Machine Learning Project Report

HARTH: A Human Activity Recognition

Student list:

Full name	Group	Section
Abdelhak Nesrine	**02**	**01**
KORICHI Anfal	**05**	**02**
Hamaidi Mohamed Idris Hamadi	**04**	**02**

Abstract

Human Activity Recognition (HAR) is a critical area of research with numerous applications in health monitoring, sports analytics, and human-computer interaction. In this study, we evaluated the performance of various machine learning algorithms on **HAR** tasks using a comprehensive dataset. We conducted experiments with six baseline algorithms: k-Nearest Neighbors (KNN), Artificial Neural Networks (ANN), Random Forest, Decision Tree, Support Vector Machine (SVM), and Naive Bayes. Our results indicate that the Random Forest algorithm exhibited the most potential for accurate **HAR**, outperforming other models. However, other benchmarked algorithms also demonstrated competitive performance. The findings from this study suggest that while Random Forest is a promising approach for **HAR**, other models also provide valuable benchmarks for future research.

Introduction:

Human Activity Recognition (**HAR**) has gained significant attention in the field of machine learning due to its wide range of applications, including healthcare monitoring, fitness tracking, and smart environments. **HAR** involves the automatic detection and classification of physical activities performed by individuals based on data collected from various sensors, such as accelerometers and gyroscopes. Previous work in HAR has utilized a variety of machine learning algorithms, ranging from traditional methods like Decision Trees and Naive Bayes to more advanced techniques such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM). These studies have demonstrated varying levels of success in accurately classifying human activities, often depending on the choice of algorithm, feature engineering techniques, and the quality of the dataset.

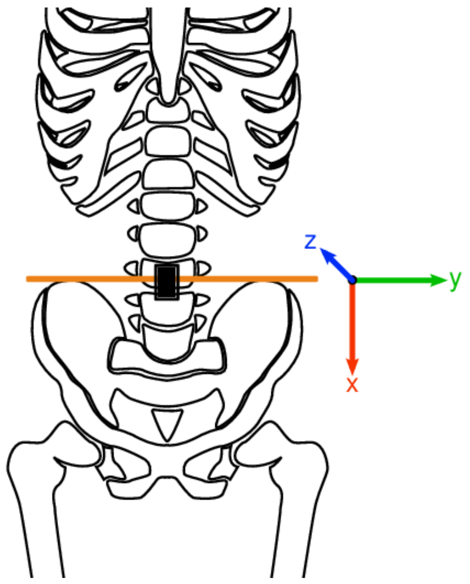
In this project, we aim to benchmark different machine learning algorithms to evaluate their performance on a **HAR** dataset. The algorithms compared in this study include K-Nearest Neighbors (KNN), Artificial Neural Networks (ANN), Naive Bayes, Support Vector Machines (SVM), Random Forest, and Decision Trees. By systematically comparing these algorithms, we seek to identify the most effective approach for **HAR**, considering factors such as accuracy, precision, recall, and computational efficiency.

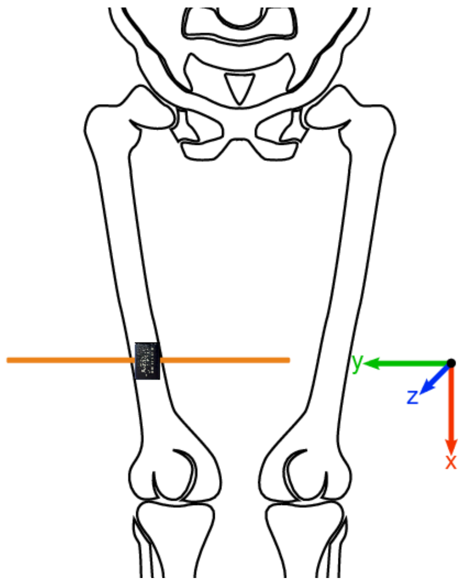
The dataset under consideration contains sensor data with six primary features representing the x, y, and z coordinates of accelerometer and gyroscope readings. The target variable consists of 12 activity levels, which poses a challenge due to the unbalanced nature of the data. To address this, we apply various data preprocessing techniques, including feature scaling, undersampling to balance the data, and feature selection to retain the most relevant attributes. Additionally, we explore the impact of including or excluding timestamp data on the performance of the algorithms.

