std
0	LR	0.711154	0.118526	0.760811	0.120261
3	HGB+PCA(90%)	0.702619	0.066123	0.757207	0.069615
1	LR+PCA(90%)	0.682259	0.120925	0.736321	0.108132
2	HGB	0.658061	0.181828	0.713408	0.147955

```
Test-fold label counts (5 folds)
```

	fold	lying	sitting	standing	walking	running	cycling	Nordic walking	\
0	0	189	178	201	254	101	199	232	
1	1	200	184	181	189	174	174	186	
2	2	87	114	81	115	0	0	0	
3	3	185	207	186	254	98	186	213	
4	4	101	47	101	134	14	89	114	

	ascending stairs	descending stairs	vacuum cleaning	ironing	rope jumping
0	111	95	178	245	86
1	112	101	173	243	50
2	39	58	80	111	24

3	119	104	176	230	29
4	68	44	85	116	0

7. Results & error analysis

We summarize the modeling results using a small set of diagnostic plots focused on **performance stability** and **error structure**.

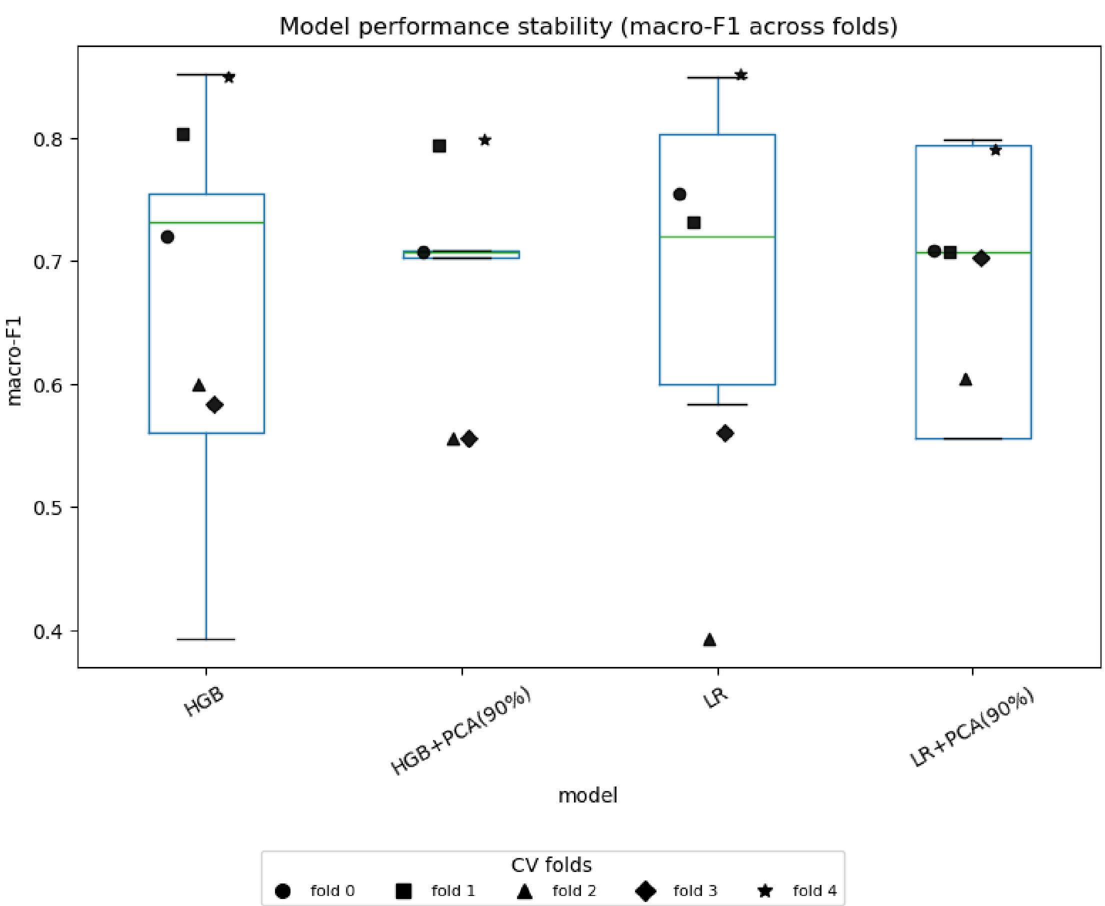
First, we compare models using their **fold-wise macro-F1 distributions**, which highlights both average performance and variability across subject-wise folds.

