

# GYM AI Tracker

Train your body & algorithme!



**Préparé par :**

*CHAKIR Fatima Ez-zahra*

**Supervisé par :**

*Mr Aziz Khamjane*



# Plan

## 01 Introduction

- Objectif
- Enjeu de données
- Inspiration

## 02 Prétraitement des données

- Compréhension des données
- Ajout des colonnes utiles
- Exportation en format pickle

## 04 Visualisation

- Tracé des colonnes
- Comparaison entre ensemble
- Analyse

## 05 Extraction des caractéristiques (Feature Engineering)

## 06 Modèle de prédiction

- Neural Network
- Random Forest
- Decision Tree
- KNN
- Naive Bayesien
- Comparaison

## 07 Modèle Personnalisé

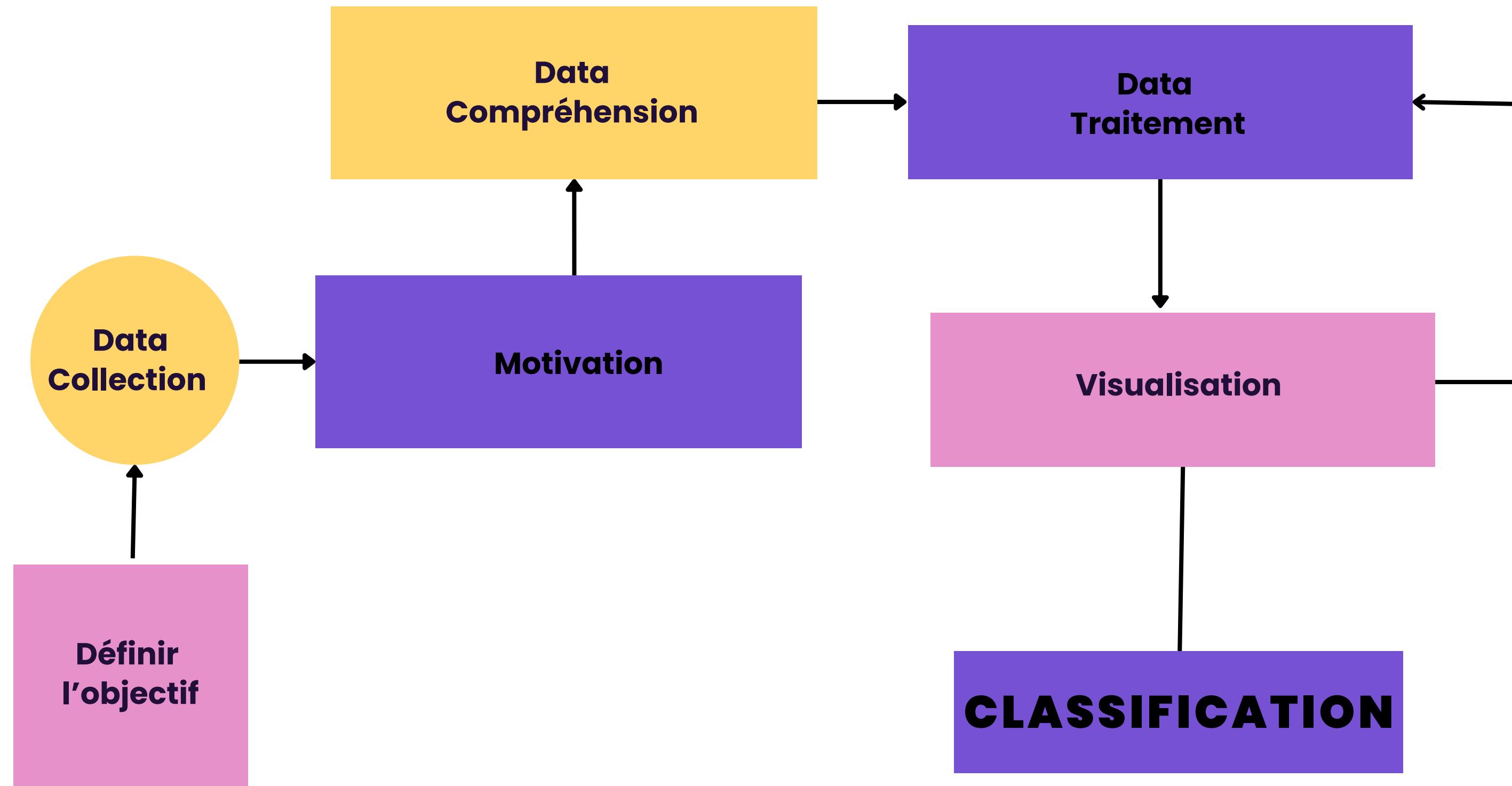
## 08 Discussion



01



# Projet Workflow



# Introduction

- **Objectif du projet**

Ce projet a pour objectif de créer des scripts Python pour traiter, visualiser et modéliser les données sensorielles des accéléromètres et des gyroscopes.

L'objectif ultime est de développer un modèle d'apprentissage automatique capable de classifier les exercices dans la salle du sport et de compter les répétitions.



