Automated Classification of Activity Groups based on empirical Motion Data: Preliminary Results

L. Hans, J. Sebastian, A. Firsova

December 15, 2021

Trier University - Research project - W21-22

Table of contents

Introduction and Research Question

Automated Classification of Activity Groups based on empirical Motion Data

Background

In the ”SiNuS-Pflege” project, data were collected during the performance of physical activities. The aim is to find out whether there are correlations between cognitive, emotional ,and physical parameters for future work.

Research Question

The central question of our elaboration is whether ***clusters can be identified in a particular subset of the collected data***.

Data preparation for device data

Physical Activity Data

The data preparation can be divided into two parts. We distinguish between the section of additional information extraction from the collected data and the data preparation of the already collected data set. The additional data comes from the terminal equipment:

- Fitbit
- Smartphone
- Accelerometer

Here, the data for each participant was available in CSV format; however, before the data of interest could be extracted, it first had to be located.

Problems in extracting the Information

A major problem in extracting the data of interest was that the correct time periods of six minutes and one minute had to be located. One advantage was that all the data was synchronized, so the time intervals only had to be located once.

To do this most cleverly, we started from the smartphone data set, which only contains two columns. The first column contains the time and the second one a numerical value. We could also use the fact, that our data contains one row per second, which made it a bit easier.

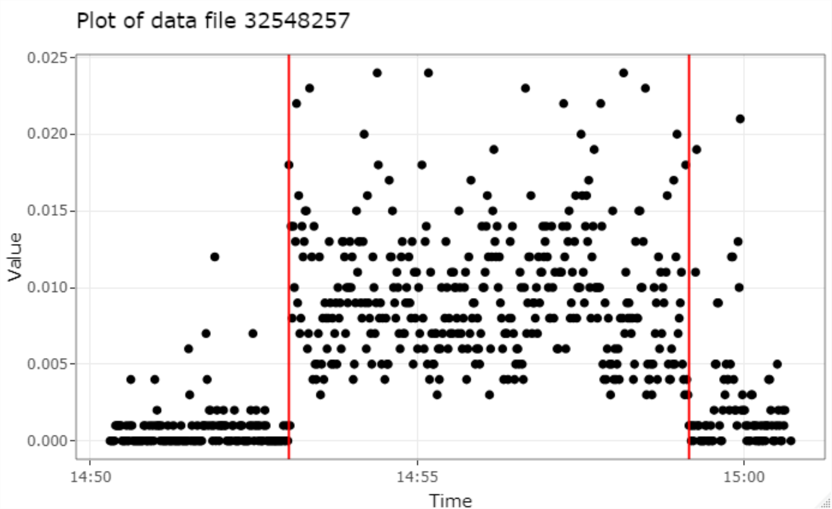
Approach to find 6MWT

Algorithm 1 pseudo code to find the time intervals

for $i = 1$ to N **do**
    $Result[[i]] = sum(smart[i : (359 + i), 2] > mean(smart[, 2]))$
end for
$Final = smart[which.max(Result[[i]]) : (which.max(Result[[i]]) + 359), 2]$

- smart = smartphone data frame
- N = Amount of rows in the data frame
- The algorithm uses R logic, so the index starts at 1

Visualisation of identified time span for 6mwt



Data preparation for the main file

Data Preparation Steps

- Missing values treatment
- Normalization
- Correlations treatment
- Feature scaling algorithms (PCA, Isomap, MDS, UMAP)

Missing Values Treatment

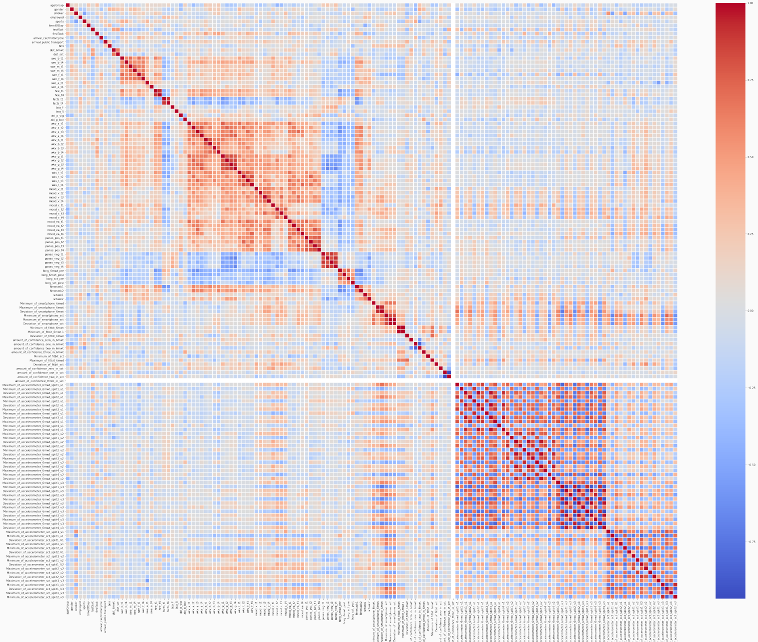
Clustering algorithms in general do not tolerate missing values. Two options:

- Where it was possible, we filled missing values with either mean (numeric values) or mode.
- Conditionally filled variables had to be excluded from our dataset (e.g. values available only for employed participants).

Normalization tools

MinMaxScaler from sklearn library[1].
Reason: MinMaxScaler scales all the data features in the range [0, 1] or else in the range [-1, 1] if there are negative values in the dataset. It causes no distortions and does not require the data distribution to be normal.

Feature Correlation



Feature Correlation, cont.

Major problems:

- Size of dataset: 42 objects with 148 features
- Large groups of correlated features:
  - swe variables: self-efficacy expectation
  - wkv variables: perceived physical condition
  - mood variables: perceived psychological condition
  - derived variables from devices data

Feature Correlation, cont.

Proposed solution:

- Harsh reduction of correlated variables (threshold = 0.5, 126 variables deleted)
- Mild reduction of correlated variables (threshold = 0.9, 26 variables deleted)
- Use of feature scaling algorithms to get at most 2 components (PCA, Isomap, MDS, UMAP)

Feature Scaling

We have tried a variety of algorithms:

- Principal Component Analysis: identifying the linear components of a set of variables
- Isomap: identifying the nonlinear components
- Multidimensional Scaling: nonlinear dimensionality reduction
- Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP): general non-linear dimension reduction[2]

All algorithms were applied based on sklearn and umap-learn libraries in Python 3.7.

Clustering Algorithms

From the variety of options[3], the following algorithms were chosen:

- Methodologically suitable for small datasets:
    - Hierarchical (agglomerative) clustering (any pairwise distance)[4]
    - Gaussian (Bayesian-based) mixture model (Mahalanobis distances to centers)
- Best suitable for low dimensional datasets (after feature scaling):
    - k-Means (general purposed; distances between points)[5]
    - DBSCAN (based on neighbourhood size; distances between nearest points)[4]
    - BIRCH (Euclidean distance between points)

Instruments for choosing optimal number of clusters

For different algorithms, one or several of the following instruments were used:

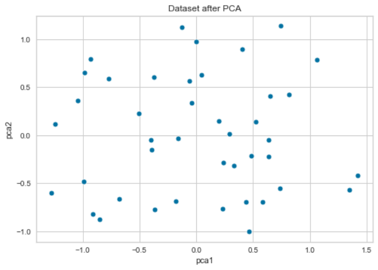
- Elbow method (SSE)
- Silhouette method
- Calinski-Harabasz method
- Gap statistic method

The choice of optimal number of clusters in case of doubt was made in favour of simple majority.