We then inspect the **aggregated confusion matrix** for the best-performing model to understand common confusions between activities.

Finally, we visualize **per-subject macro-F1 variability** for the best model to assess how performance differs across users.

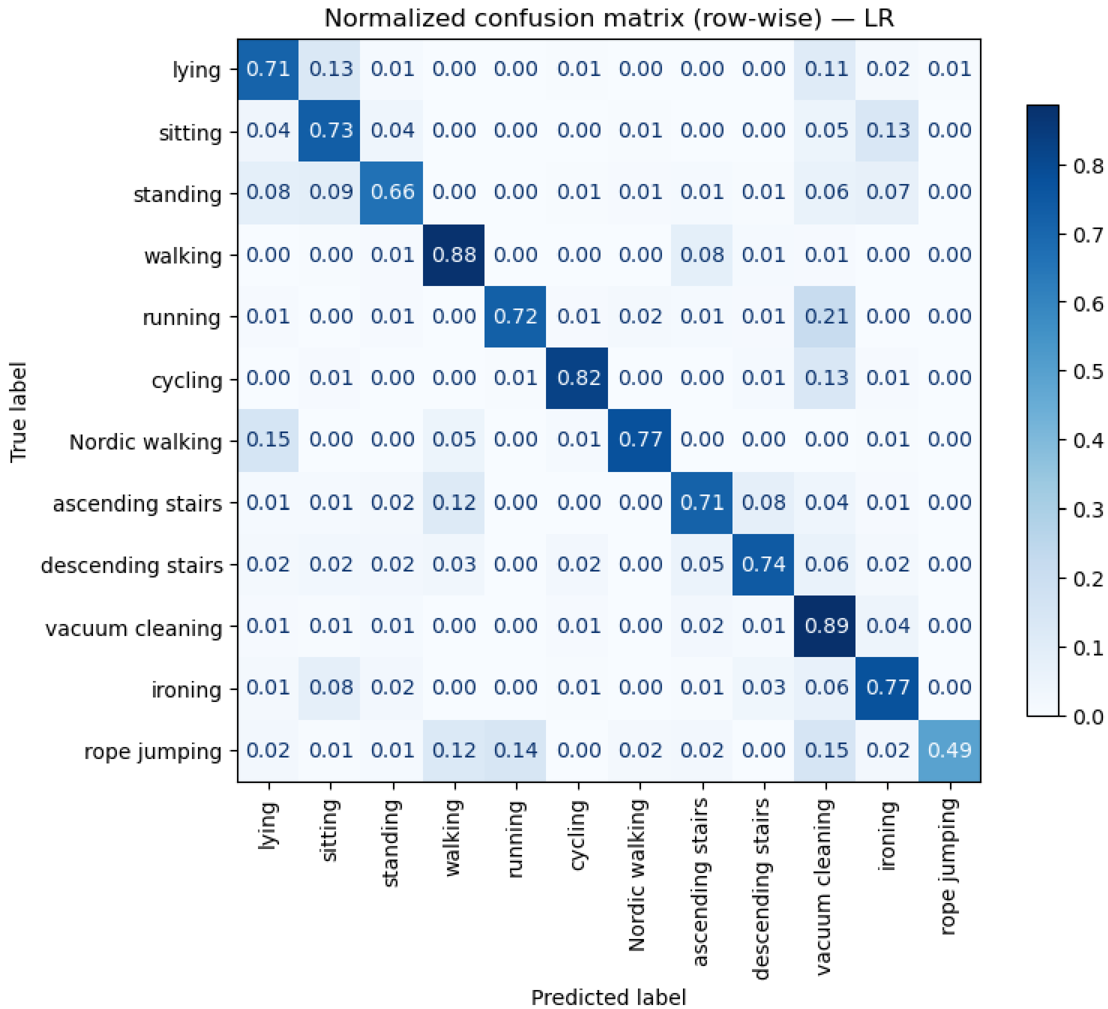
7.1 Model performance stability

We first compare models using their **fold-wise macro-F1 distributions** across subject-wise cross-validation splits. This visualization highlights not only average performance, but also **stability across folds**, which is important when evaluating generalization to unseen users.



