Human Activity Recognition Using Data from Smartphone Sensors

Aairah Bari
IIIT Delhi
aairah19003@iiitd.ac.in

Dhairya Chaudhary
IIIT Delhi
dhairya190035@iiitd.ac.in

Harshit Gulati
IIIT Delhi
harshit19044@iiitd.ac.in

Kirthana Natarajan
IIIT Delhi
kirthana190053@iiitd.ac.in

Abstract

Human Activity Recognition is an active research area that has wide applications in healthcare and human survey systems. Most smartphones have accelerometer and gyroscope sensors, data from which can be used for human activity recognition. For many people, mobile gadgets have become an inseparable part of their everyday lives. People have their phones with them at all times of the day, whether they are driving, exercising, walking, working, or taking a break. Having information about what users are doing helps applications communicate intelligently. It can also help caretakers monitor the activity of patients and the elderly, and provide assistance when anomalous activities such as falling are detected. Thus, human activity recognition from smartphone accelerometer and gyroscope data is easily accessible and widely applicable. Here we use machine learning methods to classify human activity using data collected from smartphone sensors.

***Repository**: https://github.com/dhairyachaudhary/ML-Project*

1. Introduction

Activity detection is one of the most basic and common activities performed by humans on a daily basis. This ability is what makes us context aware and helps shape our decisions. Similar awareness in our devices makes them much more intelligent and interactive.

Considering the way phones have completely morphed into the lives of users, using the data constantly being generated by the phone accelerometer and gyroscope can bring about such awareness.

Our project aims to classify human activity as 'Standing', 'Walking', 'Walking Upstairs', 'Walking Downstairs', 'Sitting', or 'Laying' given smartphone accelerometer and gyroscope readings. We trained different machine learning models (Naïve Bayes, Decision Trees, Random Forest, Logistic Regression, SVM, K-Means, KNN, and ANN) using a dataset from UCI's Machine Learning Repository for this purpose and compare their performances to select the best model.

2. Literature Survey

Research on human activity recognition has been prevalent for a long time now and academics have suggested several solutions to the problem. The current methods usually involve some combination of sensors worn by people who perform some activities which are classified using machine learning or threshold base algorithms. ML algorithms usually perform better but incur more cost [4]. Since deep learning techniques need a large quantity of training data, many researchers rely on general machine learning approaches [3].

Various researchers have used different models like Logistic Regression, Naive Bayes, Decision Trees, Random Forest, SVMs, Nearest Neighbor, Hidden Markov Model, CNN, and RNN to tackle the problem and proposed different solutions [1, , 3, 4]. In our project, we use these approaches and evaluate and compare them for our dataset.

3. Dataset & Preprocessing

We have used a Kaggle dataset titled 'Human Activity Recognition with Smartphones' provided by UCI Machine Learning and licensed under a CC0. It contains data collected from 30 volunteers who performed activities of daily living while wearing a Samsung Galaxy S II mounted on their waist. Sensor readings from the device are present in the dataset

The activities performed fall into the following categories- Standing, Walking, Walking Upstairs, Walking Downstairs, Sitting, Laying. Each entry in the dataset contains:

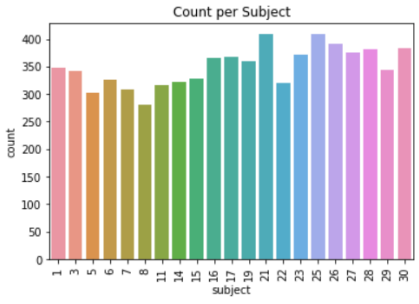
- Triaxial acceleration (from accelerometer) and estimated body acceleration
- Triaxial Angular velocity (from gyroscope)
- A 561 feature vector with time and frequency domain variables
- The label

The raw data had been extracted, pre-processed, normalized, and divided into training and testing data using a 7:3 split. We further processed the data by performing dimensionality reduction.