This study's objective is to provide a comprehensive evaluation of these machine learning algorithms for **HAR**, highlighting their strengths and weaknesses in different scenarios. By doing so, we contribute to the ongoing research in **HAR** by offering insights into which algorithms are best suited for specific types of sensor data and activity recognition tasks.

Dataset Description:

The Human Activity Recognition Trondheim (HARTH) dataset, developed by NTNU Health, serves as a resource for training machine learning algorithms aimed at recognizing human activities. Comprising data from 22 individuals, each wearing two 3-axial accelerometers for approximately 2 hours in real-world settings, the dataset offers meticulously annotated information. The accelerometers, affixed to the right thigh and lower back, track participants' movements.





Recorded data is organized into separate .csv files, containing timestamped records and accelerometer measurements for both sensors. Activity annotations, derived from a chest-mounted camera's video signal, encompass twelve distinct activities, including walking, running, shuffling, ascending stairs, descending stairs, standing, sitting, lying, cycling while sitting, cycling while standing, inactive cycling while sitting, and inactive cycling while standing. These activities are coded as follow to ensure easy reference and facilitating algorithm training and evaluation:

1: walking   2: running   3: shuffling   4: stairs (ascending)   5: stairs (descending)

6: standing   7: sitting   8: lying   13: cycling (sit)   14: cycling (stand)

130: cycling (sit, inactive)   140: cycling (stand, inactive)

the csv files contain the mentioned attributes as follow:

1. timestamp: date and time of recorded sample(year,month,day,hour,minutes,seconds,milliseconds)

2. back_x: acceleration of back sensor in x-direction (down) in the unit g(the acceleration due to gravity)

3. back_y: acceleration of back sensor in y-direction (left) in the unit g

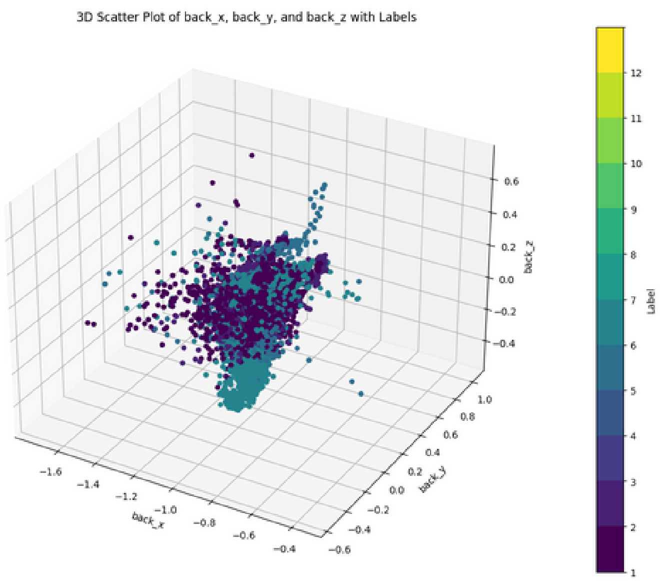
4. back_z: acceleration of back sensor in z-direction (forward) in the unit g