Bench Press

Deadlift

Overhead Press

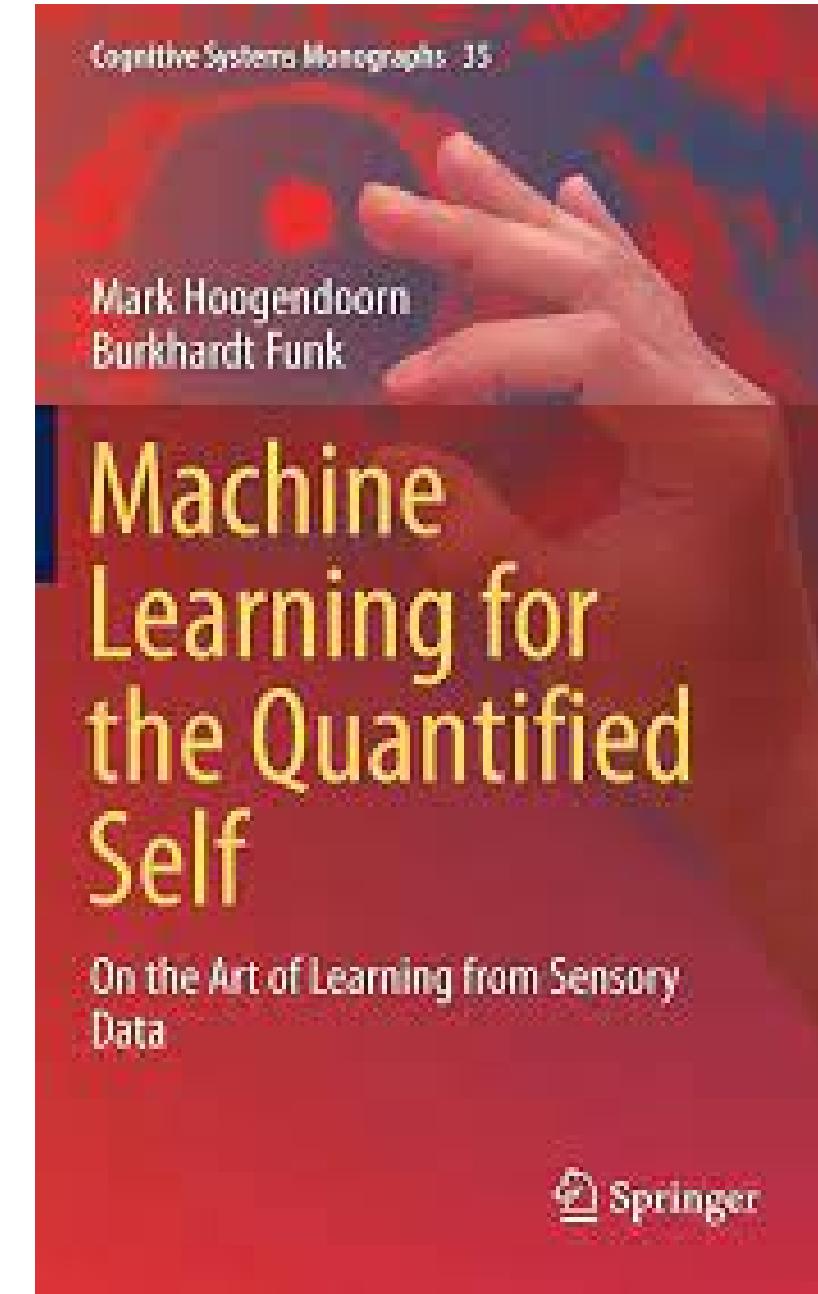
Barbell Row

Squat



## • Inspiration

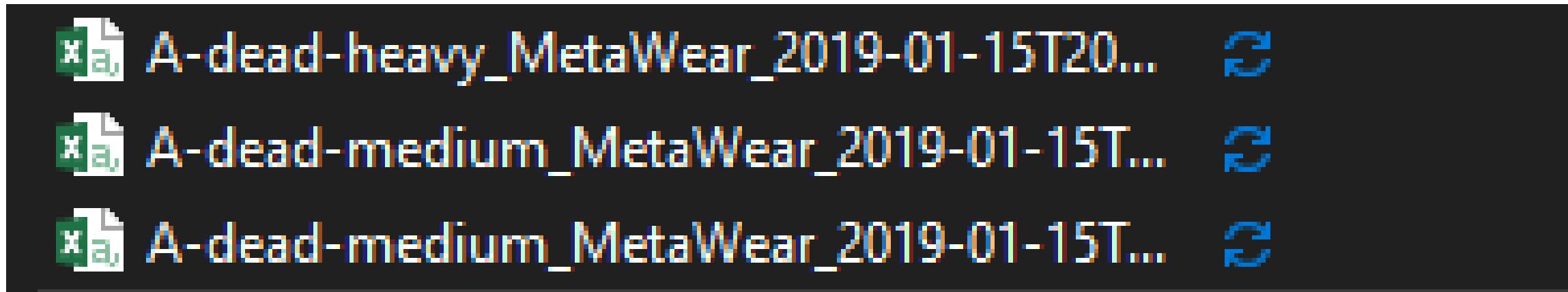
**Le quantified self** désigne toute personne engagée dans le suivi personnel de données biologiques, physiques, comportementales ou environnementales. Comme décrit par Hoogendoorn et Funk dans leur ouvrage "Machine Learning for the Quantified Self: On the art of learning from sensory data" (2018).



- **L'enjeu de données**

L'enjeu de données utilisé dans ce projet est pris à partir des annexes de livre mentionné au dessus, en effet cet enjeu est collecté par des capteurs lors d'un entraînement en salle de sport, pendant des sessions où cinq participants effectuaient divers exercices avec une barre (Figure précédente). Donc on va se servir des données sensorielles pour procéder notre projet.

NB : Les exercices sont : Bench, Dead, ohp, row, squat.



## • Motivation

La motivation de ce projet était le manque de concentration sur les programmes de force dans la littérature connexe et le soutien des traqueurs d'activité actuels.

Et vue que la majorité des suivies étaient sur L'activité sanguine, cardiaque ou respiratoire...