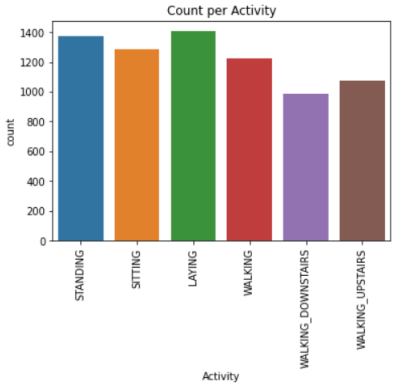
3.1 EDA

We started by data cleaning and found no null or duplicate values in the dataset.

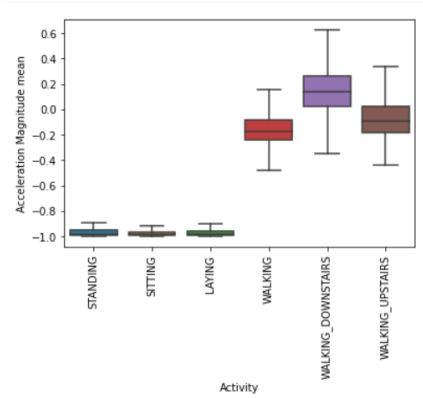
To ensure that our data is not biased towards some volunteers, we plot count of entries per subject and data per activity per subject and find no significant discrepancy.



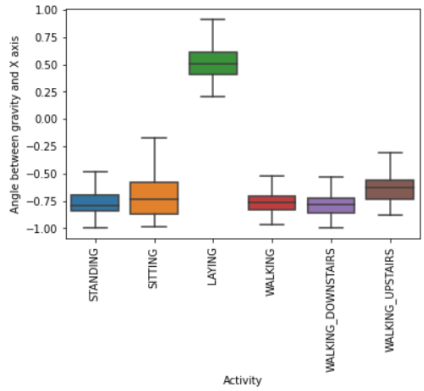
Similarly, we check to see if the data is balanced across different classes. We observe that it is fairly balanced.



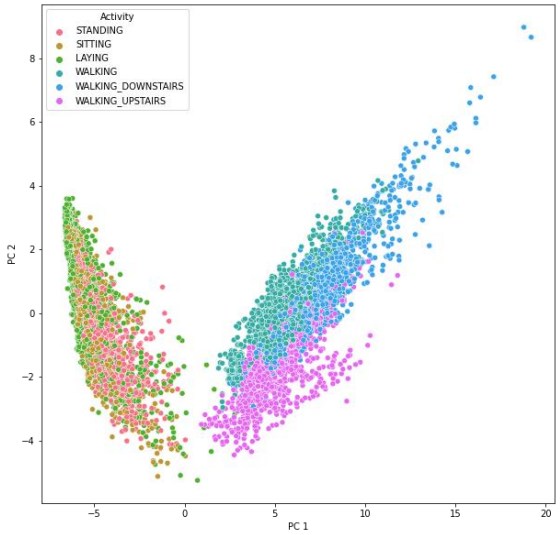
Box plots for mean magnitude of acceleration show clear distinction between static (Laying, Sitting, Standing) and dynamic (Walking, Walking Upstairs, Walking Downstairs) activities.



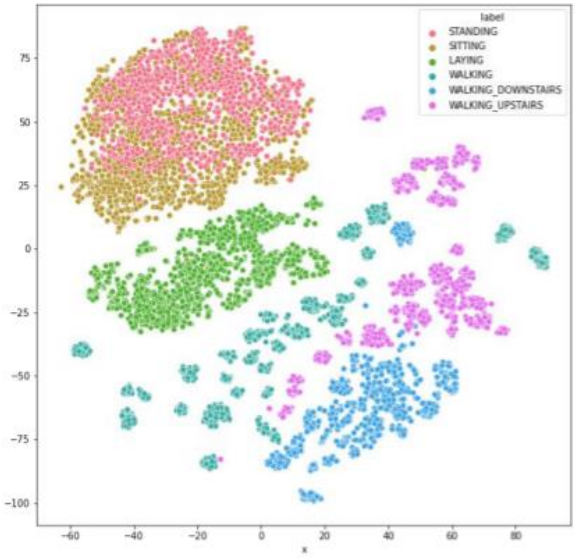
Box plots for angle between gravity and x axis show separability between laying and other classes.