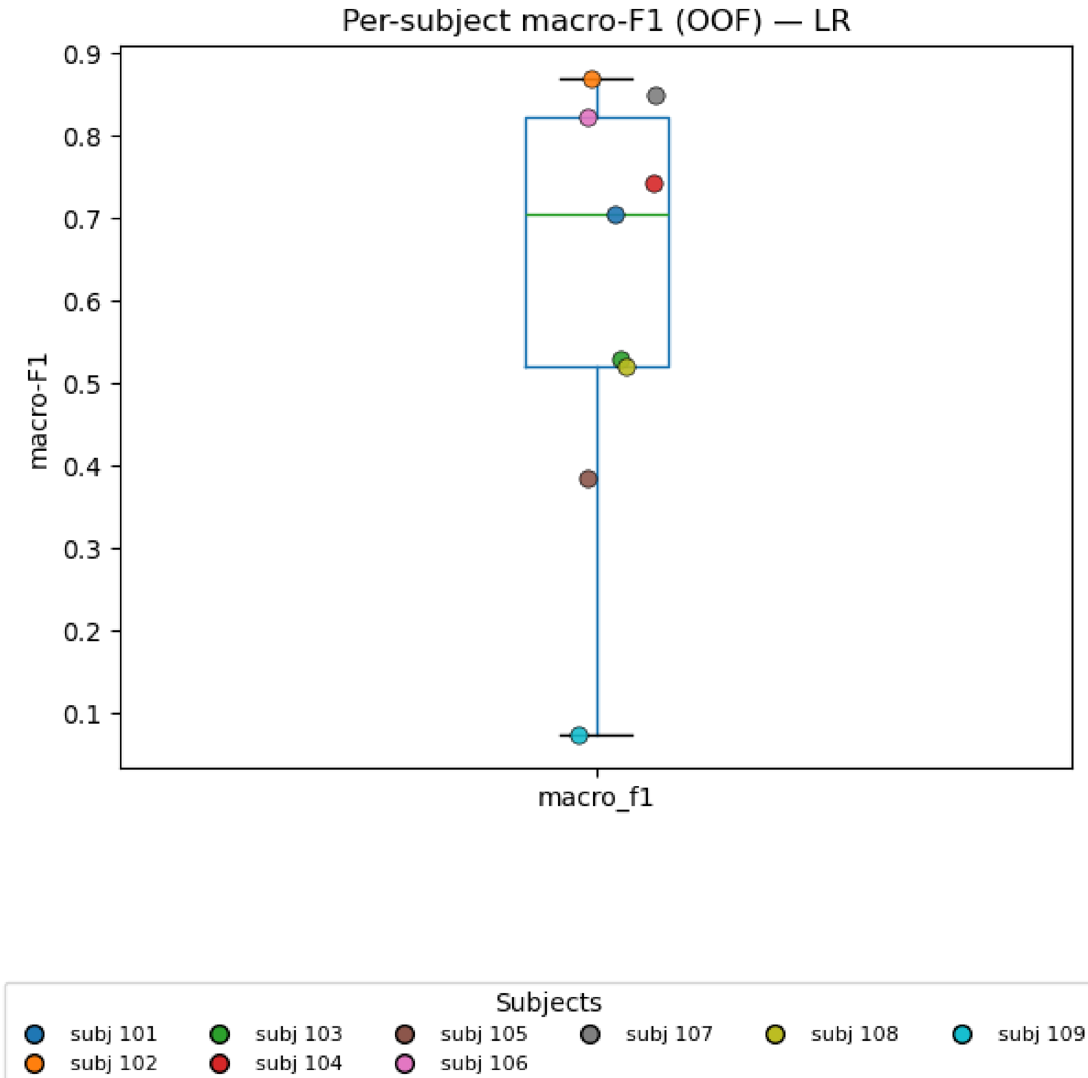
7.2 Aggregated confusion matrix

We inspect the **aggregated confusion matrix** for the best-performing model (based on mean macro-F1). This helps identify systematic confusions between activity classes and provides insight into common failure modes.



7.3 Per-subject performance variability

Finally, we analyze **per-subject macro-F1 performance** for the best model. This highlights how recognition accuracy varies across individuals, an important consideration for wearable deployment scenarios.



7.4 Results summary

Overall model comparison (GroupKFold, subject-wise):
- **Best mean performance: LR (original features)** achieved **macro-F1 = 0.71 ± 0.12** and **balanced accuracy = 0.76 ± 0.12**.
- **Most stable across folds: HGB+PCA(90%)** was close in mean performance (**macro-F1 = 0.70 ± 0.07**, **bal-acc = 0.76 ± 0.07**) with visibly tighter variability than the other variants.
- **PCA impact:** PCA slightly reduced LR performance on average (LR+PCA macro-F1 0.68), while HGB benefited from PCA in stability (HGB+PCA vs HGB).