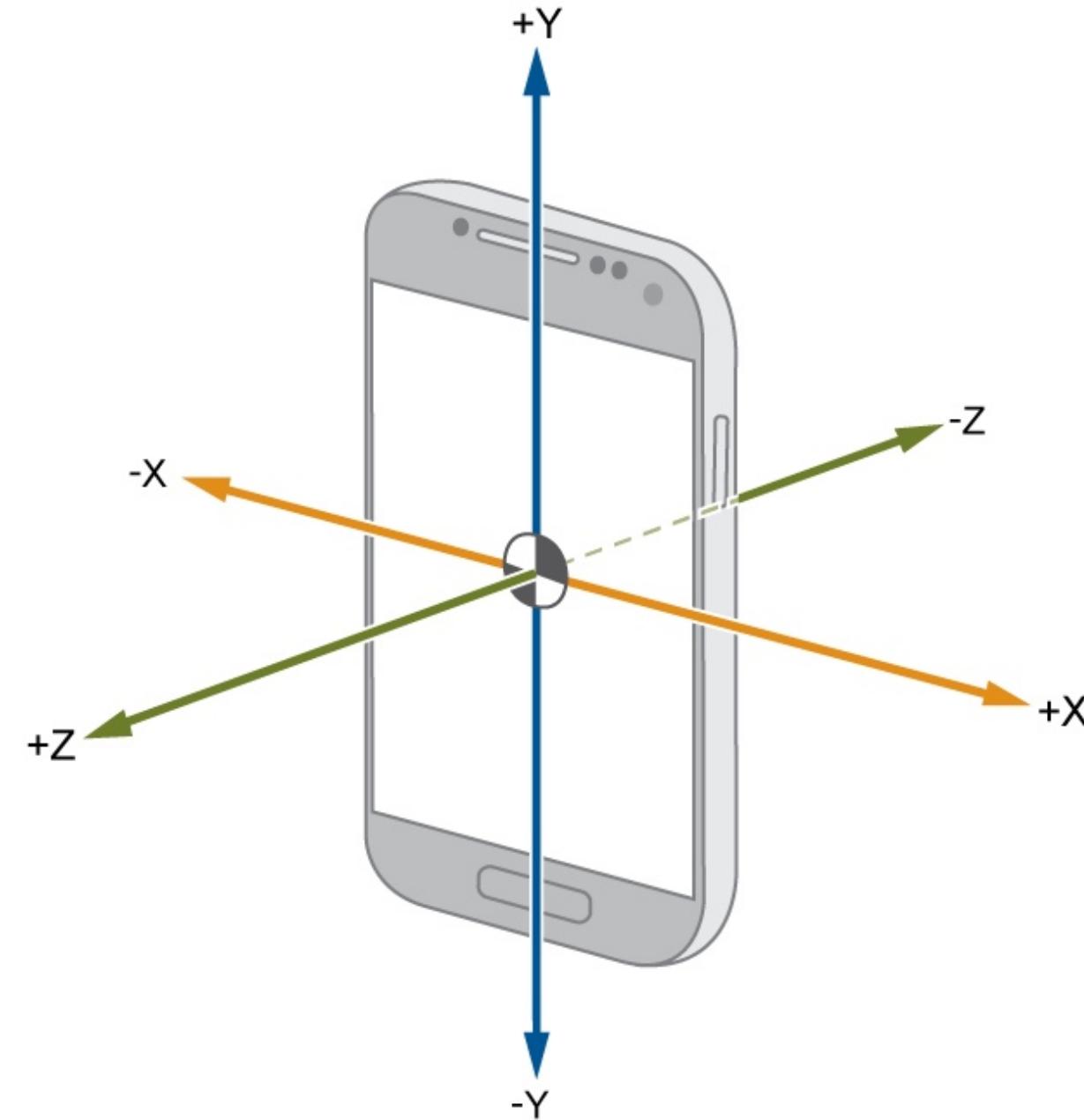


# Pré Traitement

- **Compréhension des données**

## **Données Accéléromètre (G-force) :**

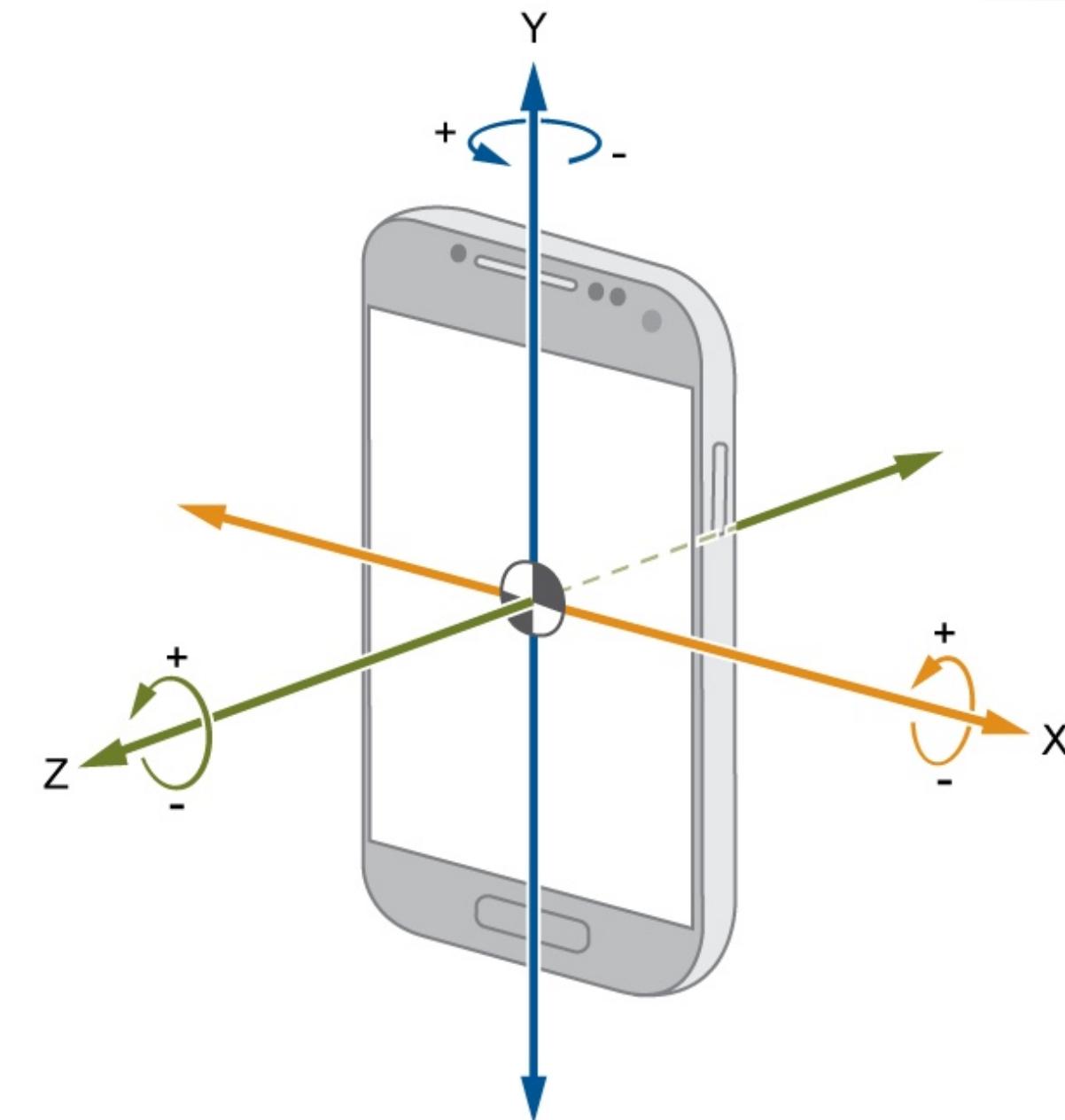
Dans le contexte de notre projet, les données de l'accéléromètre sont mesurées en unités de force G, où  $1\text{ G} = 9.8\text{ m/s}^2$ . Cette mesure, exprimée en "g," représente l'accélération gravitationnelle standard et sert de référence pour quantifier les forces appliquées sur l'axe de l'accéléromètre.



- **Compréhension des données**

### **Données Gyroscope (Degré/s):**

Les données du gyroscope sont mesurées en unités de degrés par seconde ( $^{\circ}/s$ ). Chaque valeur enregistrée dans les données du gyroscope représente la vitesse angulaire autour d'un axe spécifique, exprimée en degrés parcourus par seconde.



- Extraction des caractéristiques à partir des noms des fichiers

Exemple d'un nom de fichier csv:

A-bench-heavy\_MetaWear\_2019-01-  
14T14.22.49.165\_C42732BE255C\_Acceler  
ometer\_12.500Hz\_1.4.4.csv

☒ **Participant:** A  
☒ **Label:** Bench  
☒ **Category:** Heavy



- **Ajout des colonnes utiles**

L'ajout des colonnes importantes à partir de l'extraction des caractéristiques des noms des fichiers: participant, category (medium ou heavy), label (squat, row, dead, ohp, bench) et set (numéro de série).

```
df["participant"] = participant  
df["category"] = category  
df["label"] = label  
df["set"] = acc_set
```



- **Exportation des données en format pickle**

Le DataFrame est sauvegardé au format binaire pickle, facilitant le stockage temporaire des données transformées pour une reprise rapide sans réexécution complète du traitement. La restauration ultérieure des données se fait simplement avec pd.read\_pickle().

# Visualisation

En se basant sur dataset résultante dans la partie précédente,  
La figure 1 montre les données de l'accéléromètre pour un ensemble lourd de  
chaque exercice. La figure montre que les modèles d'exercice sont assez uniques et  
que les répétitions peuvent facilement être identifiées à partir des extrema.

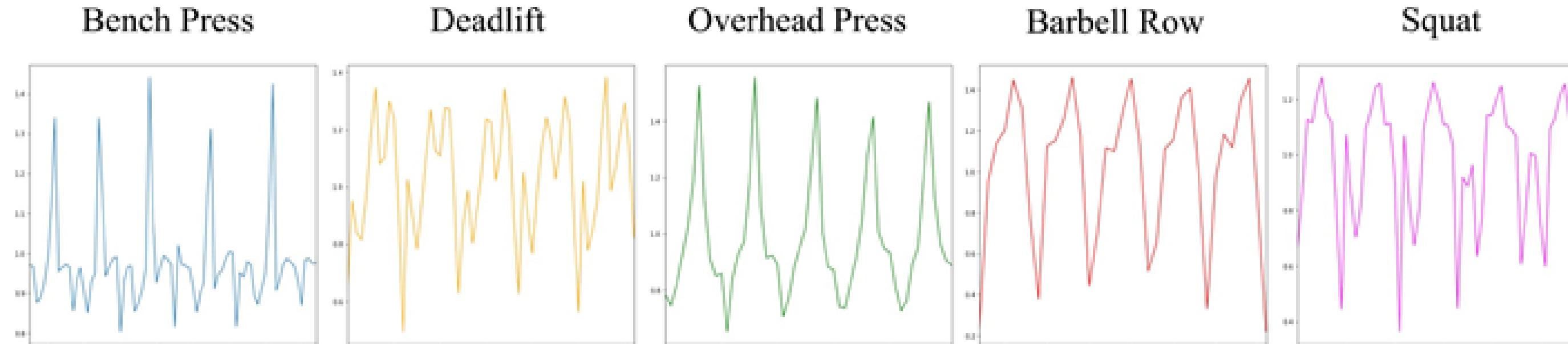


Figure 1: Données Accélèromètre



# Visualisation

La figure 2 montre les différences d'accélération "y" entre les ensembles de squat moyen et lourd. Les ensembles de poids moyen ont des pics plus élevés considérant que les ensembles lourds ont des gouttes plus profondes. Cela a du sens en tant que poids moyen, les répétitions montent plus rapidement en raison de la résistance inférieure tandis que les répétitions lourdes tombent plus rapidement en raison de la lourde charge.

