Clustering of Human Activities Using Data from Smartphone Sensors

Motunrayo Ibiyo

1 Introduction

This project involves clustering of human activity recognition using smartphone using processed dataset downloaded from the University of California Irvine (UCI) machine learning repository. The dataset was collected from 30 volunteers while they carried out six predefined activities. 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz were collected using the accelerometer and gyroscope on Samsung Galaxy S II that the participants wore on their waist [1]. The signals were pre-processed by applying noise filters and then sampled in fixed width sliding windows of 2.56 sec and 50% overlap. The sensor acceleration signal, which has gravitational and body motion components, was separated using a Butterworth low-pass filter into body acceleration and gravity [2].

Clustering analysis is one of the main techniques used in datamining for searching out hidden patterns in a dataset [3]. It is a popular method that is used in several domain for the analysis of a dataset and is categorized as an unsupervised learning method in machine learning. In general clustering analysis functions by grouping the dataset into groups of similar objects. Similarity can be measured based on some function like distance between two points or reachability.
In this project, clustering was carried out using one machine learning algorithm each from distance based and density-based approaches of clustering. The KMeans and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) were selected and trained on the human activity recognition dataset. The clusters generated by the algorithms were compared based on the relationship of the clusters with features in dataset. In addition, the project compares the clusters generated by using dataset from the results of Principal Component Analysis (PCA) to train the model and those generated by using the features from the dataset.

The project will pandas and seaborn libraries for data visualization, and scikit-learn for data preprocessing and model training and evaluation. The next sections contain the steps carried out in the project. Section two provides details of the Data analysis, preprocessing and Feature selection. The third section discusses the methodology, and this part provides details of the experimentation with PCA and without PCA. The fourth section provides the result of the models and compares the result of the two algorithms in both scenarios. The final section gives overview of the observations in this project and highlights challenges experienced.

2 Data Analysis, Preprocessing and Feature Selection

The section analyses the data and the discusses the feature selection process that was used for the model training on the raw dataset.
For this project, four main files from the dataset were used. The files used were the X_train.txt, y_train.txt, X_test.txt, and y_test.txt and they made up the of training and testing set respectively. The y_train and y_test files were used for visualization and analysis while the X_train, and X_test was used to train the clustering algorithm.

2.1 Exploratory Data Analysis

The train and test dataset consists of 7,352 and 2,947 datapoints respectively. The datasets consisted of 561 numerical features. There were no null values in the dataset and all features were normalized to range between -1 to 1.

The y_train and y_test consists of numerical encoded labels (1-6) for the activities being performed for corresponding features set. Table 2.1 shows the encodings and their interpretation. Figure 2.1 show the distribution of the activities.

Table 2.1: Label encoding for activities in the dataset.

1	WALKING
2	WALKING_UPSTAIRS
3	WALKING_DOWNSTAIRS
4	SITTING
5	STANDING
6	LAYING

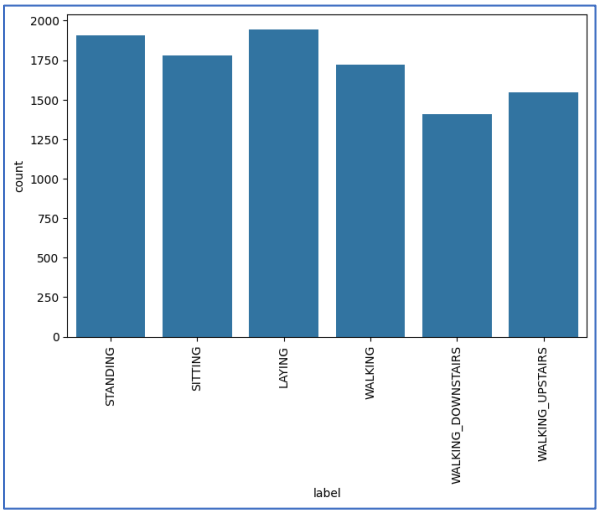

Figure 2.1: The distributions of the activities in the dataset

The train and test set were combined in this project and the clustering algorithms were trained on both datasets. The dataset explored consisted of 10, 299 datapoints and 561 features.

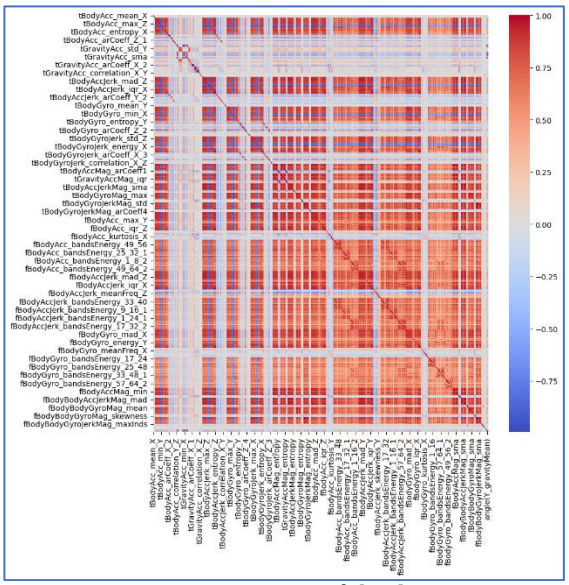

Figure 2.2: Heatmap of the dataset.