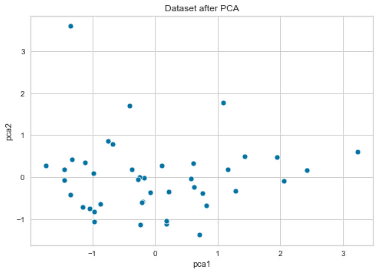
Implementation and Results

PCA

Application of PCA



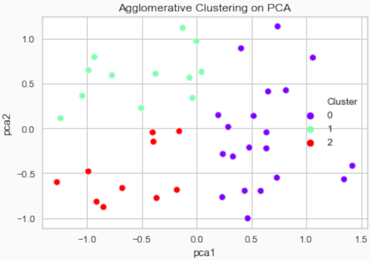
Set after harsh reduction



Set after mild reduction

Figure 1: Dataset after PCA Application

Agglomerative Clustering for PCA: optimal k = 3

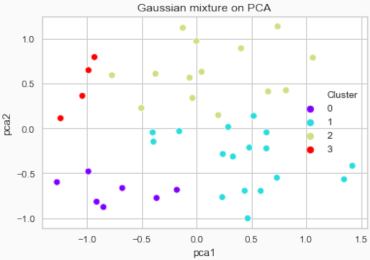


Set after harsh reduction

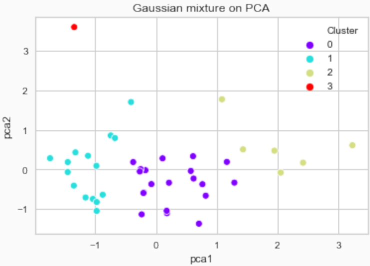
Set after mild reduction

Figure 2: Agglomerative Clustering, k = 3

Gaussian Mixture Model for PCA: optimal k = 4



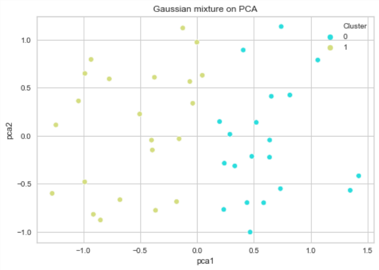
Set after harsh reduction



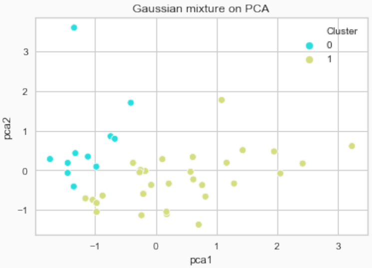
Set after mild reduction

Figure 3: Gaussian Mixture Model, k = 4

Gaussian Mixture Model for PCA: k = 2 (suboptimal)



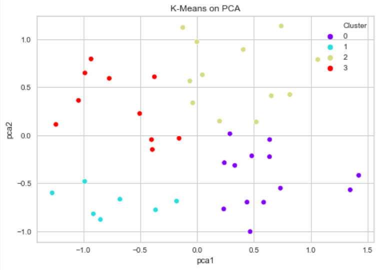
Set after harsh reduction



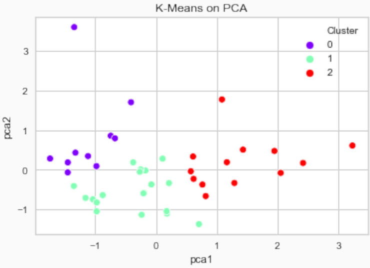
Set after mild reduction

Figure 4: Gaussian Mixture Model, k = 2

K-Means for PCA: optimal k = 4 vs k = 3



Set after harsh reduction

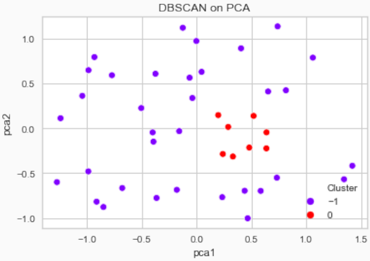


Set after mild reduction

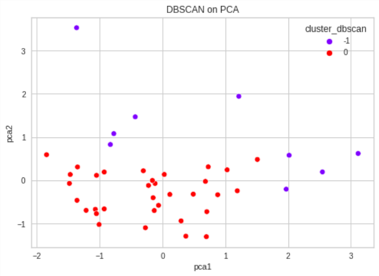
Figure 5: K-Means, k = 4 (harsh) vs k = 3 (mild)

Here, we did not get the same optimal number from our instruments!