Error structure (normalized confusion matrix, LR):
- Several activities show strong recall (row-wise): e.g., **walking (~0.88)**, **vacuum cleaning (~0.89)**, **cycling (~0.82)**.
- The hardest class in this subset is **rope jumping** (~**0.49** recall), with confusions spread across

high-motion classes (notably walking/running/vacuum cleaning).
- Some expected confusions appear among semantically/kinematically similar classes:
- **standing** mixes with **lying/sitting**.
- **running** shows noticeable confusion with **vacuum cleaning** in this split.

Per-subject variability (OOF macro-F1, LR):
- Performance varies substantially across users: from ~**0.87 (best subject)** down to ~**0.07 (worst subject)**.
- This highlights that while the global model generalizes reasonably on average, **user-level differences and intra-subject physiological variability** are a major source of uncertainty, motivating **per-user calibration or personalization** in practical wearable deployments.

8. Model interpretability in PCA space

In this final section, we build intuition about how the trained models separate activity classes by visualizing their **decision regions in PCA space**.

After cross-validation is complete, we refit the PCA-based pipelines **once on the full dataset**, using the same default settings as in Section 6. This refit is performed purely for **interpretability and visualization**, not for estimating performance.

Because the PCA-based models operate in a **higher-dimensional PCA space** (retaining 90% of variance), but decision boundary plots can only be shown in 2D, we visualize a **2D slice** of the full model in the **PC1–PC2 plane**:

- We create a grid in the **PC1–PC2** plane.

- We embed these grid points into the full PCA space by setting the remaining PCs (PC3 … PCk) to **0**, corresponding to their mean values in centered PCA coordinates.

- We map these points back to the original feature space via inverse transforms and obtain predictions from the **original trained model**.

This approach follows standard practice in applied machine learning when visualizing high-dimensional classifiers: PCA components are zero-centered by construction (see e.g. *Jolliffe & Cadima, 2016, "Principal Component Analysis: A Review and Recent Developments"*), so fixing higher components at zero corresponds to holding them at their average values. The inverse PCA transformation used here is the same mechanism described in the scikit-learn PCA documentation and textbooks on linear dimensionality reduction.

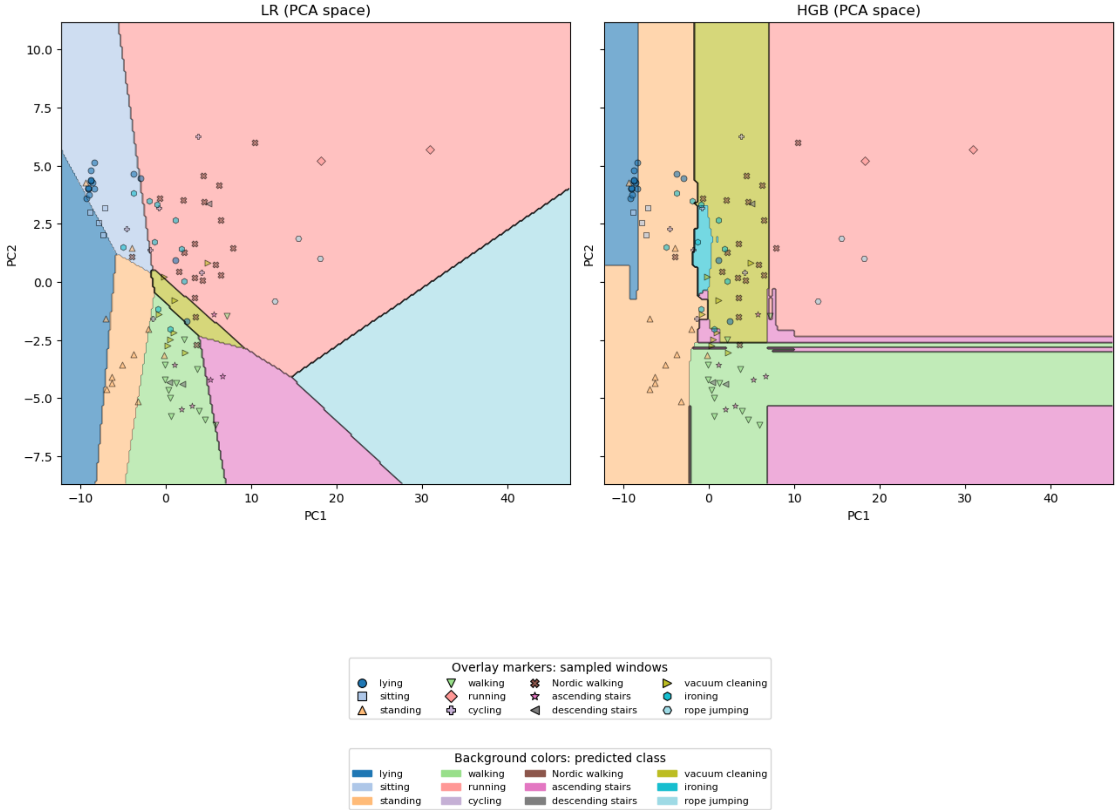
The resulting decision regions are therefore **faithful to the full PCA-based model**, but should be interpreted as model behavior along PC1–PC2 **with all other PCA directions held constant**. This kind of 2D slicing is widely used for interpretability and visualization, while performance is always evaluated in the full feature space or a sufficiently rich reduced space (e.g. 90–95% variance PCA), rather than in only two components.