The features in the data set contains highly correlated features. Figure 2.2 shows the heatmap with a high percentage of the features having a correlation close to 1 with each other.

Figures 2.3 and 2.4 shows the boxplots of some of the features against the activities label reveals shows interesting relationship. Several features like the tbodyAccjerk_mean_z,,and t_bodyGro_energy_x show that the values representing activities that involved walking were different from where the subject was stationary.

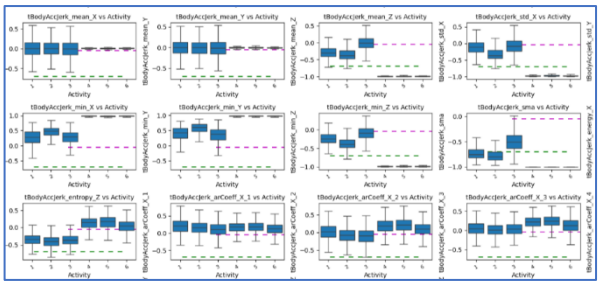

Figure 2.3:Boxplots showing the distribution of features values for each activity.

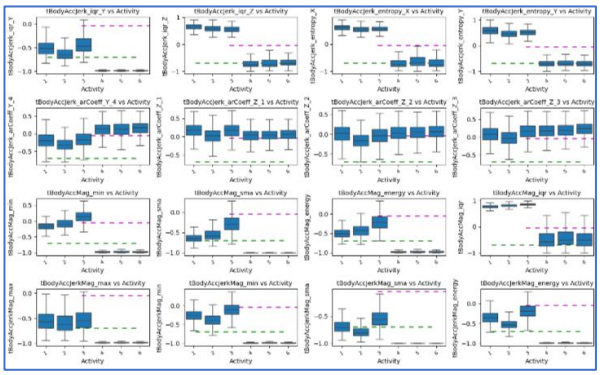

Figure 2.4:Boxplots showing the distribution of features values for each activity.

Angle(tBodyGyrojerkMean_gravityMean) and angle(X_gravityMean) distinctively differentiates laying activity from other activities as shown in figure 2.5.

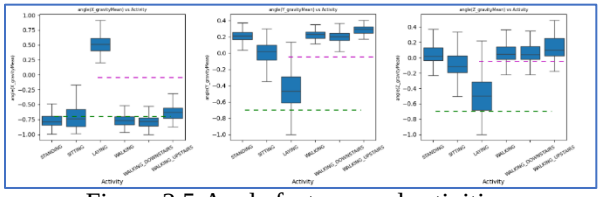

Figure 2.5:Angle features and activities.

2.2 Feature Selection

To reduce the dimension in the dataset, the dimensions was reduced by removing features that had correlation lower than 0.4 with the activity label. The features were further reduced by removing columns with a variation below 0.1. Figure 2.6 shows the correlation distribution of the features.

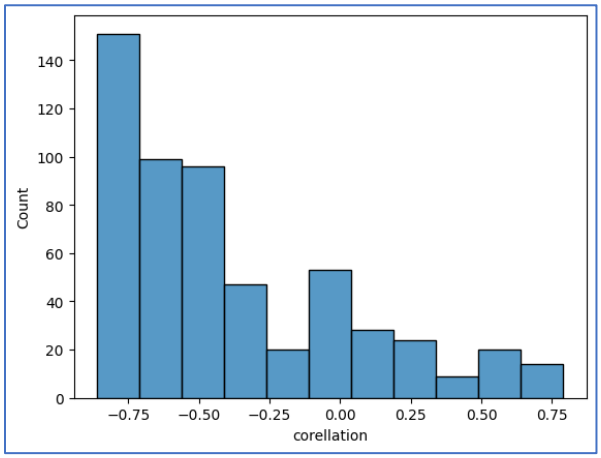

Figure 2.6:Corellation distribution of features and activity labels

From Figure 2.7, the variance is skewed to the left and the threshold of 0.1 removes more than 70% of the features.