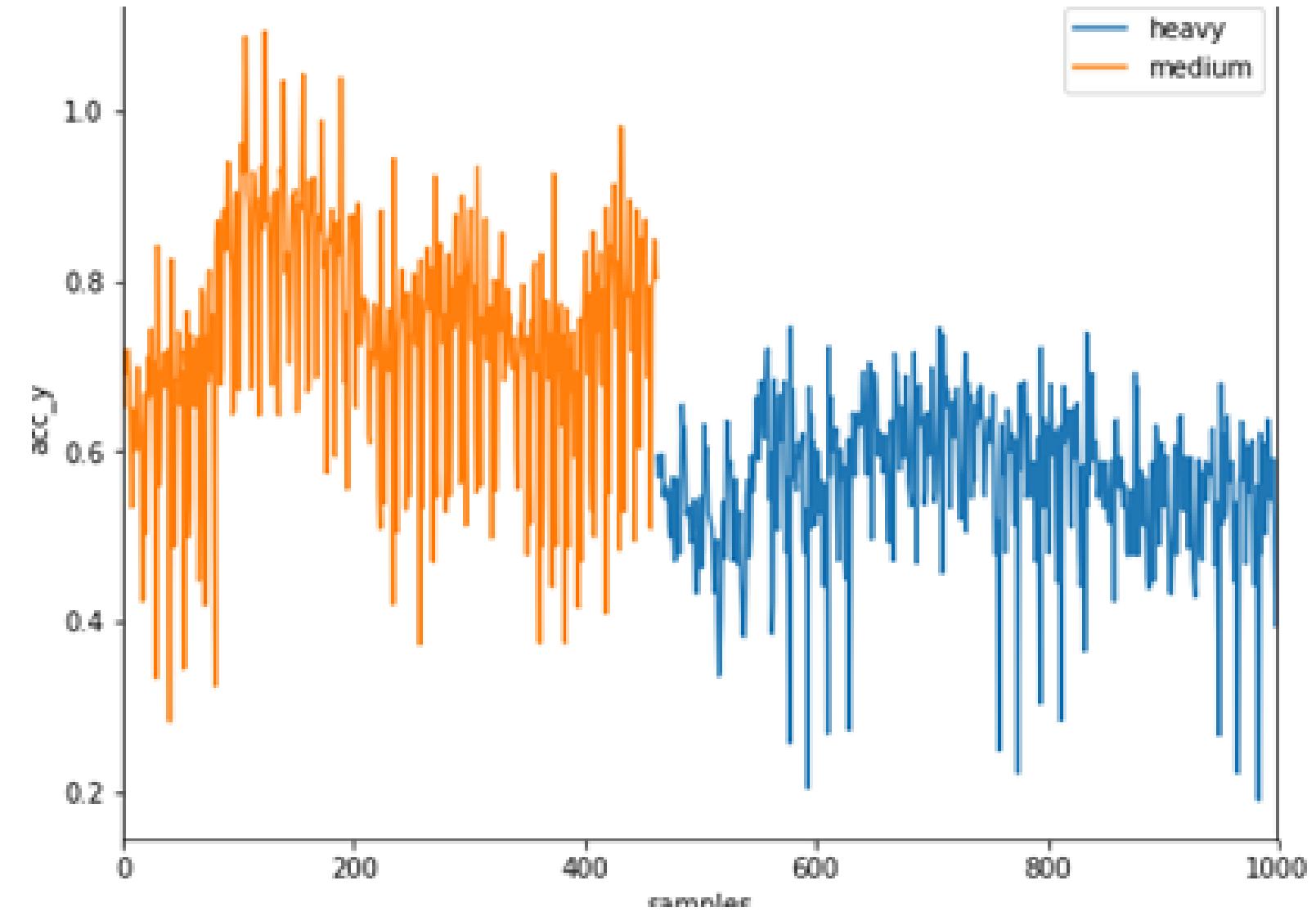


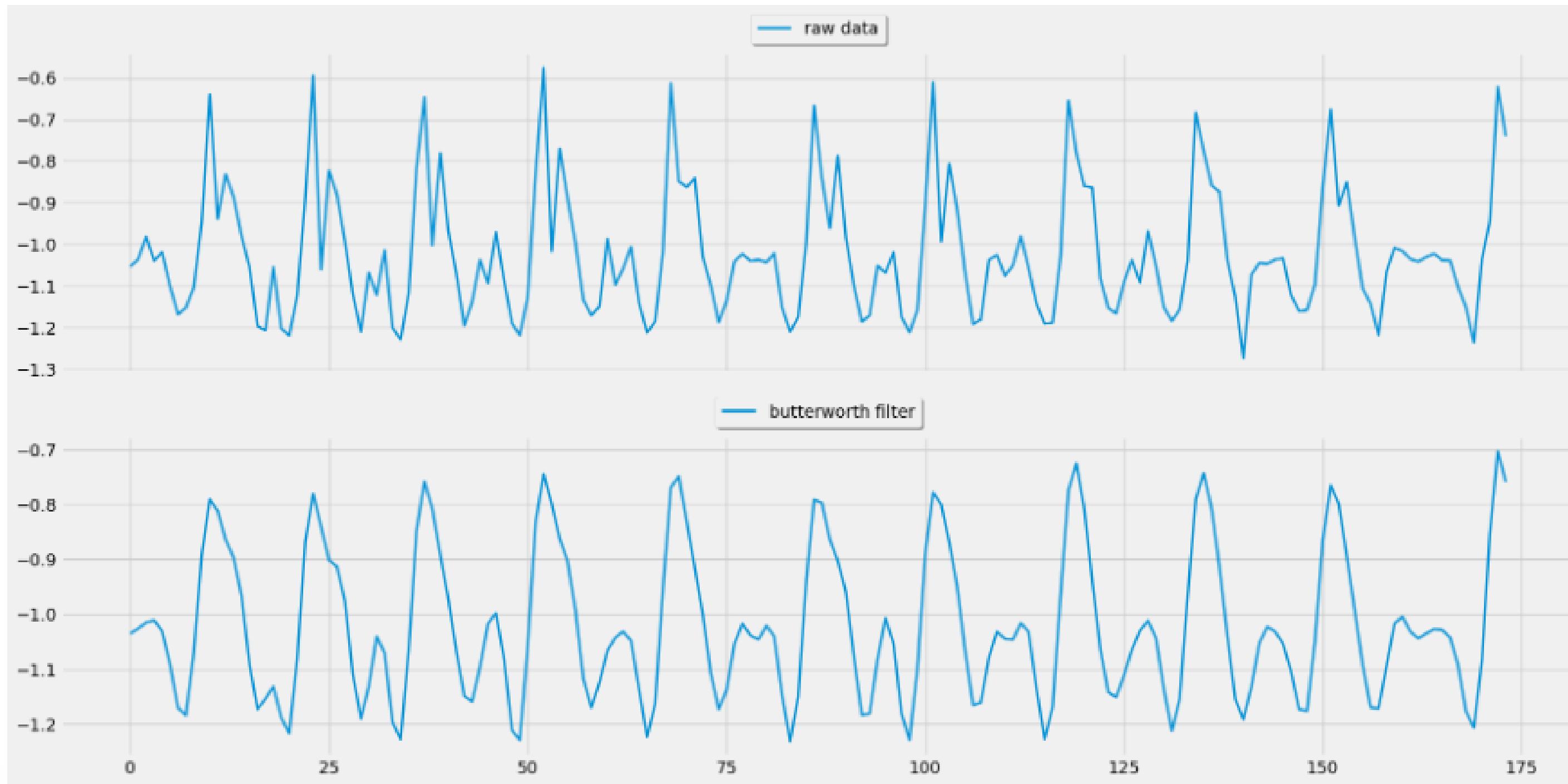
Figure 2: Heavy & Medium squat



# Extraction des caractéristiques (Feature Engineering)

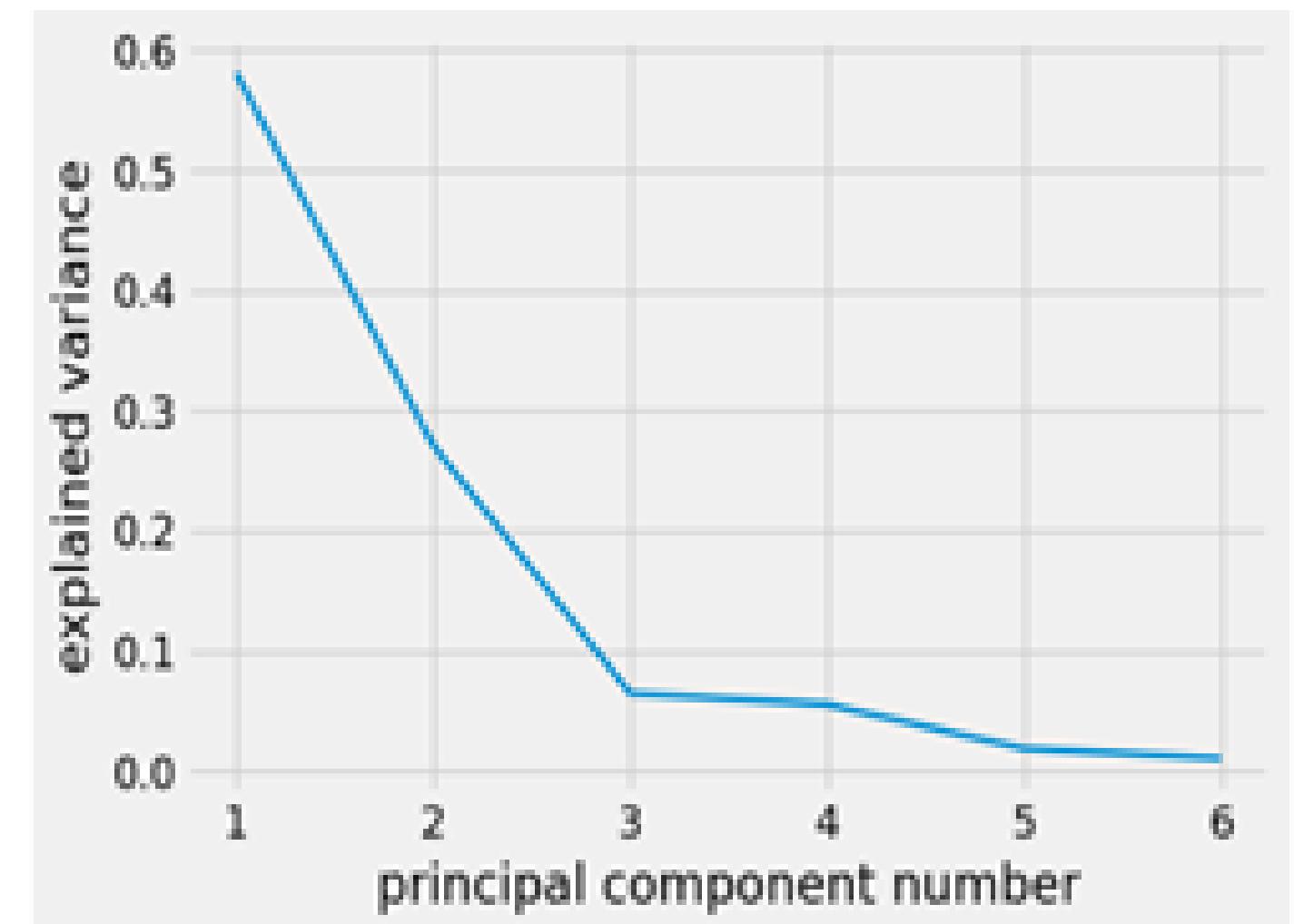
- **Filtre Passe-Bas**

Le filtre passe-bas peut être appliqué aux données de nature temporelle et suppose qu'il existe une forme de périodicité. Le filtre passe-bas Butterworth peut supprimer une partie du bruit à haute fréquence dans l'ensemble de données qui pourrait perturber le processus d'apprentissage. Le filtre a été appliqué à toutes les attributs sauf la cible.(label)



- **Analyse des composantes principales**

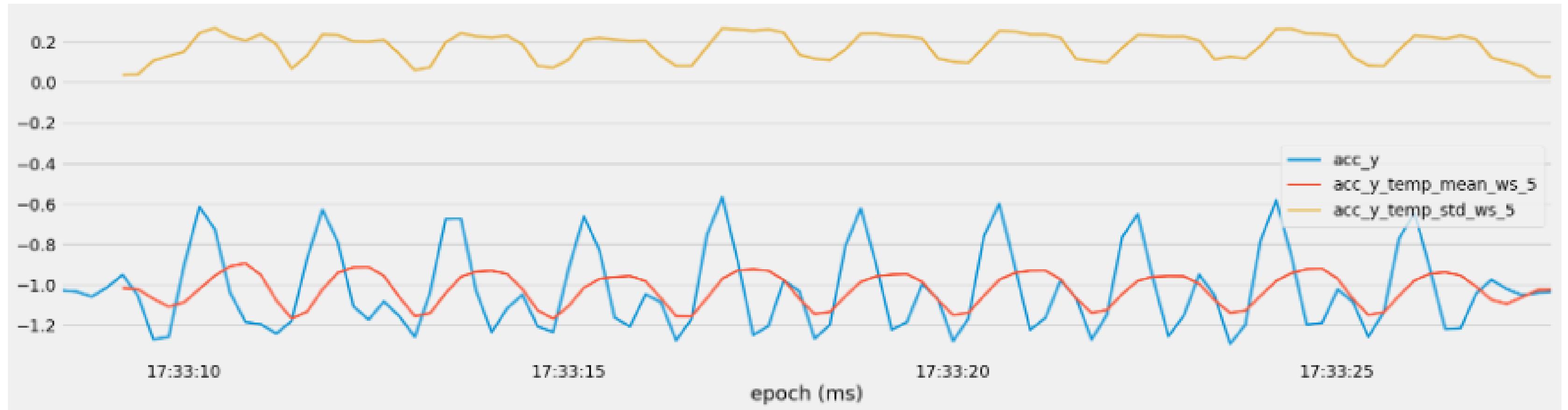
Une analyse en composantes principales (ACP) a été effectuée pour trouver les caractéristiques qui pourraient expliquer la majeure partie de la variance. ACP a été appliqué à toutes les fonctionnalités à l'exclusion des colonnes cibles. Les résultats sont visualisés dans la figure qui montre que la variance expliquée diminue considérablement après 3 composantes. (Méthode Elbow)



- **Abstraction temporelle**

Pour exploiter la nature temporelle des données, les points de données numériques ont été agrégés en prenant l'écart type (sd) et la moyenne de toutes les caractéristiques sauf les colonnes cibles sur différentes tailles de fenêtre. La sd devrait saisir la variation dans les données au fil du temps. Par exemple, le sd devrait être plus élevé pendant les exercices que pendant les périodes de repos. La moyenne temporelle indiquera plus sur les niveaux généraux des données, avec moins d'influence du bruit pointu.





