Big data analytics

Project

January, 2026

Konstantinos Tsampras

UP1083865

Supervisor: Vasileios Megalooikonomou

Contents

1. Introduction

The project assigned during the 2025-2026 class of Big Data Analytics revolves around the analysis and knowledge extraction from two datasets.

The first dataset contains clinical data of participants (such as fried, bmi, age, times for exercises etc) and the other contains timestamps of each participant entering a new room in their house.

part_id	ts_date	ts_time	room
2113	20170920	9:25:48	Kitchen
2113	20170920	9:26:03	Entrance
2113	20170920	9:29:25	Entrance
2113	20170920	9:53:09	Bathroom
2113	20170920	9:57:43	Bedroom
2113	20170920	9:58:51	Entrance
2113	20170920	9:59:03	Kitchen

Figure 1 Example of the beacons dataset

parameter	Description
fried	Categorization by Fried
hospitalization_one_year	Number of hospitalizations in the last year
hospitalization_three_years	Number of hospitalizations in the last three years
ortho_hypotension	Orthostatic hypotension detection
vision	Vision
audition	Audition
weight_loss	Unintentional weight loss
exhaustion_score	Self-reported exhaustion
raise_chair_time	Lower limb strength
balance_single	Single foot station (Balance)

Figure 2 Example fields of the clinical dataset

Based on these two datasets we are going to attempt to predict frailty and attempt to make connections to it and everyday actions.

That process was done using the Python language and certain libraries. The link to the repository containing all the code and the dependencies is this. The directory of each script, input and output file will be denoted when referred to.

2. Part A

Part A focuses on the preprocessing of the clinical dataset and classification of participants in the fried category.

2.1 A.1

The preprocessing part is handled by the script PartA/clinical_preprocessing.py.

It uses libraries such as pandas (to load the file /Raw Data/clinical_dataset_commas.csv) and numpy for quick data manipulation.

After loading the dataset, it proceeds to categorical mapping, where it converts human-readable labels and sentinel values into machine readable numerical values. For example, in the gender field “M” is converted to 0 and “F” to 1, while values like 999 or “test non realizable” are turned into NaN.

The next step is to deal with NaN values. The standard approach of using the column median is used to fill those values.

Finally, the pre-processed dataset is outputted as “/Preprocessed Data/clinical_dataset_preprocessed.csv”.

2.2 A.2

The classification procedure is split into two files: “PartA/classification_nn.py” and “PartA/classification_rf”. The former uses a neural network while the latter uses a random forest process. Both programs classify participants into one of three categories (frail, pre-frail, non-frail) based on their clinical data exempting the fields that are used to calculate frailty (namely weight_loss, exhaustion_score, gait_speed_slower, grip_strength_abnormal, and low_physical_activity).

Neural network:

The procedure for the neural network classification involves removing the above-mentioned fields and splitting the dataset, 80% for training and 20% for testing. Next up we use a standard scaler to force the values to have a median of 0 and a standard deviation of 1 to ensure all attributes contribute equally to the optimization.

The actual model consists of 4 layers:

- 1 input layer matching the number of input features
- 1 hidden layer of 16 neurons with ReLU activation
- 1 hidden layer of 8 neurons also using ReLU

- 1 output layer of 3 neurons with Softmax activation to output the probability distribution of frail, pre-frail or non-frail and choose one

The optimizer uses Adam, an adaptive learning rate optimization algorithm.

The loss function uses Sparse categorical Crossentropy, suitable for integer encoded multiclass targets (0,1,2 for the respective categories).

And the training loop consists of 50 epochs with a batch size of 8.

Results:

After training and validation, we can see the results that the neural network produced:

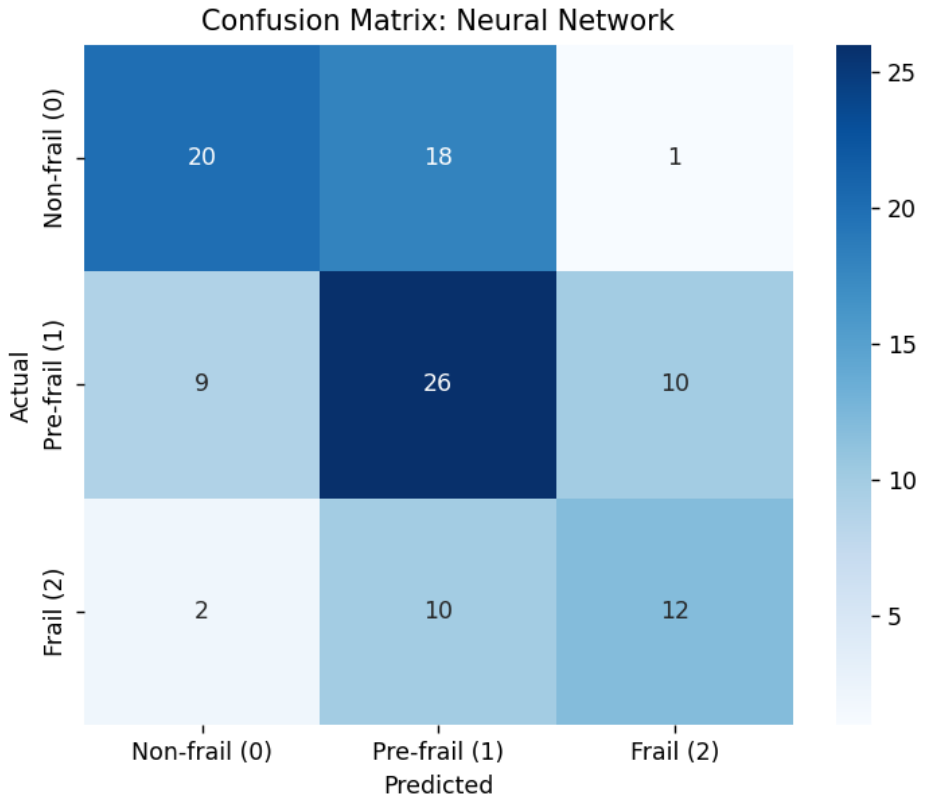

Figure 3 Confusion matrix of neural network

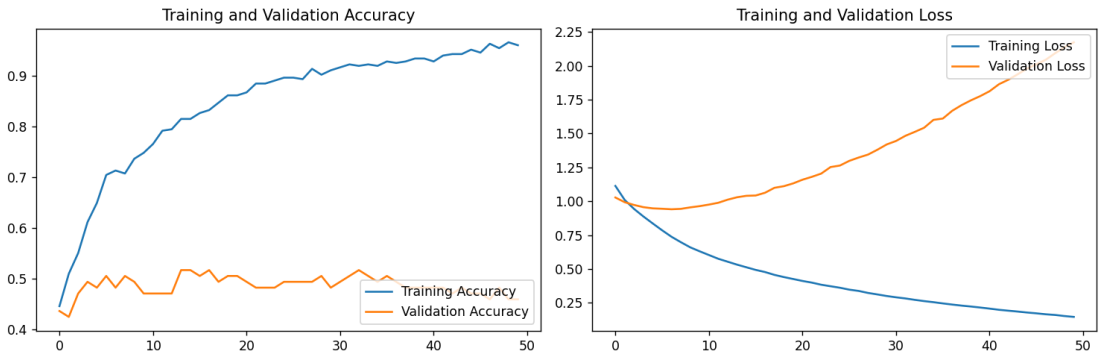

Figure 4 Training and validation accuracy and loss of neural network

The results are not very encouraging. While in the confusion matrix we can see that most people are classified correctly, we also notice (in the validation set):

- Out of the 39 non-frail participants (of the validation set), 18 of them are classified as pre-frail (thankfully only one as frail).
- Out of the 45 pre-frail participants, 9 are classified as non-frail and 10 as frail.
- Out of the 24 frail participants only half of them are correctly classified (at least only 2 of them were classified as non-frail).

In the Training and Validation accuracy graph things look even worse, while training accuracy increases and reaches up to 96%, validation accuracy plateaus at around 50%.

This is a classic case of overfitting where the "learning" from training doesn't translate to general knowledge, but to specific knowledge just for the training set.

This is not an unreasonable result, we have a dataset of around 500 participants, each with 50 fields of clinical data. We can't expect a neural network to extract general knowledge in such a multidimensional space with so few samples.

For that reason, a random forest model was also used.

Random forest model:

The random forest classifier is designed to handle class imbalance and also provides interpretability through feature importance analysis, identifying the most important clinical parameters that drive predictions.

Similarly to the neural network we exclude the variables used to frailty classification, split the dataset 80-20 and scale for consistency with other models.

The classifier is set to use 100 decision trees, while the imbalance in the dataset (some classes having fewer samples than others) is addressed by using balanced class weights.