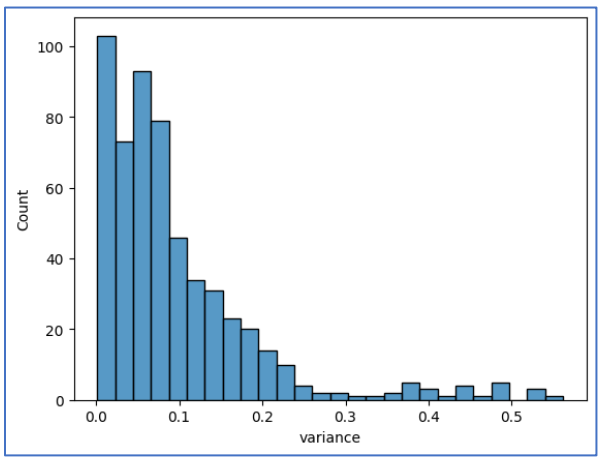

Figure 2.7:Variance distribution of features

After the application of the two feature selection methods, 49 features were left after the in the final dataset that will be used for training the algorithm. Figure show the heat map of the 49 features selected.

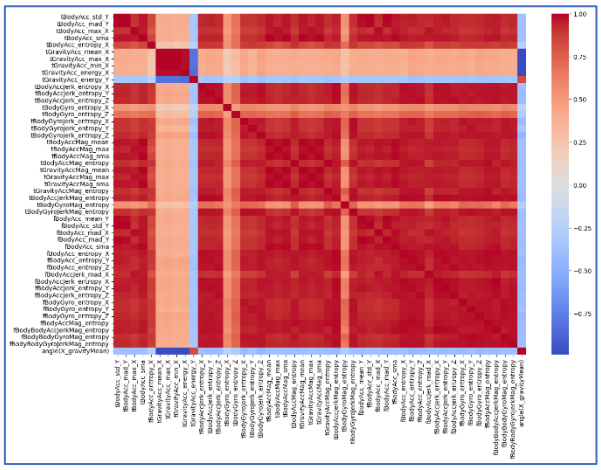

Figure 2.8: Heatmap of selected features

3 Models

This section describes the model training and the process of hyperparameter tuning. For this project, two clustering algorithms with different approaches were trained, KMeans and DBSCAN clustering algorithms.KMeans is a stochastic numerical iterative method that is simple and fast. The operation of the algorithm consists of two phases. The first phase is the selection of K centers usually randomly, where K is the number of predefined clusters. The second phase calculates the Euclidean distance between each datapoint and the cluster center from the first phase. The cluster center is recalculated when all datapoints have been assigned to a cluster. The iterative process of assigning the datapoint to a cluster continues until the criterion function is minimized [3].

DBSCAN is a simple minimum density model that finds areas of high density by grouping together all points within a specified radius epsilon (ε) that are greater than the minimum samples (*MinPts*) specified, these regions are called core points. DBSCAN searches for high density areas that are separated by lower density areas, then incorporates the intuition of reachability to form clusters.
All points within the ε radius of each core points are classified as part of the same cluster this is termed as direct density reachable points. Neighbouring core points that intersect with each other are also classified as single cluster; these points are density reachable points. Data points that are not reachable are classified as noise and do not belong to any cluster [4].
Although DBSCAN is an efficient algorithm, it is sensitive on the values for ε and *MinPts*. The algorithm does also can fail to identify a cluster if the density of the clusters varies or if the dataset is too sparse [5].

To determine the best hyperparameter for both algorithms, the elbow method combined with the analysis of the silhouette score was applied. The elbow method involves the comparison of the cost function for preselected range of clusters. At some hyperparameter value, the cost decreases exponentially after which an addition to the value of the parameter will result in a minute reduction in the cost [6]. The Silhouette score ranges from -1 to 1 and it measures the cohesion of data point within a cluster and the shortest distance between clusters[7]. The silhouette score was chosen as the metric for evaluating the model because it is the only metric that does not use the knowledge of the y label to calculate its score. Each method shows the performance of the models for several parameter values.

3.1 Models Without Component Analysis

The first part of the experiment involves using both algorithms to cluster the raw dataset.

3.1.1 KMeans

One of the limitations of the KMeans algorithm is the need to pre-select the number of clusters(k). To select K the elbow method was applied and the silhouette socre was calculated for each value of k. The results of the elbow method were compared to the silhouette score for each cluster.

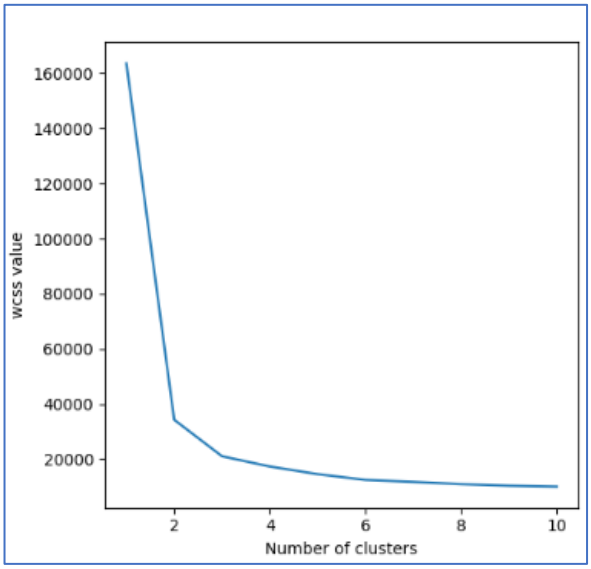

Figure 3.2:Cost of KMeans algorithm clusters.