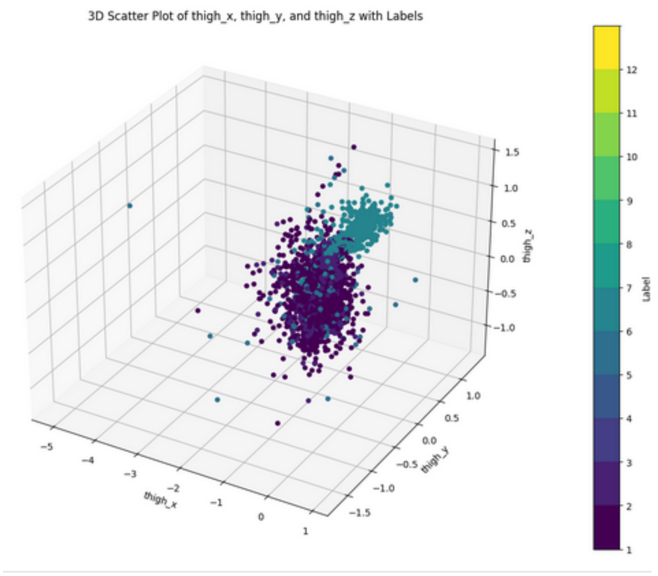
5. thigh_x: acceleration of thigh sensor in x-direction (down) in the unit g

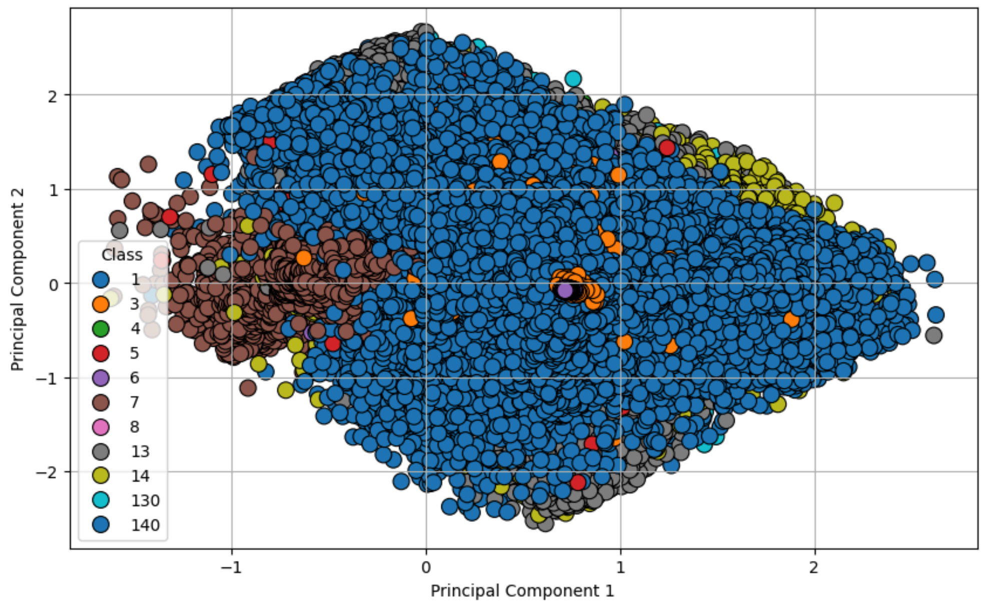
6. thigh_y: acceleration of thigh sensor in y-direction (right) in the unit g

7. thigh_z: acceleration of thigh sensor in z-direction (backward) in the unit g

8. label: annotated activity code







Data graph on 2-D plan

Methodology:

In this presented work , we are focusing on different approaches and a variety of algorithms between supervised and unsupervised to see the performance of each and evaluate the potential of the well known classifiers ,mainly presenting k-nearest neighbors,Artificial neural networks,naive bayes ,support vector machines,random forest and decision trees.We first started by exploring the provided dataset.

- **Feature Engineering and Selection Techniques:**

Feature engineering plays a crucial role in enhancing our machine learning models performance. The data was composed of 22 csv and all files were combined into a single dataframe for unified processing.After initial data exploration, the study focused on six primary features representing the x, y, and z coordinates. Feature importance and correlation analysis guided the decision to retain these features, while the relevance of timestamp data varied across algorithms. Models were compared using datasets both with and without timestamp information to understand algorithm behavior.

The target consists of **12** levels ,with the data visualization we see a clear unbalance in data ,so undersampling was taken in order to balance the data and evaluation and compare the balanced data results with the unbalanced one.