Results:

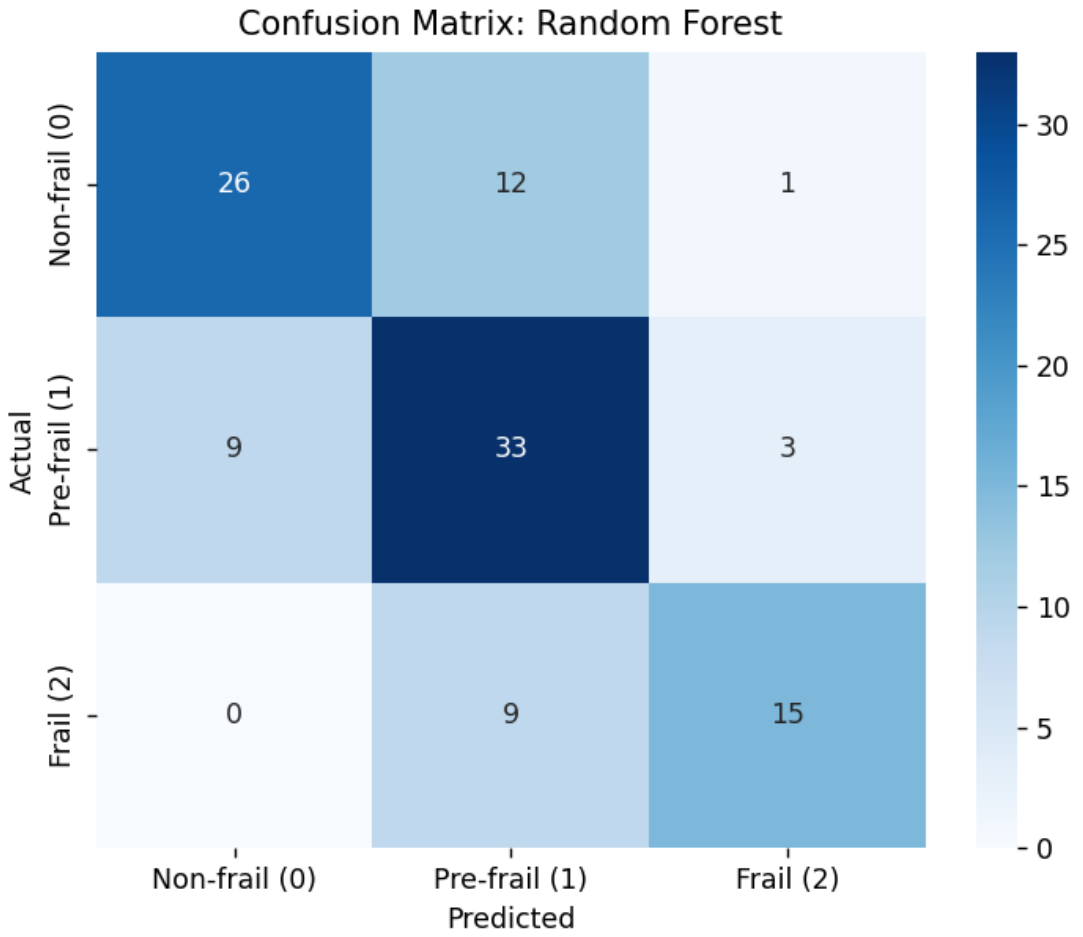

Figure 5 Confusion matrix of random forest

As we can see the confusion matrix is better than that of the neural network, with smaller "leakage" from the correct classification. The accuracy the model achieved is 68.52% which is 27% better than guessing the largest group (pre-frail).

The most important part is how the model never classifies a frail person as non-frail or a non-frail person as frail (with the exception of one participant).

```
--- Random Forest Results ---
              precision    recall  f1-score   support

Non-frail (0)       0.74      0.67      0.70        39
Pre-frail (1)       0.61      0.73      0.67        45
    Frail (2)       0.79      0.62      0.70        24

     accuracy                           0.69       108
    macro avg       0.71      0.67      0.69       108
 weighted avg       0.70      0.69      0.69       108

Accuracy: 0.6852
```

Figure 6 Random forest results

Moving on we can test the interpretability of the classifier by looking at the importance of each attribute to its decision making:

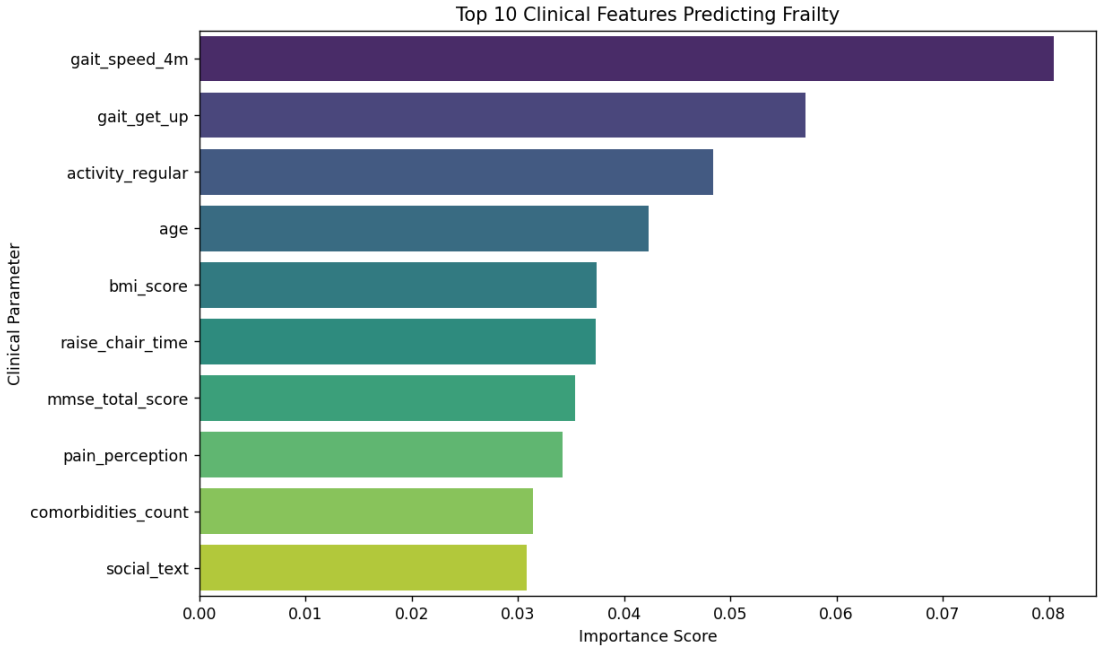

Figure 7 Top 10 attributes for rf classification

We deduce that physical tests (such as gait_speed_4m and gait_get_up) impact the decisions a lot, while regular activity is also very important. Age seems to also play an important role in the final decision while the rest of the variables seem to be in the same range of importance.

3. Part B

Part B revolves around preprocessing the beacons dataset and extracting useful features out of it. Next up is merging it with the clinical dataset and clustering the merged dataset.

3.1 B.1

In the preprocessing of the beacons dataset (file "PartB/beacons_preprocessing.py") we start by loading the raw data and then dropping the participant with invalid ids (such as test, test1, etc). Then we proceed to fix the mistaken room names found in the dataset. As an example, here are the mistaken names that are supposed to mean living room

```
# --- Living Room Variants ---
aliases['Livingroom'] = [
    "Leavingroom", "LivingRoom", "Sittingroom", "Sittigroom",
    "SittingOver", "LeavingRoom", "SeatingRoom", "LuvingRoom",
    "Livingroom1", "Liningroom", "Leavivinroom", "livingroom",
    "Living", "Livingroon", "LivibgRoom", "Luvingroom1", "Sittinroom",
    "SittingRoom", "Sitingroom"
]
```

Figure 8 Living room variants

It's worth noting that in cases where a label such as "Livingroom2" was encountered, it classified as a different room than just "Livingroom" since it is important to differentiate between different rooms as we will see in the feature extraction.

The next step is to fix continuity issues:

The renaming process (and perhaps even the raw data itself) created continuity issues when it comes to the beacons data. For example, a timestamp of a participant entering the kitchen might be followed by another timestamp of the same participant entering the kitchen again, without leaving first. That behavior would cause issues in feature extraction if not addressed.

Finally, in the "PartB/generate_features.py" script we take the "clean" beacons dataset, and we decide which features we will produce to be used for clustering. After consideration, the following features were selected:

- Percentage of time in bedroom
- Percentage of time in bedroom during night-time
- Percentage of time in bathroom
- Percentage of time in bathroom during night-time
- Percentage of time outdoors
- Room changes during the night
- Room changes total
- Average room changes per day
- Average time per room

After those features were clalculated, the newly created dataset was named "PartB/participant_features.csv".

After some examination however, we can see that there are some issues with the data. For example most participants show 0% of their night being spent in the bedroom, or even 0% of their entire day in it.

	A	B	C
1	part_id	pct_time_bedroom_night	pct_time_bedroom
2	124	0	0
3	1001	0	14.22
4	1003	0	0
5	1005	0	0
6	1006	0	6.66
7	1007	0	15.62
8	1022	0	4.35
9	1023	0	0
10	1030	7.03	16.52
11	1035	0	0
12	1036	0	0
13	1038	0	0.41
14	1039	0.08	2.36
15	1040	0	0.13
16	1041	0	38.21
17	1043	34.62	52.26
18	1044	0	0
19	1049	0.46	11
20	1051	10.66	80.07
21	1054	0	0
22	1055	0.71	40.85
23	1066	1.34	57.41
24	1068	0	3.36
25	1084	0	0
26	1085	0	0
27	1086	0	0
28	1088	0	3.87
29	1089	0	16.02
30	1090	0	0
31	1091	0	0
32	1092	0	24.85
33	1094	0	0
34	1095	0	0

Figure 9 Issues with times in bedroom

3.2 B.2

The merging of the datasets happens in the “/PartB/merged_clustering.py” file.