DBSCAN for PCA: optimal k = 2



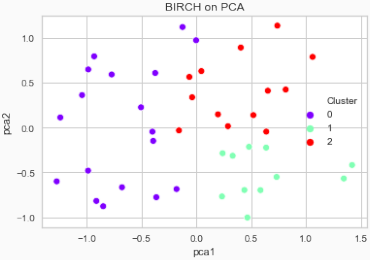
Set after harsh reduction



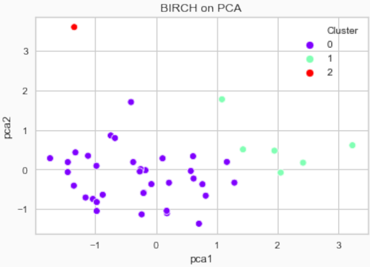
Set after mild reduction

Figure 6: DBSCAN, k = 2

BIRCH for PCA: optimal k = 3



Set after harsh reduction



Set after mild reduction

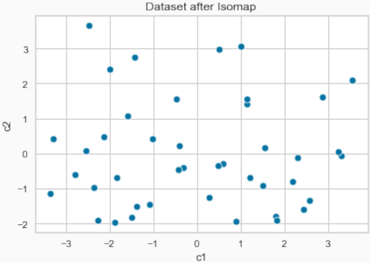
Figure 7: BIRCH, k = 3

Conclusion on PCA

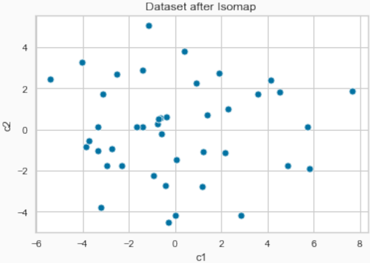
For the mild version, K-Means, Agglomerative clustering and GMM provide similar results. But all in all, robust results are achieved: clusters differ from algorithm to algorithm.

Isomap

Application of Isomap



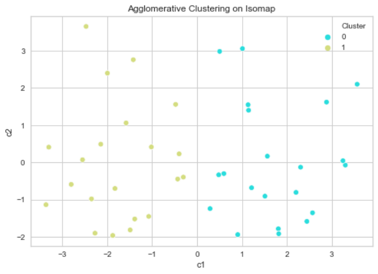
Set after harsh reduction



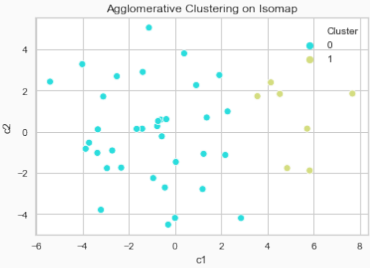
Set after mild reduction

Figure 8: Dataset after Isomap Application

Agglomerative Clustering for Isomap: optimal k = 2



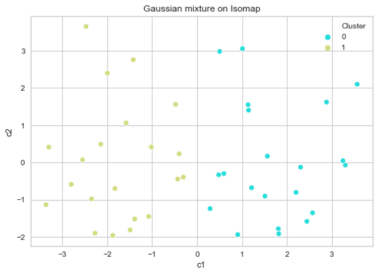
Set after harsh reduction



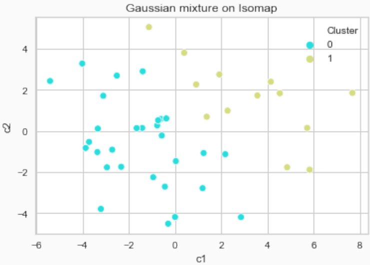
Set after mild reduction

Figure 9: Agglomerative Clustering, k = 2

Gaussian Mixture Model for Isomap: k = 2 (suboprimal)



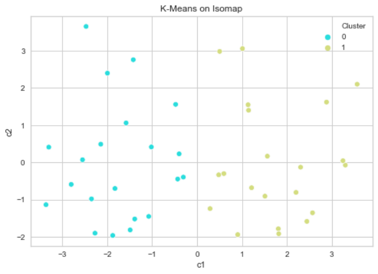
Set after harsh reduction



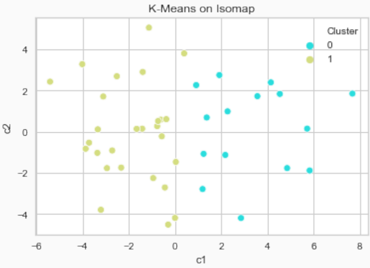
Set after mild reduction

Figure 10: Gaussian Mixture Model, k = 2 (suboptimal)