All the methods mentioned above was tested on mostly all the algorithm ,separated into with and without timestamp and applied on full dataset ,sampling and with undersampling

- **Model Training and Evaluation Methods:**

Cross-Validation: To mitigate overfitting and accurately estimate model performance, k-fold cross-validation (k=5) was applied, ensuring a variety of target values during the training process.

Data Sampling: Models were evaluated using the full dataset, samples comprising 10% or more of the data, and undersampled datasets to handle class imbalance.

- **Performance Metrics Used:**

Accuracy	**Precision**	**Recall**	**F1-score**	**Confusion Matrix**
Classification report	**Average class Accuracy**	**Roc and AUC**		

Focusing mainly on Accuracy ,Precision, Recall, and F1-score to provide insights into the classifier's performance for each target level.

- **Human Activity Recognition Models:**

1- Knn:

For the K-Nearest Neighbors (KNN) methodology, data scaling was applied since KNN is a distance-based model, although it did not significantly affect performance due to the close scale between features. A set of candidate models were presented, compared, and evaluated using different metrics. The candidates consisted of KNN on the full dataset, a 10% sample for comparison with other algorithms that cannot run on the full dataset, and undersampling for balancing the data. Each model was tested with and without the timestamp to capture the behavior of KNN with the presence of date information in the data. While accuracy was initially considered, the unbalanced nature of the data made accuracy misleading. Therefore, we focused on average class accuracy, recall, precision, and compared the classification report for each model candidate to ensure a comprehensive evaluation of the model's performance across all activity classes.

2- ANN :

For the Artificial Neural Networks (ANN) methodology, the algorithm was run and compared using different activation functions, including ReLU, Leaky ReLU, Sigmoid, and Swish, as well as various configurations of layers and neurons. These choices were determined after extensive experimentation to identify the optimal setup. A comprehensive comparative analysis was conducted to evaluate these configurations. The inclusion of the timestamp was also evaluated but was found to negatively impact accuracy. Due to the unbalanced nature of the data, undersampling was applied to balance the dataset. Comparative analysis was performed on the unbalanced data and the undersampled data. While undersampling resulted in lower overall accuracy, it showed significant improvement in the classification report and average class accuracy, indicating better performance across all activity classes.

3- decision tree:

The timestamp attribute was divided into year, month, day, hour, minute, and second components for training, and the dataset was cleaned by removing outliers using Z-scores. Model training was divided into three parts: full data, sampled data, and undersampled data, with the decision tree trained both with and without the time attributes in each part.

4-SVM :

Outliers were removed, and the timestamp was divided into components for training. Training the SVM model on the full dataset was challenging, so the model was trained using a sample and an undersample, both with and without the timestamp, to assess the impact of time on the predictions. The best model was then selected to compare its metrics with those of other algorithms.

5- naive bayes :

When features are correlated, they violate the Naive Bayes classifier's assumption of independence, potentially affecting performance. To address this, we will use Gaussian Naive Bayes, ensuring the features follow a normal distribution, and also bin the continuous data to apply Multinomial and Complement Naive Bayes with bins based on frequency and amplitude. Additionally, we will employ undersampling and oversampling strategies, handle features such as timestamps, and remove outliers using Z-score normalization and keep it. Furthermore, we will utilize incremental learning with the partial_fit function to update the model with new data post initial training, enhancing its adaptability and performance.

6- random forest

To address the computational expense associated with using a random forest algorithm for this case, we opted to train the model using samples consisting of 10% and 20% of the entire dataset. The remaining data was used for testing to ensure generalization.