Figure 3.1:Silhouette score for KMeans clusters

Figure 3.1 and 3.2 shows the result of the elbow method and silhouette score respectively. From both methods, the best cluster to be used in the model is two (2).

3.1.2 DBSCAN

To overcome the behaviour of DBSCAN to highly dense dataset, the dimension of the dataset was reducing using feature selection techniques explained in section 2.2. Then the elbow method and silhouette scores were used to select the best value for ε and min_samples.

The selection of both hyperparameters were done in two steps. The first step involves finding the best ε values from integers between 1 and 20, 1 inclusive. This was done by training the model for different values of ε and setting the min_sample at a fixed value of 5. The second step involved choosing the best value for min_samples parameter. This was done in similar way as the epsilon.
Figures 3.3 and 3.4 shows the result of the elbow method and silhouette scores for selecting ε value. From both figures the best values for ε was 2.
From figure 3.4 it is observed that the silhouette score is only calculated for epsilon values 1 and 2. This is because silhouette score cannot be obtained when the number of generated clusters is 1.

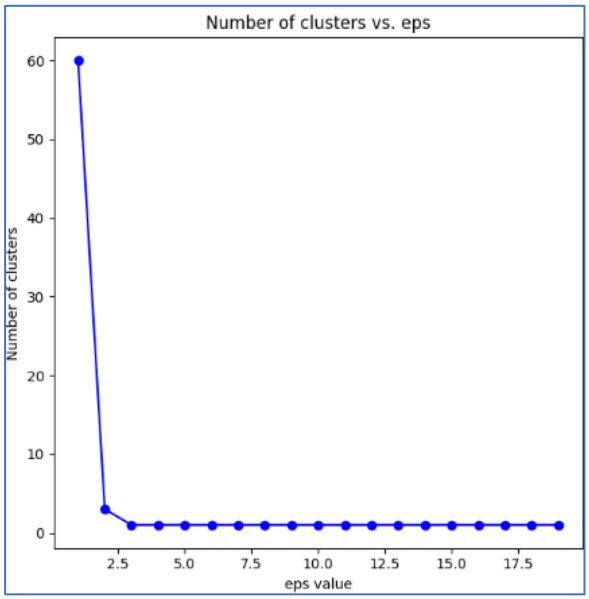

Figure 3.3:Number of clusters generated for epsilon values.

To ensure that the best value of epsilon is chosen, in the second step the silhouette score of the model was computed by iterating over epsilon values 2.0, 2.5 and 3.0 for min_samples ranging from 2 to 10. The result of the second iteration revealed that the best parameters were 2 and 8 for epsilon and min_samples respectively.
Parameters selected:
Epsilion: 2.
Min_samples: 8.

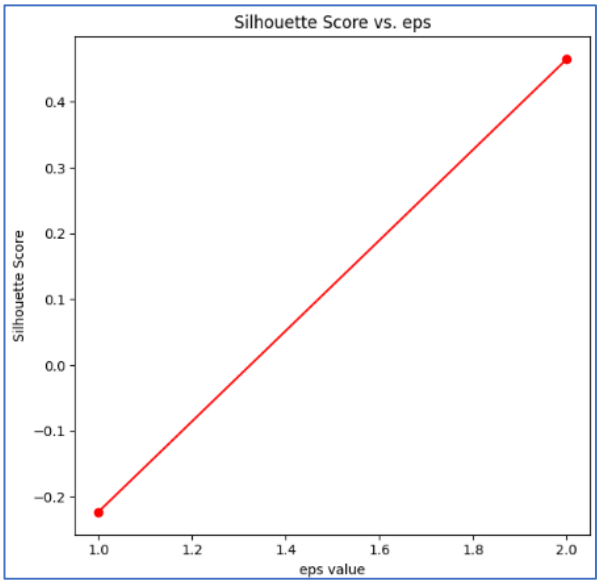

Figure 3.4:Silhouette score for epsilon Values.

3.2 Models With Component Analysis

This section describes the training of the models after component analysis was applied to the full dataset.
The Principal Component Analysis (PCA) technique was used to reduce the dimensions of the dataset. PCA identifies the meaningful basis to represent a given dataset with the expectation that this new representation filters out noise and reveals hidden information in the dataset. PCA is useful for dimensionality reduction and data visualization [8].