To prevent high orders of dimensionality in the dataset, a subset of the clinical and beacon features were selected for the final clustering. The group of features includes the most important clinical features in the random forest classification (gait_speed_4m, gait_get_up, raise_chair_time, activity_regular, age) and 5 of the most (subjectively) important beacon features that did not suffer from dataset issues (as the bedroom_at_night issue mentioned above), those being: room_changes_total, room_changes_at_night, percentage_outdoor, average_time_per_room, average_room_changes_per_day.

With 10 attributes selected we are barely not over the 10-15 dimensions limit where distance loses its meaning.

3.3 B.3

The clustering algorithm selected was Kmeans, the results of the clustering will be shown and explained in part C

4. Part C

In this final part we will examine the results of the clustering.

Since we used the Kmeans algorithm we ran it for different values of K to use the elbow method and see if for any value we gain significant cost reduction for the number of clusters:

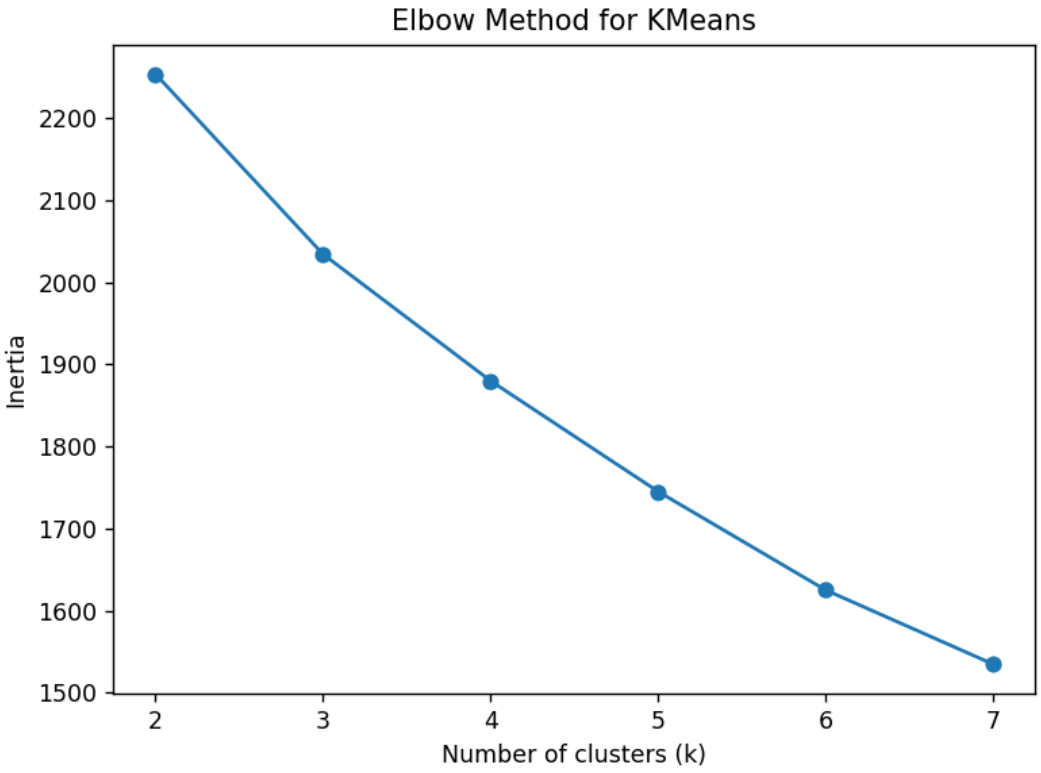

Figure 10 Kmeans elbow method

Similarly for the silhouette index of those clusterings:

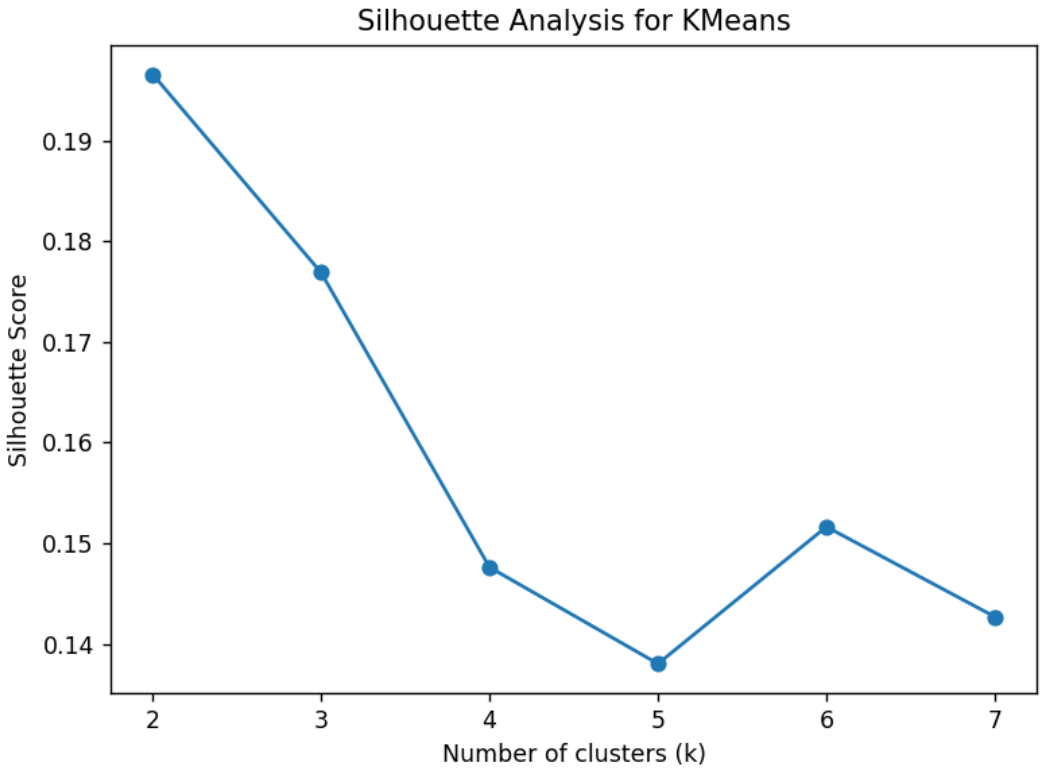

Figure 11 Shiluette index for different numbers of clusters

As we see, silhouette index values are rather low (0.14-0.19) and there is no visible elbow-method-value for k-means.

For that reason, we will proceed with 3 clusters just because of the 3 categories of the fried attribute.

For visualization of he clustering (given that it takes place in a 10-dimensional space) we use the PCA with 2 components and plot the clustering and the frailty side by side:

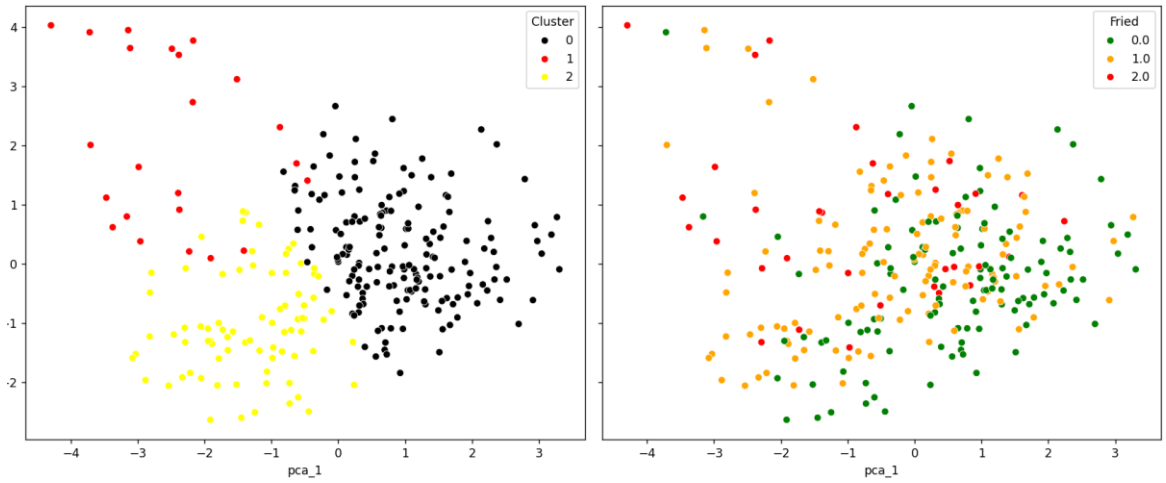

Figure 12 Clustering and frailty PCA2

We can't discern any strong correlation between the two variables in the 2D space.

Therefore we move on to PCA with 3 components to hopefully see if there is any connection between the clusters and frailty:

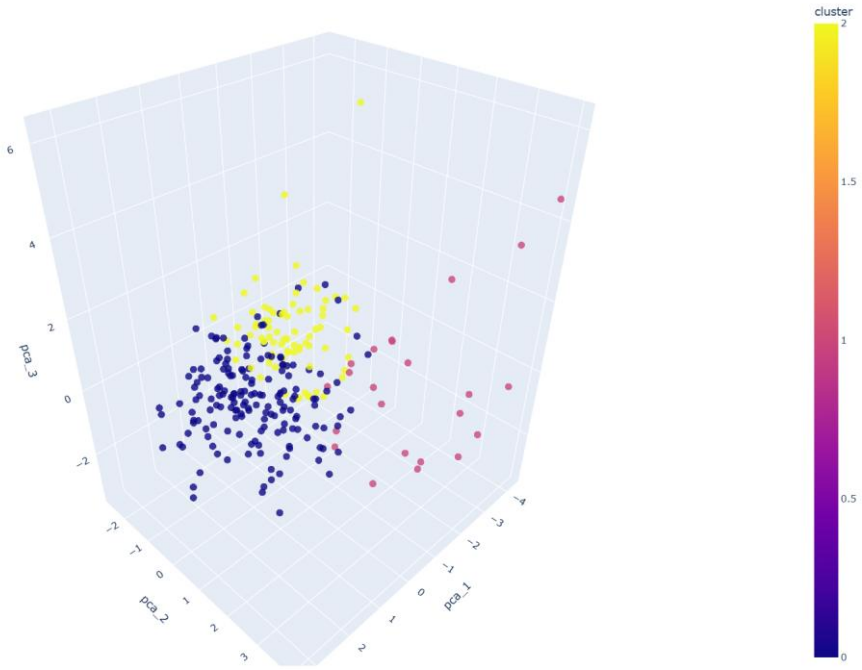

Figure 13 Clustering in PCA 3

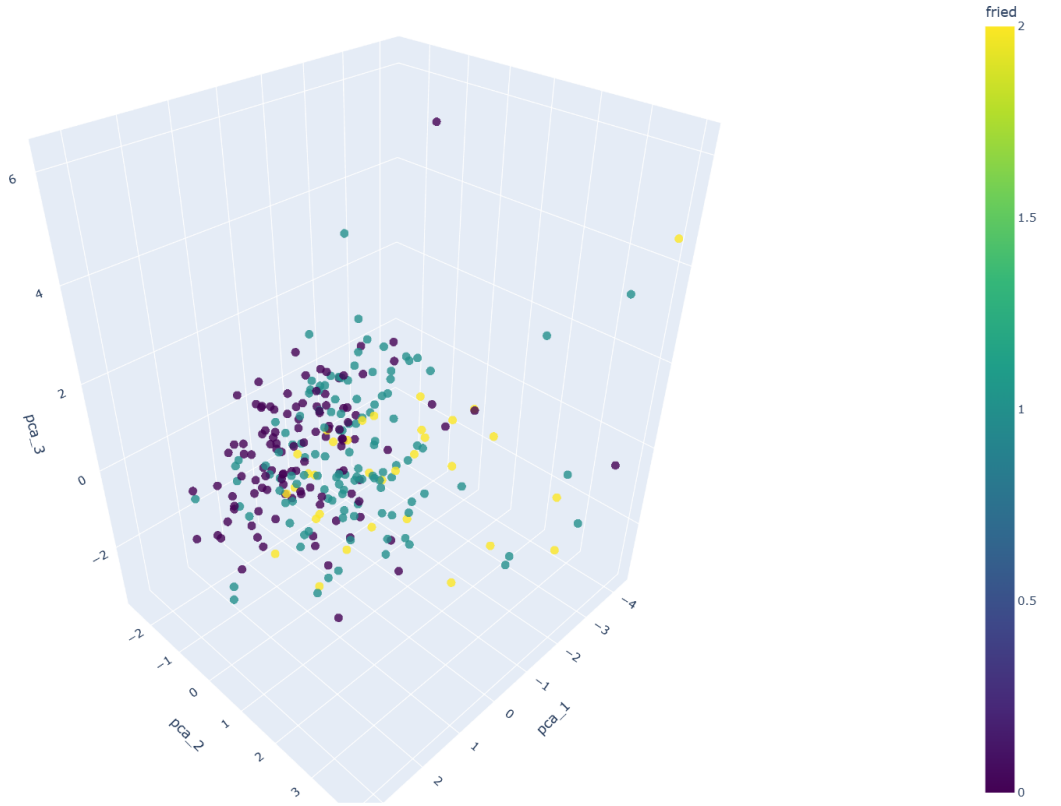

Figure 14 Fried in PCA 3

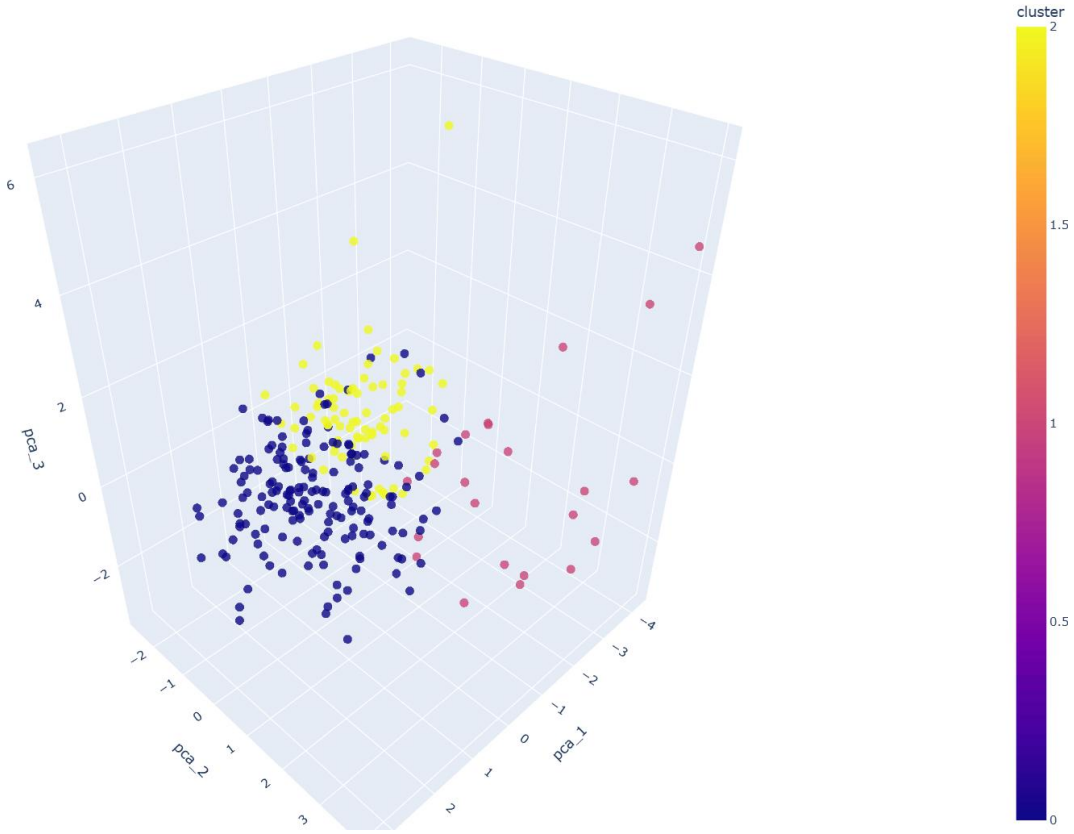

Figure 15 Clustering vs Fried in PCA 3

As we can see (more so in the gif and the 3d viewing that the program produces) there is some correlation between the clusters and frailty.

The same result can be seen in the mean FRIED score of each cluster:

- Cluster 0: 0.569
- Cluster 1: 1.391
- Cluster 2: 0.759

Meanwhile using the ANOVA p-values we can see which characteristics played an important role in the clustering:

```
ANOVA p-values by clustering feature:
gait_speed_4m              : 3.7019e-35
gait_get_up                : 8.8842e-34
raise_chair_time           : 7.5076e-07
activity_regular           : 1.2034e-01
age                        : 1.2455e-04
room_changes_total         : 1.5149e-15
room_changes_night         : 1.4642e-05
pct_time_outdoor           : 1.1139e-04
avg_time_per_room          : 3.7206e-01
avg_rooms_per_day          : 4.8076e-26
```

Figure 16 Anova p-values for clustering

The most important ones being gait_speed_4m, gait_get_up and average rooms per day.

Going more in depth on the identity of each cluster we can look at the mean values for each attribute for each cluster:

cluster	0	1	2
gait_speed_4m	3.94	10.74	5.44
gait_get_up	8.49	17.77	8.60
raise_chair_time	12.62	17.53	12.83
activity_regular	2.10	1.70	2.08