Additionally, we employed various techniques to preprocess and enhance the data:

- **Undersampling**: We applied different undersampling strategies and weights to balance the classes.
- **Class Weight Adjustment**: We assigned different penalties to the classes to address class imbalance.
- **Outlier Treatment**: We experimented with removing outliers using Z-score normalization and compared the results with keeping the outliers.
- **Feature Engineering**: We included and excluded timestamp features such as year, month, and day to evaluate their impact on model performance

- **Hyperparameter Optimization:**

The hyperparameter optimization with cross-validation was carried out by using the validation train and test taken from the dataset (15% from the dataset size),on the level of **Knn algorithm**, focusing on the following parameters :n_neighbors,weights,algorithm and metric with a set of values ,the hyperparameter resulted in Knn with k=7

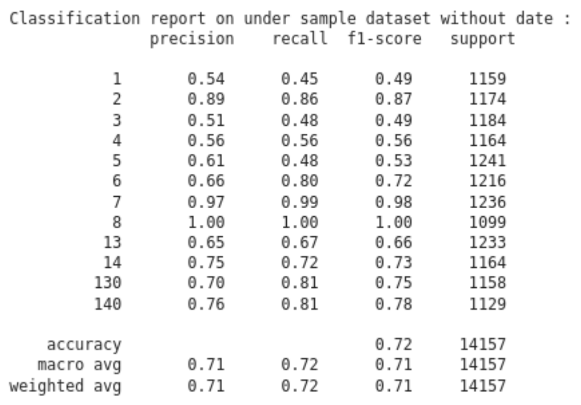
Results and Analysis:

In the next section, we will present the performance evaluation of each algorithm, conduct a comparative analysis with visualizations, and discuss the key insights and conclusions derived from the results.

Table of the different metrics for the best models found for each algorithm on the HAR dataset:

	Knn	Ann	Naive bayes	SVM	Decision tree	Random forest
Accuracy	0.7172	0.64	0.78322	0.67	0.90	0.92136
Recall	0.7131	0.65	0.78322	0.68	0.78	0.92136
Precision	0.7172	0.65	0.75184	0.67	0.77	0.92161
F1-score	0.713	0.65	0.76061	0.67	0.78	0.91373

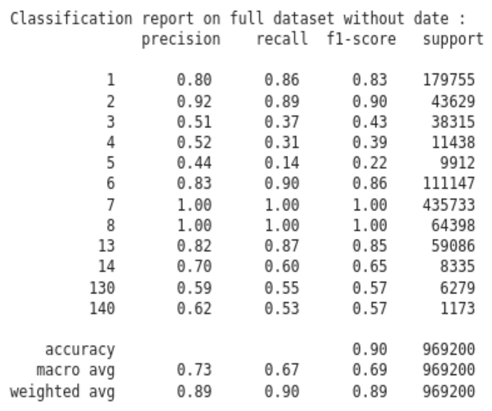
On the level of **Knn**,We have discussed several candidate models ,with the hyper parameter tuning and the manual experiments taken ,resulted in Knn with **k=7** as best model , then a set of tests was taken with this best model,by comparing the use of full data and undersampling both with and without timestamp ,we noticed that the model on full data has high accuracy(0.8986) as shown in this table for analysis_sheet, but when coming to further analysis we see on the classification report on full data that it did not predict well the target levels that are the minority in the dataset especially with levels **4** and **5** ,while is the opposite remark on the undersampled data , in which the model resulted in lower accuracy 0.7172 (see the table above as it is taken as best model in knn) but with better results on the classification report in which using balanced data give the model the ability to predict all the target levels ,see the comparison on the classification report below



```
Classification report on under sample dataset without date :
              precision    recall  f1-score   support

           1       0.54      0.45      0.49      1159
           2       0.89      0.86      0.87      1174
           3       0.51      0.48      0.49      1184
           4       0.56      0.56      0.56      1164
           5       0.61      0.48      0.53      1241
           6       0.66      0.80      0.72      1216
           7       0.97      0.99      0.98      1236
           8       1.00      1.00      1.00      1099
          13       0.65      0.67      0.66      1233
          14       0.75      0.72      0.73      1164
         130       0.70      0.81      0.75      1158
         140       0.76      0.81      0.78      1129

    accuracy                           0.72     14157
   macro avg       0.71      0.72      0.71     14157
weighted avg       0.71      0.72      0.71     14157
```



```
Classification report on full dataset without date :
              precision    recall  f1-score   support

           1       0.80      0.86      0.83    179755
           2       0.92      0.89      0.90     43629
           3       0.51      0.37      0.43     38315
           4       0.52      0.31      0.39     11438
           5       0.44      0.14      0.22      9912
           6       0.83      0.90      0.86    111147
           7       1.00      1.00      1.00    435733
           8       1.00      1.00      1.00     64398
          13       0.82      0.87      0.85     59086
          14       0.70      0.60      0.65      8335
         130       0.59      0.55      0.57      6279
         140       0.62      0.53      0.57      1173

    accuracy                           0.90    969200
   macro avg       0.73      0.67      0.69    969200
weighted avg       0.89      0.90      0.89    969200
```