The PCA technique was selected for component analysis because the EDA shows the dataset had a form of linearity and PCA is known to perform well on dataset with linear relationship. To use the PCA method in scikit learn, an important parameter to select is the number of components we want the method to return. To select the best values for n_compunents, the explained variance was calculated, and the cumulative sum of the variance was used to determine the elbow point. The elbow point is the point where the change in variance starts to reduce, hence a flattening of the cumulative variance.

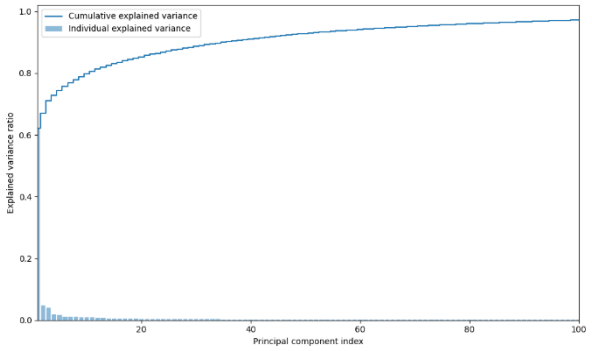

Figure 3.5: Scree plot of PCA

Figure 3.5 shows the scree plot of the cumulative variance for n_components from 1 to 100. Using the second derivative to compute the point where the change in variance was greatest, it was determined that the optimal number of PCA was 3.
N_components selected: 3.

3.2.1 KMeans with PCA

The new dataset from the PCA (n_components: 3) was used to train the KMeans algorithms again. The same technique as in section 3.1.1 was used to select the best number of clusters. Figure 3.6 shows the cost function of the model at each cluster and the silhouette score for each value of k.

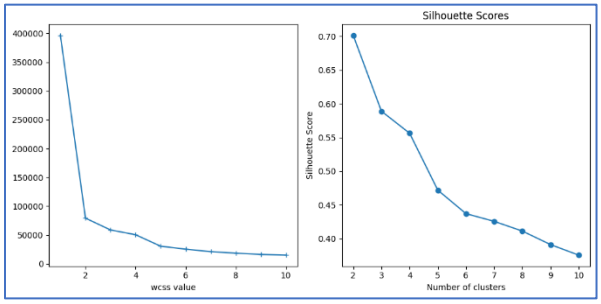

Figure 3.6:cost function of KMeans model for each value of k (Left), silhouette scores for each value of k (right).

The best value of k was 2 and this was used to train the model.

3.2.2 DBSCAN with PCA

The PCA dataset was also used to train the DBSCAN algorithm and the values for epsilon and min_samples were selected using the same two step technique described in section 3.1.2.

Figure 3.7 shows the result of the hyperparameter tuning carried out in the first step. From the image on the left, it is observed that only one cluster was generated by the algorithm from epsilon values greater than 1.5. Hence there are no silhouette scores available for those values on the right figure.

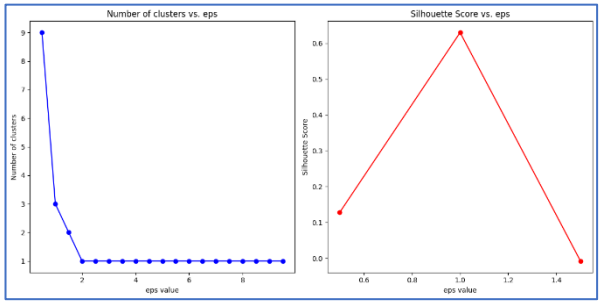

Figure 3.7: Number of clusters generated for epsilon values (left). Silhouette score for each epsilon value.

After the second step was completed, the best values for both parameters were 1.0 and 7 with a silhouette score of 0.6672.
Selected parameters:
Epsilon: 1
Min_sample: 7
These parameters values were then used to train the DBScan algorithm.

4 Model Performance Evaluation

This section describes the result of the four models trained and shows the result of clusters generated by the models. To evaluate the performance of the algorithms, the following metrics were used.

- **Training time**: Time spent to train the model.
- **Model tuning time**: Time spent to determine the best hyperparameters.
- **Silhoutte Score**: Measures cluster similarity and cluster [9]descrimimation. It uses the dataset and the clusters obtained from the algorithm to calculate a value between -1 and 1.

To further understand the datapoints in the clusters of each model, exploratory data analysis (EDA) was carried out on selected features and the clusters. The features used for the EDA were selected by finding similar features in the 49 features from feature selection and the union of the top 5 features that make up the each of the three principal components obtained from PCA. In total 11 similar features were obtained and used to analyse the clusters.

4.1 Evaluation of Models Without PCA

To visualize the clusters that was built on the 49 features selected in section 2.2, the PCA of the dataset with the features was computed and visualized with a scatter plot.
Figure 4.1 and 4.2 shows the clusters from both algorithms.