K-Means for Isomap: optimal k = 2



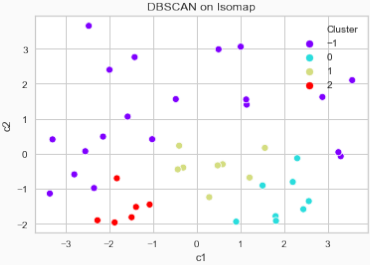
Set after harsh reduction



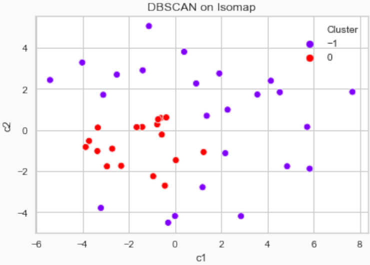
Set after mild reduction

Figure 11: K-Means, k = 2

DBSCAN for Isomap: optimal k = 4 vs k = 2



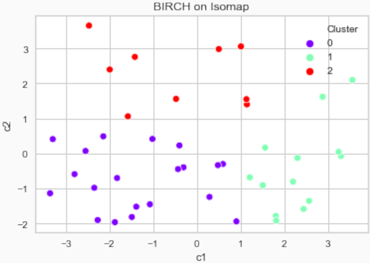
Set after harsh reduction



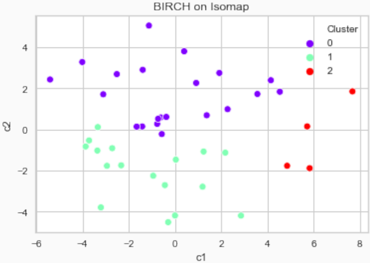
Set after mild reduction

Figure 12: DBSCAN, k = 4(harsh) vs k = 2(mild)

We do not get the same number of optimal clusters here!



Set after harsh reduction



Set after mild reduction

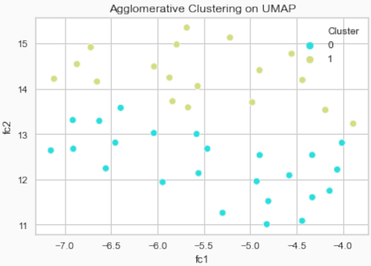
Figure 13: BIRCH, k = 3

Conclusion on Isomap

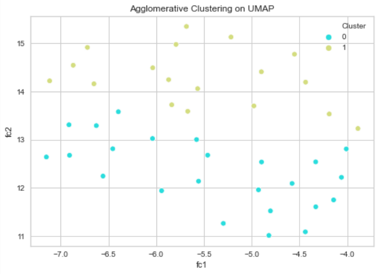
Application of nonlinear algorithm did not significantly improve situation: we still cannot get consistent results. However, for the harsh version, K-Means, Agglomerative clustering and GMM perform similarly.

UMAP

Agglomerative Clustering for UMAP: k = 2 (suboptimal)