To aid interpretation, we overlay:
- a random global subsample of windows

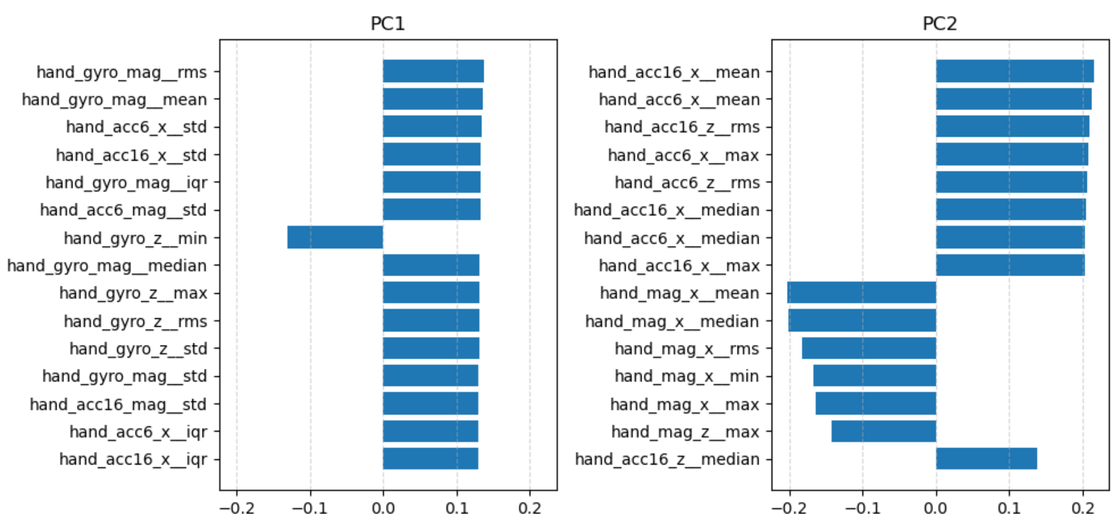
Finally, we inspect the **PC1 and PC2 loadings** (top absolute contributors of each independantly) to see which original features most strongly drive the dominant PCA direction used in the visualization.

8.1 Global overlay of random subsample of windows

```
PCA components kept: 17
```



8.2 PCA loadings



PCA loadings (signed; ranked by absolute magnitude)

8.3 Comments

The PCA-based visualizations provide qualitative insight into how the models structure activity separation, without serving as a performance assessment tool.

- **PC1–PC2 structure:**
  The first two principal components reveal some organization of activities.
- **Model behavior differences:**
  The Logistic Regression decision regions are smooth and globally linear in PCA space, while the Gradient Boosting model forms sharper, axis-aligned regions and stronger nonlinearity observed.
- **Overlap and ambiguity:**
  Significant overlap remains between related activities.
- **PCA loadings:**
  PC1 is dominated by **gyroscope magnitude and axis-specific statistics** (mean, RMS, IQR, std), with additional contribution from accelerometer magnitude features. This indicates that PC1 primarily captures **overall wrist motion intensity**, combining rotational and translational energy.
  PC2 shows strong contributions from **accelerometer axis statistics** (especially x/z mean, RMS, max, median), contrasted by **magnetometer features** with large negative loadings. This suggests PC2 separates **directional linear motion** from **orientation-related or heading-stability cues**.

Overall, these visualizations support the quantitative results: the models learn reasonable global structure, but **class overlap and subject-specific variability** remain fundamental challenges—motivating personalization or temporal modeling in more advanced setups.