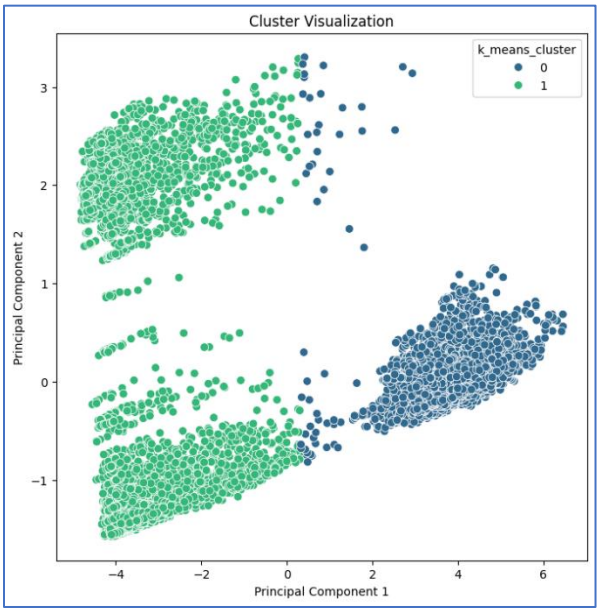

Figure 4.1: Cluster genereated by KMeans algorithm.

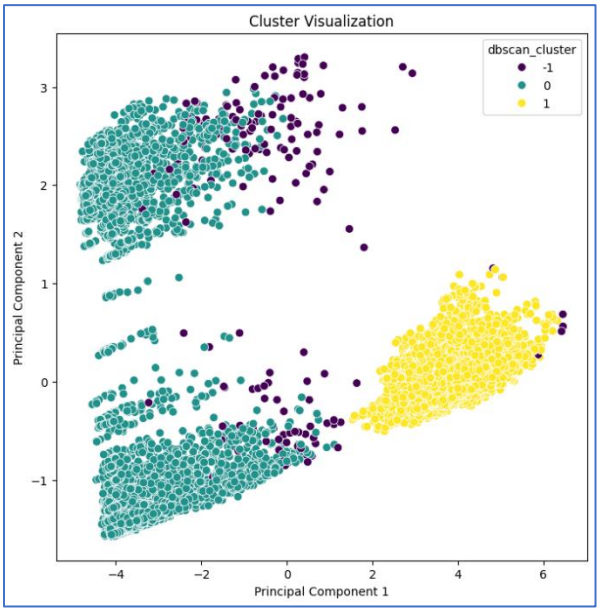

Figure 4.2: Clusters generated by DBScan Algorithm.

Analysing the result from both clusters show that both algorithms identified 2 clusters from the dataset. DBscan also identified a few outliers in the dataset.

4.2 Evaluation of Models With PCA

Figure 4.3 and 4.4 shows the clusters from Kmeans and DBSCAN algorithms.

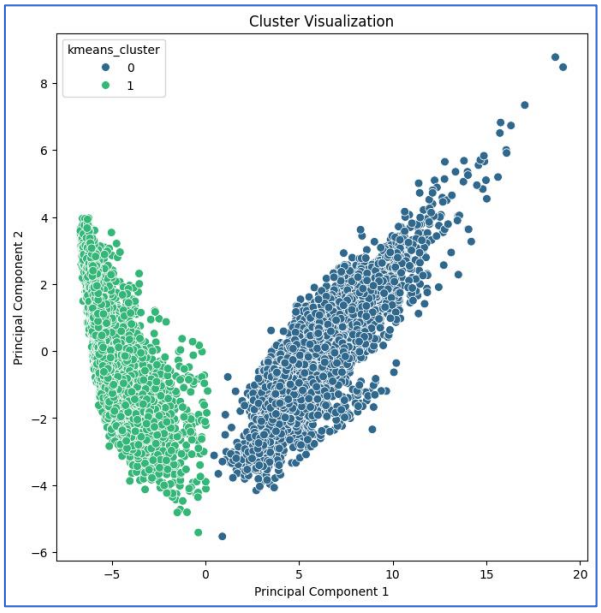

Figure 4.3:luster genereated by KMeans algorithm from PCA dataset.

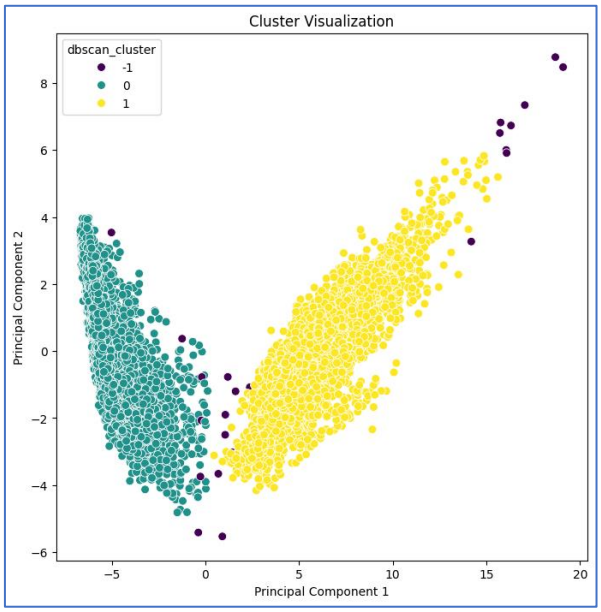

Figure 4.4:Clusters generated by DBScan Algorithm using PCA dataset.