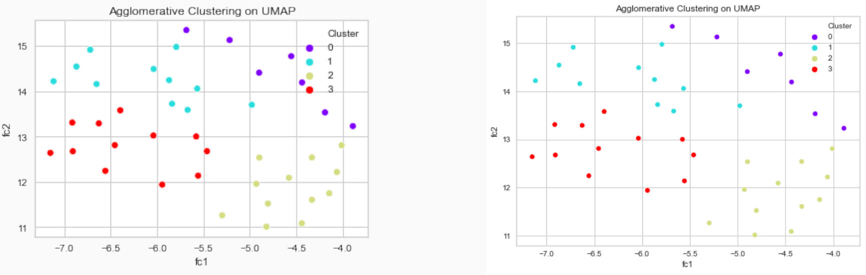
Set after harsh reduction



Set after mild reduction

Figure 14: Agglomerative Clustering, k = 2

Agglomerative Clustering for UMAP: optimal k = 4

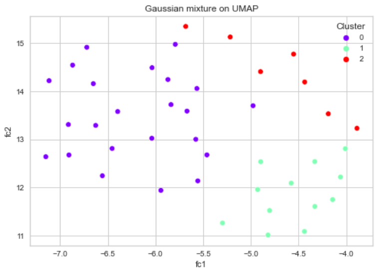


Set after harsh reduction

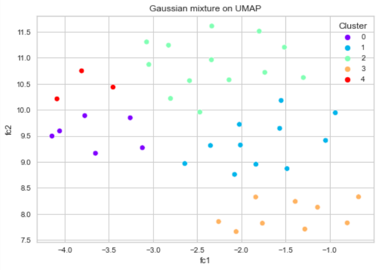
Set after mild reduction

Figure 15: Agglomerative Clustering, k = 4

Gaussian Mixture Model for UMAP: optimal k = 3 vs k = 5



Set after harsh reduction

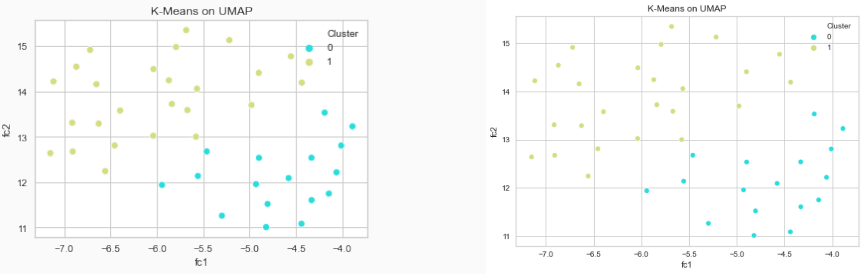


Set after mild reduction

Figure 16: Gaussian Mixture Model, k = 3(harsh), k = 5 (mild)

We do not get the same optimal number of clusters here!

K-Means for UMAP: optimal k = 2

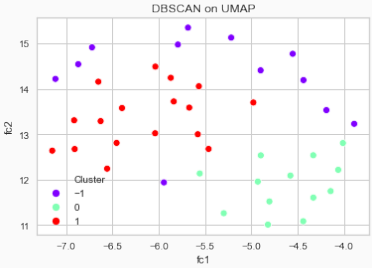


Set after harsh reduction

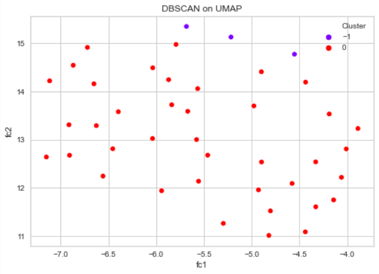
Set after mild reduction

Figure 17: K-Means, k = 2

DBSCAN for UMAP: optimal k = 3 vs k = 2



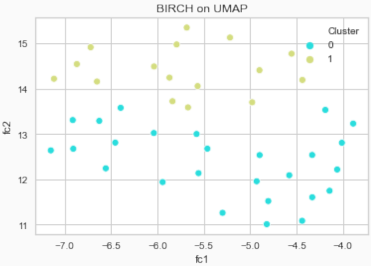
Set after harsh reduction



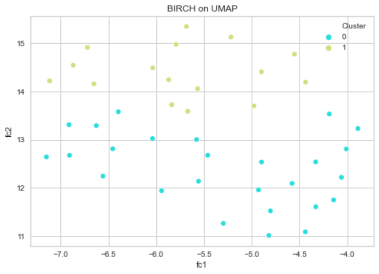
Set after mild reduction

Figure 18: DBSCAN, k = 2

BIRCH for UMAP: optimal k = 2



Set after harsh reduction



Set after mild reduction

Figure 19: BIRCH, k = 2

Conclusion on UMAP

No similarities in optimal cases for both versions of dataset.

Conclusion

Conclusion

No ”happy end” to the current state of our research. Here are possible reasons:

- Unconventional dataset demanding sophisticated solutions: it violates all known rules of thumbs for feature scaling and clustering algorithms.
- No automatic feature reduction provides stable results: pure algorithmic solution may not take into account complexity of obviously nonlinear relationships inside our data.
- The group of participants may not be heterogeneous and large enough to provide trustful findings.

What goes next?

We have some more ideas to check before reaching a verdict:

- Further investigation of feature selection algorithms beyond simple correlation analysis.
- Application of further feature extraction algorithms (e.g., LLE - Locally Linear Embedding).

References

[1] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[2] Healy J. McInnes, L. Umap: Uniform manifold approximation and projection for dimension reduction. *ArXiv e-prints 1802.0342*, 2018.

[3] Guido S. Müller A.C. *Introduction to Machine Learning with Python*. O'Reilly Media, Inc, 2016.

[4] Crouse J.J. Abdalla A. Moustafa A.A. Alashwal H., El Halaby M. The application of unsupervised clustering methods to alzheimer's disease. *Front. Comput. Neurosci.*, 2019.

[5] Forsberg F. Alvarez Gonzalez P. Unsupervised machine learning: An investigation of clustering algorithms on a small dataset. *Thesis no: URI: urn:nbn:se:bth-16300*, 2018.

THANK YOU FOR ATTENTION