Along with comparing balanced and unbalanced data, we examined the effect of including timestamps, which were aggregated into year, month, day, hour, minute, second, and millisecond. The KNN model (k=7) was run on full and undersampled data, with and without timestamps. Results were evaluated using accuracy, confusion matrix, and classification report, consistently showing that models without timestamps outperformed those with timestamps.

On the level of **Ann**,a variety of models were selected to be the candidates,by first determining the best hyperparameters to use.**Relu** ,**Leaky Relu,Swish** and **Sigmoid** were competing on giving the best model accuracy ,resulting in having the ANN with Relu as activation function ,2 layers and 32 neurons on the first and 16 neurons on the second layer,after determining the best models hyperparameters ,a further study on the data were conducted ,in which the undersampled data with accuracy 0.6441 outperformed the model on full data with accuracy 0.8743 , the reason for not considering only the accuracy in here is in the unbalance in data where the minority targets are not predicted or predicted with low values ,while the accuracy was high while using the full dataset yet the average class accuracy was 0.51 while on undersampled data we got average class accuracy with 0.65 (find all results in this analysis_sheet) ,the classification report and others plots proved this as well,For the use of the timestamp with Ann ,the results were always better without the use of timestamp.

In the **decision tree** model, the training was performed using three different data frames: full data, sampled
data, and undersampled data, each with and without timestamps. Training on the full dataset without time attributes resulted in an accuracy of 0.85 and an F1 score of 0.59 for all classes. Including the timestamps improved the results, achieving an accuracy of 0.90 and an F1 score of 0.78. The inclusion of timestamps increased the model's accuracy since the data is a time series, and the time attribute helps the model capture dependencies between consecutive classes. For the sampled data, similar results were observed, with the timed dataset performing better (accuracy of 0.87) compared to the one without time (accuracy of 0.83), with a small decrease in accuracy due to the reduced data size. In the undersampling approach, where the data was balanced by reducing all class sizes to match the minority class, the accuracy decreased further due to the size reduction. The timed dataset still performed better than the one without time, with accuracies of 0.74 and 0.62, respectively.

For the **SVM models**, we aimed to determine if the timestamp affected the algorithm similarly to the decision tree, so we included the time attributes and trained the model on sampled and undersampled datasets, excluding the full dataset due to its large size. Through hyperparameter tuning, we identified the Radial Basis Function (RBF) kernel and a regularization parameter of C=10 as optimal. For the sampled dataset, the model without timestamps outperformed the one with timestamps, achieving an accuracy of 0.85 and accurately predicting instances from all classes. In contrast, the timed sampled dataset only predicted instances from the majority class, resulting in an accuracy of 0.49, indicating that timestamp attributes negatively impacted the SVM model. For the undersampled dataset, the model with time attributes could predict other classes but had a low accuracy of 0.14, whereas the model without timestamps performed better with an accuracy of 0.67 and the ability to predict all classes.

For the **random forest**, we conducted extensive testing using various techniques such as sampling methods, feature engineering, and outlier handling. The best model we achieved was a random forest with 100 n-estimators and an average tree depth of 46 on 10% of the data. This model was trained on data with timestamps aggregated into year, month, and day, resulting in an accuracy of 92%. The classification report was robust, indicating that the model effectively predicted most classes, including those with very few instances, the remaining 90% of data was used to test the model to assure it was trained well, and the results remained great, with 92% accuracy.