Principal Component Analysis (PCA) is used to transform high dimensional data to lower dimensions and can be used to visualize high dimensional data. We plotted the first two principal components which explain 68.75% of the variance in our data. We notice that the dynamic and static activities are easily separable.



t-SNE (t-Distributed Stochastic Neighbor Embedding) is a technique used to visualize high dimensional datasets. It creates a mapping of the data points in lower dimensions by considering the local relationships between the points, which makes it a better choice to account for non-linear structure in the data as compared to PCA. From the plot, we can clearly distinguish 'Laying' activity while the other static and dynamic activities are much harder to distinguish from each other.



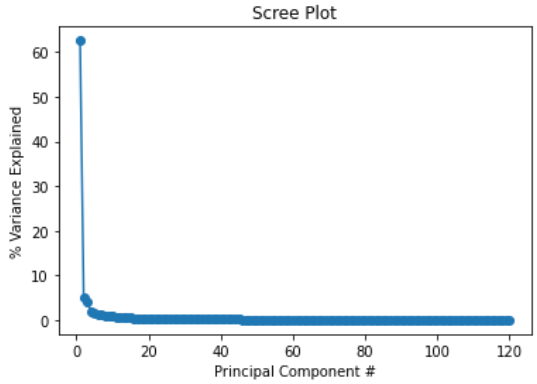
3.2 Dimensionality Reduction

Our dataset consists of 561 features. To reduce the dimensionality, we explored feature selection of k best features using ANOVA f_classif scores based on relevant literature [3] and explored the Principal Component Analysis (PCA) feature extraction method.

Principal Component Analysis (PCA) is a feature extraction method that is used for dimensionality reduction. We chose to retain 120 components as 98.14% of the variance in our data is explained by 120 components.

SelectKBest is a univariate feature selection method that chooses the K best features based on statistical tests. We used the Analysis of Variance f_classif scores based on the literature we surveyed. We chose the value of K as 120 to compare with PCA with 120 components.

PCA performs better in our case. PCA generates new uncorrelated components such that the variance is maximized.



Cumulative Variance: 98.14%

4. Methodology

We tried four models from the python sklearn library to classify our dataset - Gaussian Naïve Bayes, Decision Tree, Random Forest and Logistic Regression. 5-fold cross validation was used to tune the hyperparameters for all the models.

4.1 Gaussian Naïve Bayes

A Naïve Bayes classifier is a simple "probabilistic classifier" based on Bayes' theorem. It assumes strong (naïve) independence between the features. This assumption of independence lends Naïve Bayes its simplicity. Thereby making it widely popular as a baseline model. Feature vectors in our dataset contains continuous data, so we use Gaussian Naïve Bayes which presumes Normal Distribution of feature values.

4.2 Decision Trees

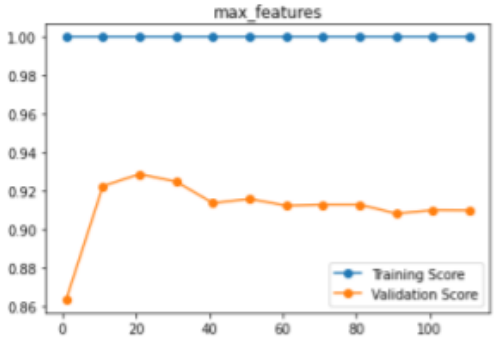
Decision Trees are supervised classifiers, which use the training data to construct simple tests for class prediction. The Decision Tree Classifier of sklearn when used with default parameters gives an unpruned tree which overfits the training data. We use validation curves to tune the hyper parameters and get a generalized tree with low variance.

Nonetheless, we should also remember that a Decision Tree is used because of its interpretability and not accuracy. To classify the data with a better accuracy, we instead use random forests.

4.3 Random Forest

Random Forest uses an ensemble approach by aggregating over several decision trees. Trees are generated from bootstrapped samples that include randomly chosen samples with replacement. At each split, the best feature is selected as the node from a randomly chosen subset of features.

We performed Randomized Search CV over a range of possible values for the hyperparameters: number of estimators, maximum depth of the decision trees, maximum number of features from which to select the best feature at each split. We plotted validation curves for each of the above mentioned hyperparameters for selected values.