Like the clusters generated from the dataset of selected features, both algorithms generated 2 clusters from the PCA. DBSCAN identified fewer outliers in the in the PCA dataset than in the dataset with feature selection.

4.3 Comparative Analysis of all Model Performance

This section analyses the performance of the models based on the duration for training and hyperparameter tuning, and the silhouette score. Table 4.1 show the score of each model for each metrics. The values showed that the KMeans models had the best performance for all the three metrics.

Table 4.1:Performance Metrics for the models.

Metrics	Without Component Analysis		With Component Analysis	
Algorithm	KMeans	DBSCAN	KMeans	DBSCAN
Model tuning duration (sec)	17.9964	131.3195	18.0746	63.3218
Training duration (sec)	0.1431	0.8541	0.0398	0.3730
Silhouette score	0.6834	0.1436	0.6826	0.2309

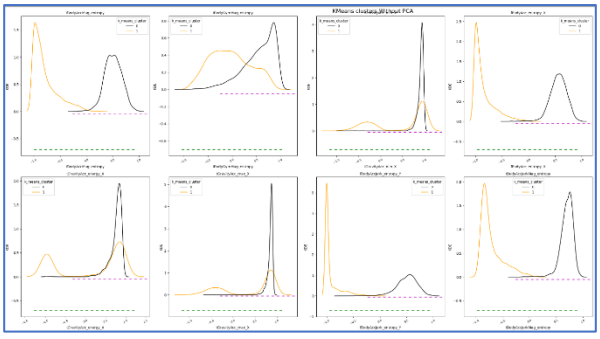

Figure 4.5:Distribution of selected features in KMeans clusters.

Evaluation of the models on 11 selected features shows that the range of values for the features for both clusters is similar in all 11 features for both KMeans and DBSCAN, with subtle difference in the cluster 0 as it contains most of the datapoints classified as noise by DBSCAN.

Figures 4.5 and 4.6 shows the density estimation plot for clusters generated by KMeans and DBSCAN respectively.

Comparing Figure 4.5 and 4.6 show that both models built with PCA and those without PCA had the same distribution for the 11 features investigated.

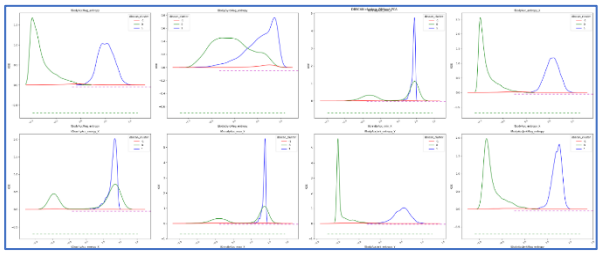

Figure 4.6:Distribution of selected features in DBscan clusters

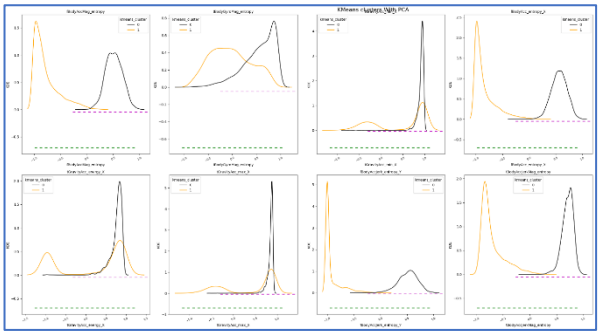

Figure 4.7:Distribution of selected features in KMeans clusters trained with PCA dataset.

Comparing the plots in figure 4.5 and 4.6 with those in figures 4.7 and 4.8 show that the clustering don without PCA and the done with PCA were similar for both techniques.

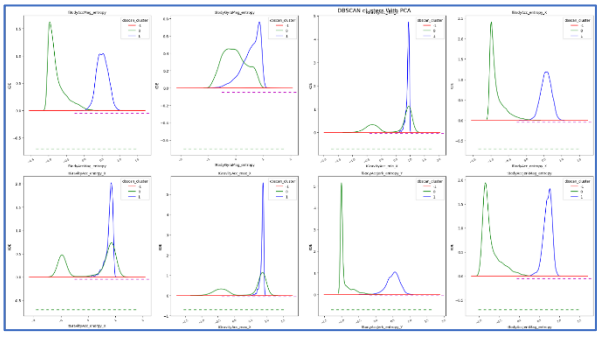

Figure 4.8:Distribution of selected features in DBSCAN clusters trained with PCA dataset.