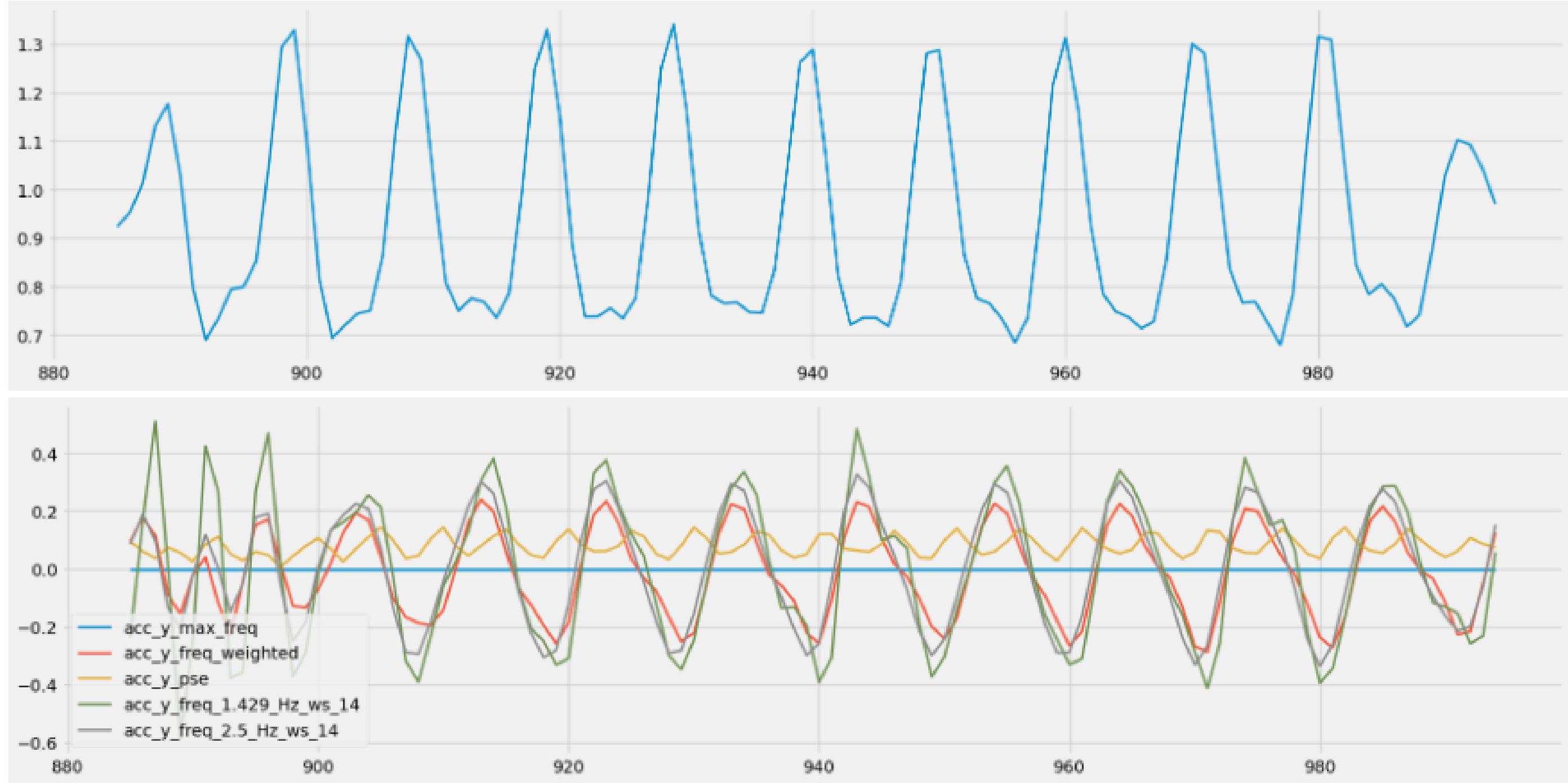
- **Transformation de Fourier**

Outre l'abstraction temporelle, le domaine fréquentiel sera également exploré. L'idée d'une transformation de Fourier est que toute séquence de mesures peut être représentée par une combinaison de fonctions sinusoïdes avec des fréquences différentes. (voir figure au dessous)

### **Attributs ajoutés:**

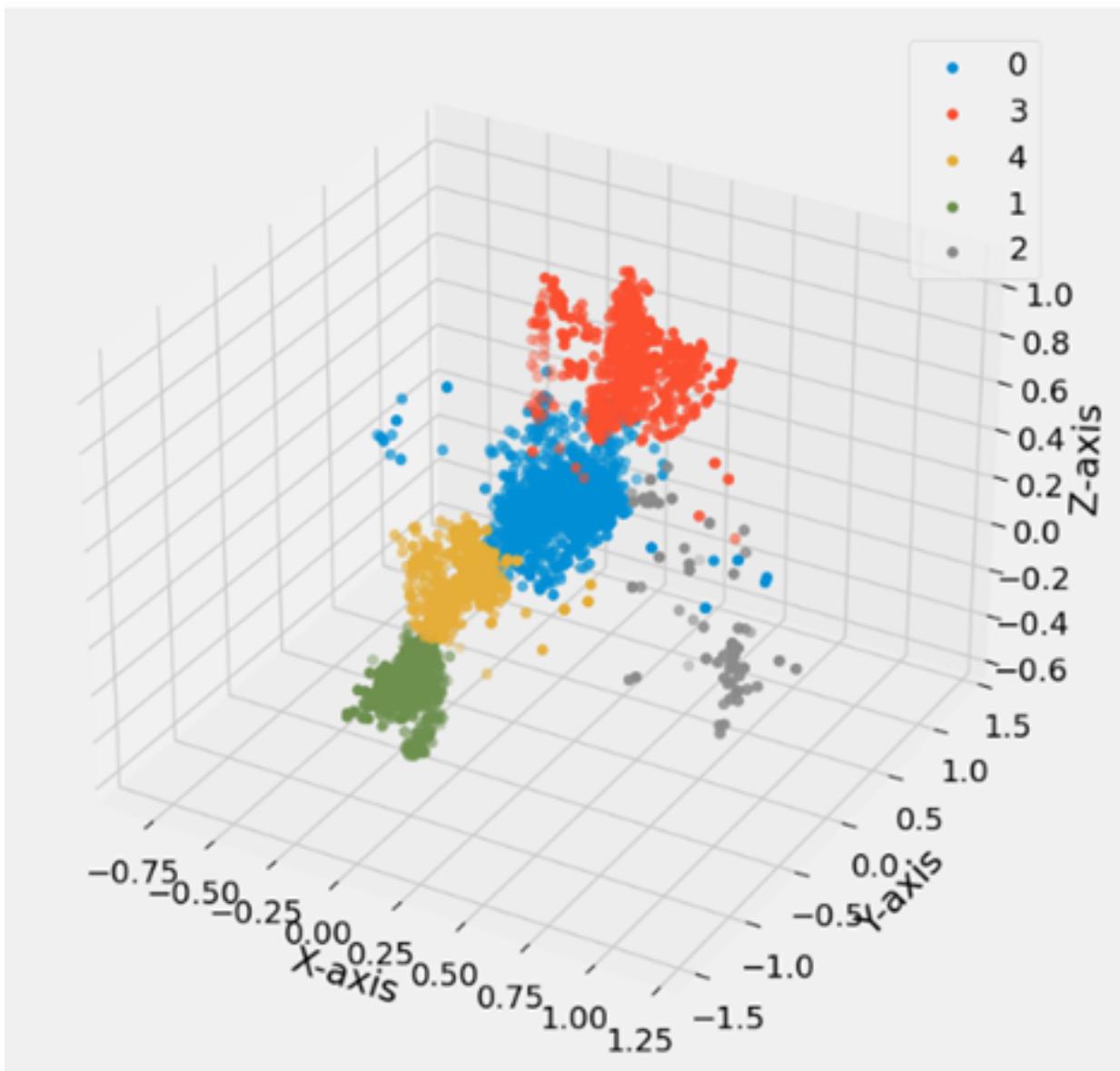
- la fréquence maximale
- la moyenne pondérée
- l'entropie spectrale de puissance .