4.4 Logistic Regression

Logistic regression is a classification model that transforms its output using the logistic sigmoid function to return a probability value.

We used sklearn's implementation of logistic regression with the liblinear (Library for Large Linear Classification) solver. We used GridSearchCV with the various solvers included in sklearn's logistic regression model (newton_cg, sag, saga, lgbf, liblinear). Liblinear uses coordinate descent algorithm and performs one-vs-rest multi-class logistic regression. Liblinear performs well with high dimensionality, therefore it works well for our case.

4.5 Multilayer Perceptron

A multilayer perceptron (MLP) is a feedforward artificial neural network that generates a set of outputs from given features. An MLP is characterized by several layers of input nodes connected as a directed graph between the input and output layers.

We used sklearn's implementation of multilayer perceptron and used GridSearchCV with the various activation functions to figure out the ideal number of hidden layers and neurons. For our problem, the tanh activation function worked best with a single hidden layer with 100 neurons.

4.6 SVM

Support Vector Machines are supervised machine learning models that find optimal hyperplanes to separate the data. SVM algorithm creates higher dimensional space from lower dimension input using a function called SVM kernel such that the data becomes more easily separable. We performed hyperparameter tuning using grid search over a range of values for C, the regularisation parameter, gamma, the kernel coefficient and tried the linear, poly and rbf kernel types with a range of values for degree of poly kernel.

4.7 Adaboost

AdaBoost, short for Adaptive Boosting, is another ensemble classifier. It also uses multiple Decision Trees (weak classifiers) to classify data just like Random Forests. However, it also adapts after each iteration by tweaking subsequent weak learners in favor of those instances misclassified by previous classifiers.

In our final AdaBoost model, we used 50 decision trees with max depth = 15. We plotted validation curves to tune the above hyper parameters.

4.8 KNN

KNN (K-nearest neighbours) is a supervised distance-based machine learning model. It presumes similarities between a new sample and existing data and places it in the category where it is most similar to the existing members. We performed hyperparameter tuning using sklearn's GridSearchCV for the number of neighbors, leaf size, and power parameter and found the values to be 8, 3, and 1.

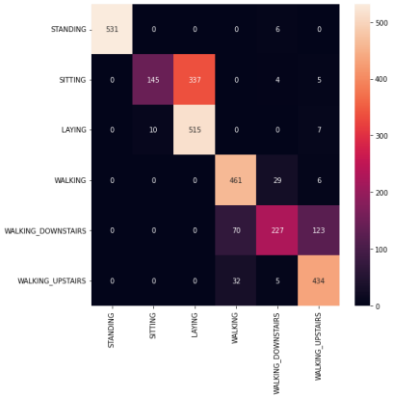
5. Results & Analysis

We can observe that performance is best for 'Laying' which can be explained since the maximum data is available for Laying and is distinctly different from the other activities as observed in EDA. We also observe that sitting, laying and walking, walking upstairs, walking downstairs are often confused, which is consistent with our observations from EDA.

5.1 Gaussian Naïve Bayes

The Gaussian Naive Bayes model trained using the default parameters yielded an accuracy of 77.28% on the testing data.

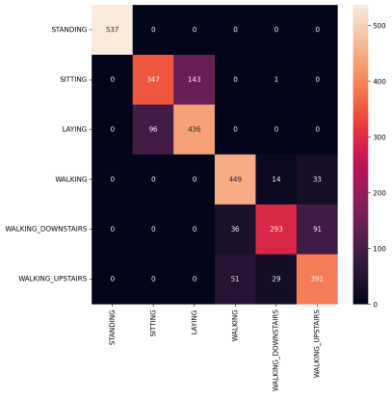
We then used hyperparameter tuning for var_smoothing and used that value to fit the model. This improved the accuracy to 82.25%. This behavior is in line with expectation as Naïve Bayes is a baseline model and assumes independence of features which is not true for our dataset.