For the **Naive Bayes classifier**, we tested multiple models using different variants such as Gaussian, Complement, and Multinomial Naive Bayes. We applied various techniques, including hyperparameter tuning of smoothing, binning data based on frequency and amplitude, and employing sampling strategies. The model that performed best overall, with an accuracy of 78%, was one where outliers were removed using Z-score and timestamps were excluded. However, this model struggled to predict certain labels accurately, performing well only for classes 1, 6, 7, and 8.

We also tested another model using oversampling without timestamps, which provided the best results for classes 4, 5, 14, 130, and 140, despite a lower overall accuracy of 50%. Additionally, an undersampling model without timestamps performed well for classes 2, 3, and 13, achieving an accuracy of 60%.

In summary, we identified three models that predicted specific classes effectively, highlighting the varied performance across different subsets of the data.

Discussion :

The **Decision Tree** achieved its best performance with the full dataset including time attributes, with an accuracy of 0.90. However, it struggled to accurately predict class 3, despite the class's significant representation in the dataset.

The best-performing **SVM** model, which used the "RBF" kernel with C=10 on the undersampled data without time attributes, achieved an accuracy of 0.67. However, it had weak prediction capabilities for classes 1, 3, and 4, with f1 scores around 0.4.

On the **knn**,the analysis resulted in choosing k=7 and training the model on undersampling data without the timestamp that effectively gave remarkable results.

Ann resulted in good results in all the tests but only one was chosen to compete with other machine learning algorithms ,which is an undersampled data model with Relu .

The **random forest** model achieved a high accuracy of 92% and performed well across most classes, including those with few instances. However, the model's depth and computational cost were significant, making it slow to run on the entire dataset.

On the other hand, the **Naive Bayes** models had varied accuracy, with the best achieving 78%, and showed differential performance across classes. Some classes were poorly predicted due to the model's assumption of conditional independence, which is uncommon in real-life scenarios. Future improvements could focus on optimizing the complexity of the random forest model and enhancing feature engineering and sampling techniques for Naive Bayes to better handle class predictions.

As a result from all these algorithms ,we have selected the best model in each one ,but the one that outperform all was Random forest

Potential improvements and future work

After this deep analysis on each algorithm ,we see the potential in random forest ,but this does not stop us from going further to combine the best performing algorithms with random forest ,for future work we suggest the use of ensemble learning to combine these classifiers

Conclusion

With the study conducted on the HAR dataset, which consists of over 6 million instances and 12 target classes, we thoroughly analyzed and compared six different machine learning algorithms: **ANN**, **KNN**, **SVM**, **Naive Bayes**, **Decision Tree**, and **Random Forest**. Each algorithm was evaluated extensively to identify the best-performing model in each category. We conducted a deep analysis of each algorithm's performance using various metrics such as accuracy, precision, recall, and the F1-score. By comparing the performance of each algorithm, we aimed to determine the most effective approach for Human Activity Recognition (HAR). Based on our evaluation, we identified the best model for HAR, considering factors such as model accuracy, computational efficiency, and the ability to handle the dataset's scale and complexity. This project contributes to the field by providing insights into the strengths and weaknesses of different machine learning algorithms for HAR, helping to guide future research and application development. The findings have significant implications for applications in healthcare monitoring, fitness tracking, and smart environments, where accurate human activity recognition is crucial.

References and links

Link to all our work can be found in this drive

Main paper

Multiclass Classification Using Support Vector Machines

Why do tree-based models still outperform deep learning on tabular data?

Who did what in the project?

Tasks	Who did it
Knn model	Abdelhak Nesrine
Ann model	Abdelhak Nesrine
Naive bayes	Hamaidi Mohamed Idris Hamadi
Random forest	Hamaidi Mohamed Idris Hamadi
Decision tree	KORICHI Anfal
SVM	KORICHI Anfal
Report and analysis	All the team