Further analysis of the clusters against the activities label revealed that the clustering algorithms classified the dataset into stationery and Walking. Stationary represents standing, siting, and laying activities. Fi

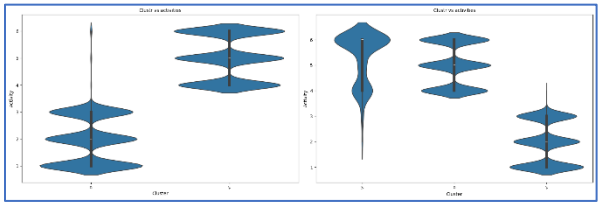

Figure 4.9: Violin plot of KMeans (left) and DBSCAN (right) clusters and activities.

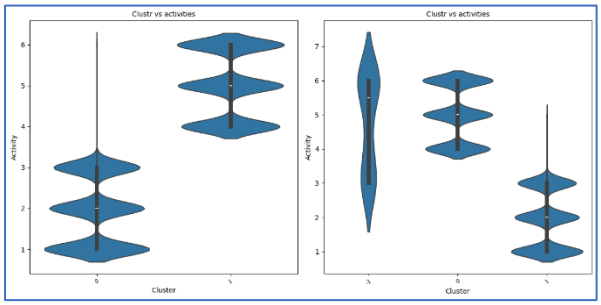

Figure 4.1: Violin plot of KMeans (left) and DBSCAN (right) clusters and activities for PCA dataset.

5 Conclusion

This project successfully clustered the human activity recognition dataset using KMeans and DBSCAN algorithms. The performance of the models were using the raw features were then compared with clustering using PCA of the datasets. It was observed that the models trained faster with the PCA dataset than with the raw dataset and similar clusters was obtained for both algorithms using the raw features and PCA dataset. This show that no information was lost and PCA is a good tool for dimensionality reduction for datasets with a high dimension. In addition, the model trained on PCA identified fewer outliers than the model trained on the raw data.

Overall, the KMeans algorithm performed better than the DBSCAN algorithm, this can be attributed to its inability to generalized well when the density of the cluster is different, and the dataset is sparse.

The two clusters identified by both algorithms in the two scenarios can be termed as stationary activities (laying, sitting, and standing) and no-stationary activities (Walking).

These two clusters shows that the dataset can be used to differentiate when the subject is walking and when they are stationary from the measurement obtained from the accelerometer and gyroscope of a smartphone.

A major issue encountered during the implementation was the inability of DBSCAN to find clusters within the full dataset. To overcome this dimensionality reduction using feature selection was used for the first experiments without PCA.

Another challenge was the selection of hyperparameters for both KMeans and DBSAN. This were resolved by carrying out hyper parameter tuning and analysis using the elbow method.

6 References

[1] Reyes-Ortiz, D. Anguita, A. JorgeGhio, L. Oneto, and X. Parra, ‘Human Activity Recognition Using Smartphones’, *UCI Machine Learning Repository*. 2012. doi: https://doi.org/10.24432/C54S4K.

[2] ‘Human Activity Recognition Using Smartphones - UCI Machine Learning Repository’. Accessed: Mar. 09, 2024. [Online]. Available: https://archive.ics.uci.edu/dataset/240/human+activity+recognition+using+smartphones

[3] S. Na, L. Xumin, and G. Yong, ‘Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm’, in *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*, IEEE, Apr. 2010, pp. 63–67. doi: 10.1109/IITSI.2010.74.

[4] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, ‘DBSCAN Revisited, Revisited’, *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, Jul. 2017, doi: 10.1145/3068335.

[5] S. Weng, J. Gou, Z. Fan, V. N. Balasubramanian, and I. Tsang, ‘h-DBSCAN: A simple fast DBSCAN algorithm for big data’, *Proceedings of Machine Learning Research*, vol. 157. PMLR, pp. 81–96, Nov. 28, 2021. Accessed: Mar. 10, 2024. [Online]. Available: https://proceedings.mlr.press/v157/weng21a.html

[6] P. Bholowalia and A. Kumar, ‘EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN’, 2014.

[7] K. R. Shahapure and C. Nicholas, ‘Cluster quality analysis using silhouette score’, *Proceedings - 2020 IEEE 7th International Conference on Data Science and Advanced Analytics, DSAA 2020*, pp. 747–748, Oct. 2020, doi: 10.1109/DSAA49011.2020.00096.

[8] T. Kurita, ‘Principal Component Analysis (PCA)’, in *Computer Vision*, Cham: Springer International Publishing, 2020, pp. 1–4. doi: 10.1007/978-3-030-03243-2_649-1.

[9] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, ‘DBSCAN Revisited, Revisited’, *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, Jul. 2017, doi: 10.1145/3068335.