5.2 Decision Tree

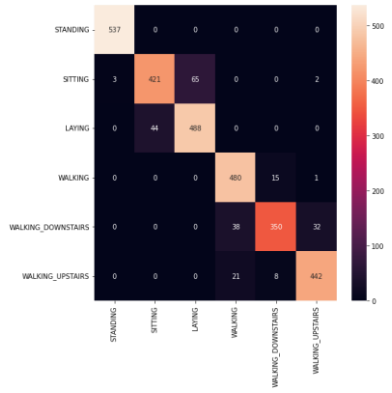
We were able to achieve an accuracy of 80.35% using the Decision Tree Classifier. The results are satisfactory, especially considering that Decision Trees are weak classifiers.

Through the visual representation of the tree, we observe that our model is able to separate static and dynamic activities using only a few tests and misclassifies similar activities only for a small fraction of the dataset. Based on these results, we expect the Random Forest method to perform reasonably well.



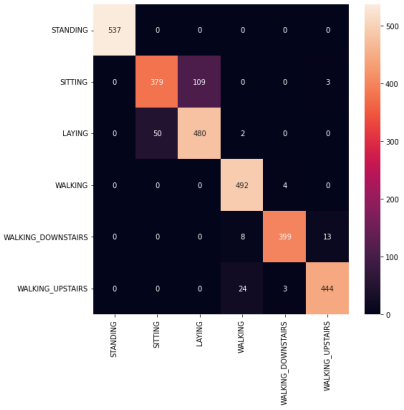
5.3 Random Forest

Generalizability and accuracy is improved as compared to individual decision trees as randomness is introduced while growing the trees. As expected, the random forest classifier performed much better on our dataset as compared to the decision tree classifier. Using Randomized Search CV, 91.72% accuracy was obtained with the best parameters of 184 trees, maximum features using sqrt, maximum depth of 23, criterion as entropy. Validation curve plots showed that as the number of estimators and maximum depth of trees is increased, the performance of the model improves up to a point after which further increase does not have any effect. As the maximum number of features used in each split to determine the best feature increases, the performance increases initially but performance starts reducing after a point.



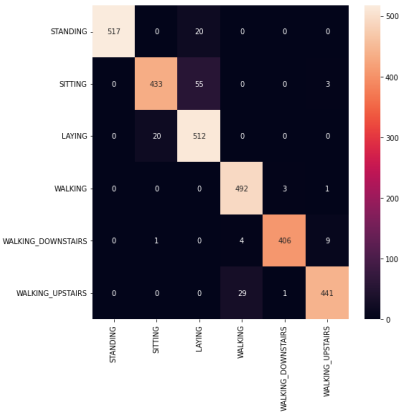
5.4 Logistic Regression

Logistic regression achieves an accuracy of 95.11%. We found that previous research has also achieved good results using logistic regression, and one of the papers we studied also considered it as a benchmark model [3]. In higher dimensional spaces, it is easier to separate data linearly and logistic regression functions well with such data. The simple model is able to achieve better generalization, hence this result is expected.



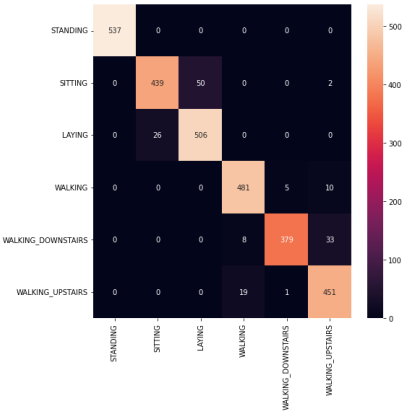
5.5 Multilayer Perceptron

We were able to achieve an accuracy of 95.01% using MultiLayer Perceptron with 1 hidden layer with 100 neurons, after hyper parameter tuning to find the ideal number of layers and neurons. This model had a good performance and was among the top performing models studied. This can be attributed to MLPs ability of using hidden layers to enhance separability.