age	75.87	80.35	75.22
room_changes_total	201.73	56.13	23.14
room_changes_night	23.44	4.30	0.56
pct_time_outdoor	19.52	23.89	9.34
avg_time_per_room	2169.17	2701.80	2718.29
avg_rooms_per_day	4.21	3.13	2.73

Figure 17 Attribute means for each cluster

Cluster 0:

- The most “athletic” (often close to cluster 2) in fields like gait_speed_4m
- As young as cluster 2, significantly younger than cluster 1
- A lot more room changes total, per day and at night

- Less time per room than the other two clusters

Makes sense that it has the lowest frailty of the three clusters.

Cluster 1:

- Least athletic (worst gait_speed_4m and other fields)
- Oldest (80 average age)
- Least regular activity
- Medium number of room changes

It is the most frail group as we saw earlier with a mean value of 1.391

Cluster 2:

- Almost as athletic as cluster 0
- Regular activity
- Same age as cluster 0
- Smallest number of room changes out of all groups
- Least amount of time outdoors

It is more frail than cluster 0 by a small margin 0.759 vs 0.569, probably explained by the small athletic differences.

This raw interpretation of the clusters and their frailty values gives us hints that room changes are not necessarily a big frailty indicator.

We have two groups (0 and 2) with complete opposite room changing frequences that have almost similar frailty ratings, while the group with the most frail participants has an average amount of room changes.