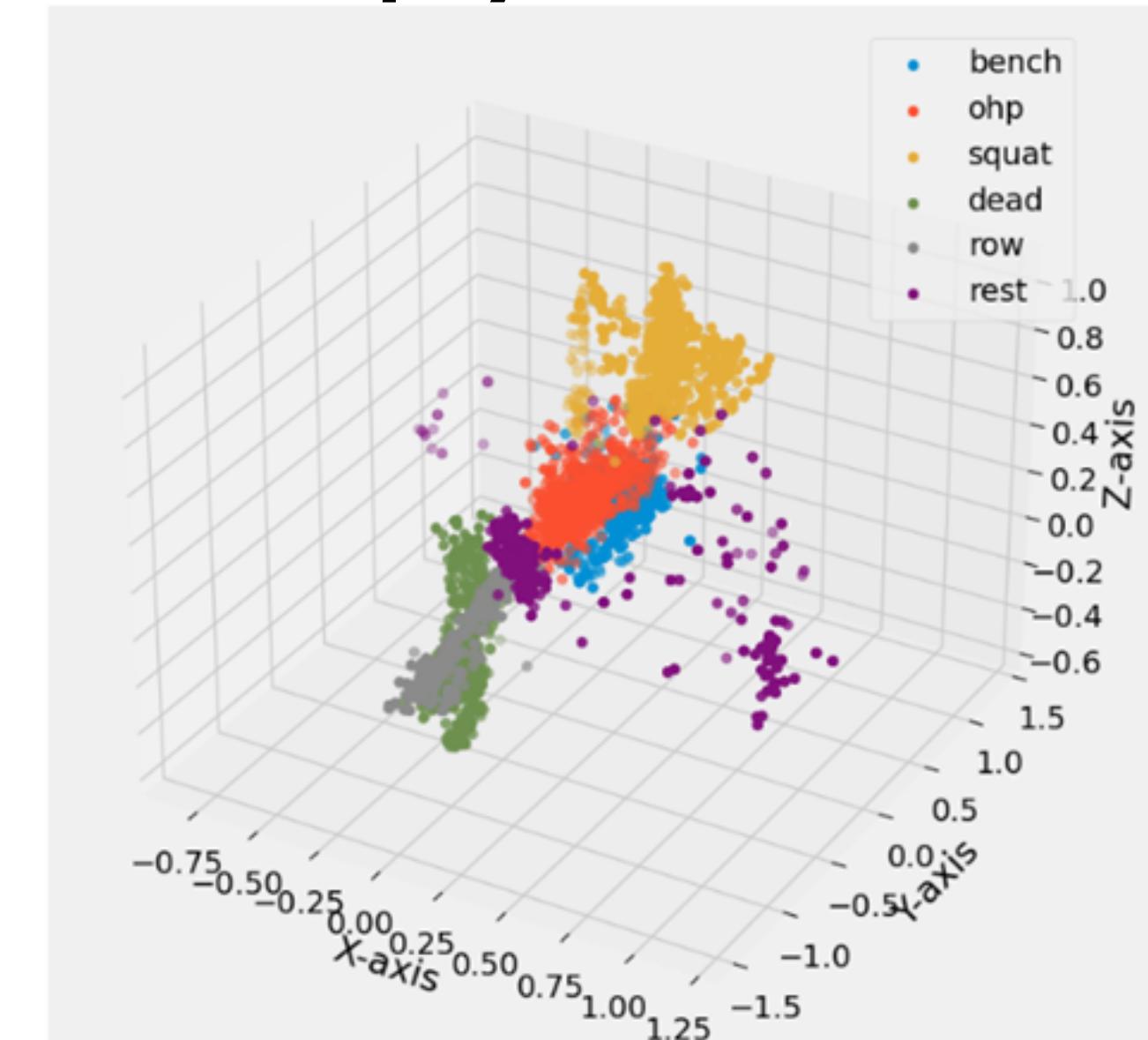


- Clustering

K=5



Groupby(label)



Le tableau montre la répartition des mesures. Le groupe 1 couvre presque toutes les données de Bench et over head press. Cela est logique, car les deux mouvements de pression sont très similaires. Le squat est capturé dans le cluster 2 et le dead et row sont capturés dans le cluster 3 avec un score parfait. Le cluster 4 contient certaines des données restantes, mais ne parvient pas à les capturer avec précision.

<b>Label</b>	<b>Cluster 1</b>	<b>Cluster 2</b>	<b>Cluster 3</b>	<b>Cluster 4</b>
Bench-press	99.88 %	0.12 %	0.00 %	0.00 %
Deadlift	0.00 %	0.00 %	100.00 %	0.00 %
OHP	99.28 %	0.72 %	0.00 %	0.00 %
Row	0.00 %	0.00 %	100.00 %	0.00 %
Squat	2.98 %	97.02 %	0.00 %	0.00 %
Rest	4.14 %	3.78 %	50.45 %	41.62 %



# Modèles de prédiction

L'ensemble de données est maintenant traité et prêt pour la formation. On va entraîner 5 algorithmes; Random Forest, Neural Network, Decision Tree, KNN et Naive Bayesin.



- **Forward Selection**

En utilisant un arbre de **décision simple**, et en ajoutant progressivement les meilleures caractéristiques qui contribuent le plus aux performances, car des fonctionnalités inutiles pourraient affecter les performances des algorithmes.

Forward Selection reçoit comme paramètres :

\***X\_train et y\_train**

\***max\_features = 10** (nombre maximal des colonnes qu'on veut)



- **Forward Selection**

```
•selected_features = [  
    'acc_z_freq_0.0_Hz_ws_14',  
    'acc_x_freq_0.0_Hz_ws_14',  
    'duration',  
    'gyr_r_temp_mean_ws_5',  
    'acc_y_temp_mean_ws_5',  
    'pca_2',  
    'gyr_x_temp_std_ws_5',  
    'acc_z_freq_2.143_Hz_ws_14',  
    'pca_1',  
    'gyr_x_freq_2.5_Hz_ws_14']  
  
•ordered_scores = [  
    0.885556704584626,  
    0.9903481558083419,  
    0.9989658738366081,  
    0.9993105825577387,  
    0.9996552912788693,  
    0.9996552912788693,  
    0.9996552912788693,  
    0.9996552912788693,  
    0.9996552912788693]
```



# Random Forest

**GridSearch** pour des meilleurs hyperparamètres!

- **Criterion** : Entropy
- **min\_samples\_leaf** : 2
- **n\_estimators** : 50



# Random Forest

## Matrice de confusion

- **TPR** : 98.29%
- **TNR**: 99.74%
- **Accuracy**: 99.48%

		Confusion matrix						
		bench	dead	ohp	rest	row	squat	
True label	bench	173	0	2	0	0	0	
	dead	0	172	0	0	0	0	
ohp	3	0	172	0	0	0	0	
rest	0	0	0	128	0	0		75
row	0	0	0	0	142	0		50
squat	0	0	0	0	0	175		25

# Existe il un Overfitting ?

Sur l'ensemble de Train

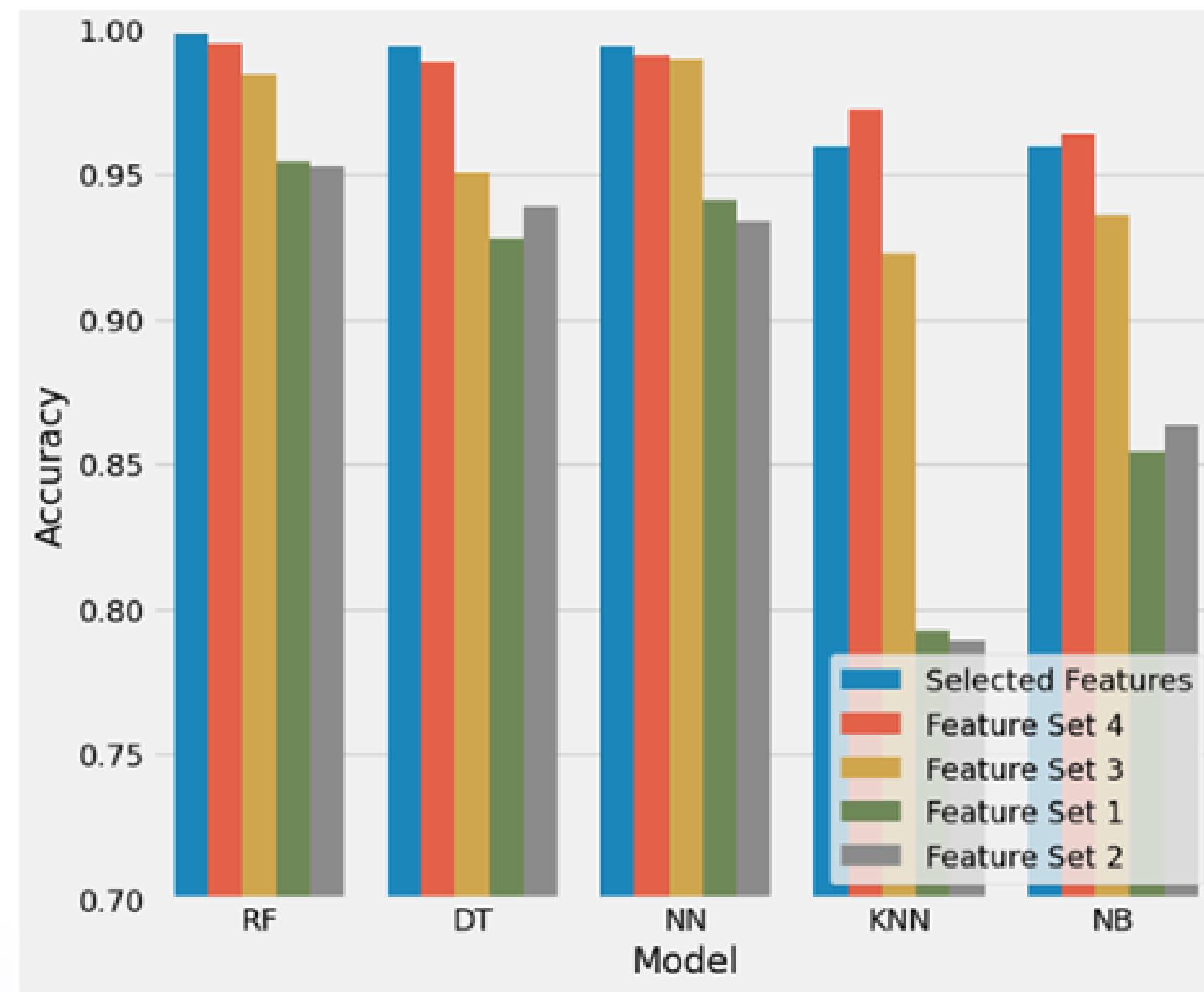
	model	feature_set	accuracy
1	RF	Selected Features	0.999655
3	DT	Selected Features	0.999655
3	DT	Feature Set 4	0.998966
3	DT	Feature Set 3	0.995863
1	RF	Feature Set 1	0.995174
0	NN	Selected Features	0.995174

Sur l'ensemble de test

	model	feature_set	accuracy
1	RF	Feature Set 4	0.996898
0	NN	Feature Set 4	0.996898
1	RF	Selected Features	0.995863
0	NN	Selected Features	0.994829
3	DT	Selected Features	0.993795
3	DT	Feature Set 4	0.992761



# Le même procédure pour les autres modèles!



## Analyse:

On a RF et NN et DT sont un peu plus proches, Donc on a choisi Random Forest le modèle finale et efficace pour classifier nos exercices du sport.

# Si le modèle apprend sans comprendre ?

```
#training data without the participant A's infos and the label
participant_df = df.drop(["set", "category"], axis = 1)
x_train = participant_df[participant_df["participant"] != "A"].drop(["label"], axis = 1)
y_train = participant_df[participant_df["participant"] != "A"]["label"]

#test data with the participant A's infos on the Label
x_test = participant_df[participant_df["participant"] == "A"].drop(["label"], axis = 1)
y_test = participant_df[participant_df["participant"] == "A"]["label"]
```



# Généralisation du modèle

## Résultat De Random Forest

**Accuracy** = 99.14%

Cet algorithme RF conviendrait à chaque participant, quel qu'il soit.

		Confusion matrix						
		bench	dead	ohp	rest	row	squat	
True label	bench	111	0	0	0	0	0	300
	dead	0	240	0	0	0	0	250
ohp	11	0	288	0	0	0	0	200
rest	0	0	0	342	0	0	0	150
row	0	0	0	0	29	0	0	100
squat	0	0	0	0	0	271	0	50
	bench	dead	ohp	rest	row	squat		0
	Predicted label							



# Comptage des répétitions

## Creation d'une fonction count\_rep()

### Signature :

```
def count_rep(dataset, cutoff = 0.4, order = 10, column = "acc_r"):
```

- Filtre pass-bas avec différents cutoff pour chaque exercice
- Détection des pics avec argrelextrema()
- Tracé du signal filtré et des pics
- Nombre de répétitions en retourne



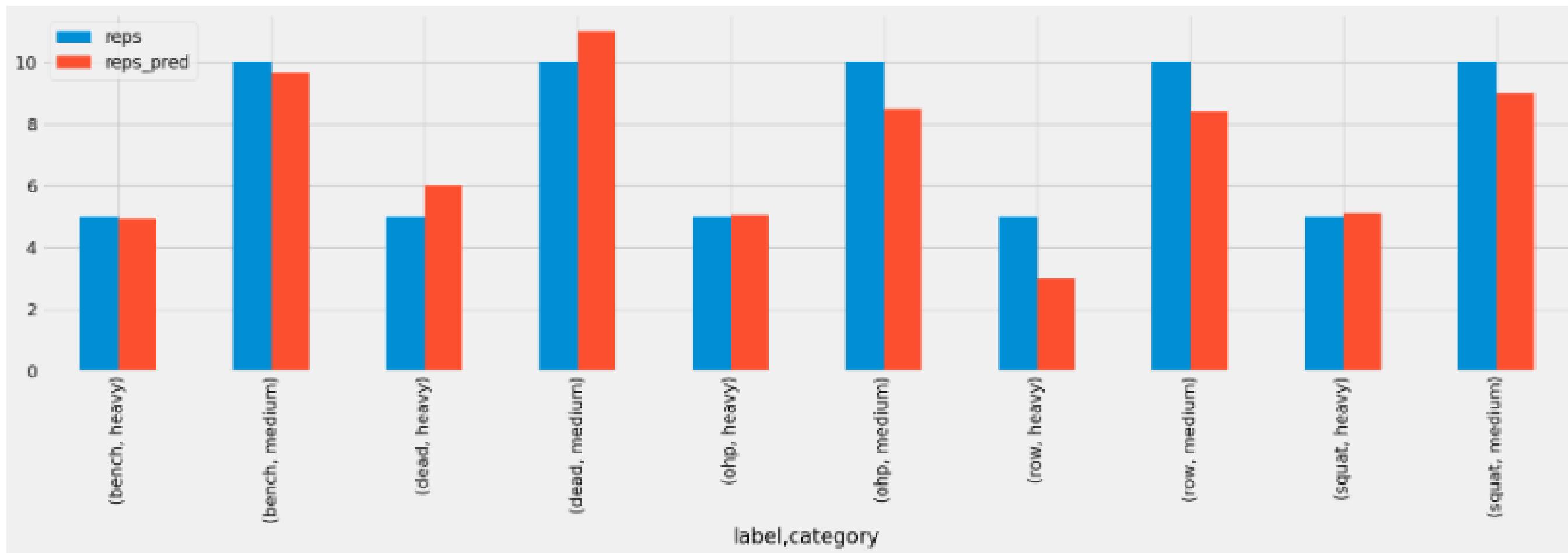
- **Benchmark dataset résultante!**

	label	category	set	reps	reps_pred
0	bench	heavy	1	5	5
1	bench	heavy	2	5	5
2	bench	heavy	3	5	5
3	bench	heavy	4	5	5
4	bench	heavy	30	5	4
...					

**PS:** La colonne “reps” est calculée par l’application d’un groupby() et max() pour chaque set et label



## • Comparaison!



**MAE = 0.93**

Rappel:  $MAE = \frac{1}{N} \sum |y_i - \hat{y}_i|$

# Discussion!

Le fait qu'on a seulement une précision positive de **98%** revient à dire que notre algorithme a des limitations.  
Quelles sont alors ces limitations?

Comme les exercices **ohp** et **bench** se ressemblent au niveau de la position des mains et de corps, aussi pour l'exercice **dead** et **row** qui se ressemblent au niveau d'inclination du corps . Donc on peut clairement justifier la baisse de **TPR= 98%** causé par la classification des 3 exercices **ohp** comme exercice type **bench**.(Voir matrice de confusion)



# Références

[1] Le livre : "Machine Learning for the Quantified Self: On the art of learning from sensory data" (2018).

<https://link.springer.com/book/10.1007/978-3-319-66308-1>

<https://ml4qs.org/t>

[2] Github page de meme livre :

<https://github.com/mhoogen/ML4QS/>

[3] Understanding Pickle files

<https://www.geeksforgeeks.org/understanding-python-pickling-example/>

[4] Temporal Abstraction (Rolling mean & std).

<https://datagy.io/rolling-average-pandas/>

[5] Transformation de Fourier

[https://en.wikipedia.org/wiki/Discrete\\_Fourier\\_transform](https://en.wikipedia.org/wiki/Discrete_Fourier_transform)

[6] Elbow method

<https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/>



**Merci ;)**