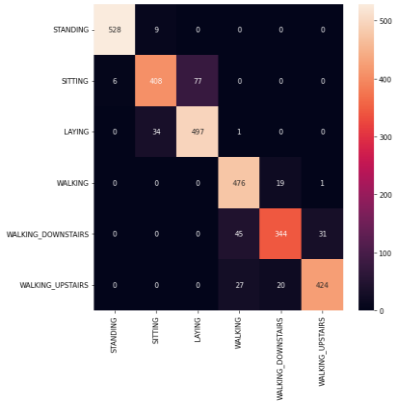
5.6 SVM

Support Vector Classifier achieves an accuracy of 94.77%. After hyperparameter tuning using grid search, we chose the best performing hyperparameters which gave accuracy of 95.31%. Hyperparameters: rbf kernel, gamma 0.01, degree 1, C 1000. SVMs perform well on both linear as well as non-linearly separable data. They tend to underperform when the dataset is imbalanced or when the data is noisy, neither of which is the case with our dataset. SVMs perform well on high dimension classification problems. Accordingly, SVC model performed well here.



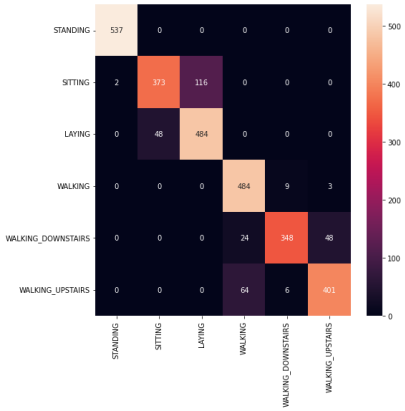
5.7 Adaboost

Our AdaBoost classifier, with 50 decision trees of max depth 15, classified the test data with an accuracy of 90.56%. Increasing the number of estimators may further improve the performance marginally, but that would result in higher computational complexity.



5.8 KNN

KNN yielded an accuracy of 88.25 % for our problem with the default parameters. After performing hyperparameter tuning using GridSearchCV, we achieved an accuracy of 90.60%. KNN's performance is not comparable to the performance of models such as SVM and Logistic Regression since it does not perform well with larger datasets and higher dimensional data.



5.5 Summary

Classifier	**Accuracy**
GNB	82.25%
Decision Tree	80.35%
Random Forest	91.72%
Logistic Regression	95.11%
MLP	95.01%
SVM	95.31%
Adaboost	90.56%
KNN	90.60%

6. Conclusion

6.1 Learnings

We started the project by exploring our problem and the importance of a robust solution. Through the course of the project, we got familiarized with all steps of the design cycle of a machine learning problem and understood the importance of all steps. We learned about multiclass classification and the applications and intricacies of different machine learning models, and how to evaluate and choose a model for a given problem.

6.2 Future Work

A future problem to be addressed is the capacity of a human activity categorization system to mimic humans' proficiency in detecting human behaviours in real time. Machine-learning algorithms that include knowledge-driven approaches may be critical for human activity modelling and recognition in unconstrained situations. Other possible interesting research avenues are to predict the next action of a person based on current activity and detecting activities performed together by groups of humans.

6.3 Individual Contribution

Aairah Bari: Literature review, EDA, Naïve Bayes, MLP

Dhairya Chaudhary: Literature review, Logistic Regression, KNN, Report

Harshit Gulati: Literature Review, Feature Selection - PCA, Decision Trees, Adaboost

Kirthana Natarajan: Literature Review, Feature Selection – PCA and Select K, EDA, Random Forest, SVM

7. References

[1] X. Yin, W. Shen, J. Samarabandu and X. Wang “Human activity detection based on multiple smart phone sensors and machine learning algorithms” 2015 IEEE 19th International Conference on Computer Supported Cooperative Work in Design (CSCWD), 2015, pp. 582-587, doi: 10.1109/CSCWD.2015.7231023.

[2] Khimraj, P. K. Shukla, A. Vijayvargiya and R. Kumar, "Human Activity Recognition using Accelerometer and Gyroscope Data from Smartphones," 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3), 2020, pp. 1-6, doi: 10.1109/ICONC345789.2020.9117456.

[3] J. Rabbi, F. Tahmid and Md. Awal, “Human Activity Analysis and Recognition from Smartphones using Machine Learning Techniques” 2021 arXiv:2103.16490

[4] A. Rasekh, C. Chen and Y. Lu, “Human Activity Recognition using Smartphone” 2011 Texas A & M University Fall Projects

[5] U. Khan and S. Masood, "A Machine Learning Approach to Human Activity Recognition," 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), 2020, pp. 167-171, doi: 10.1109/PDGC50313.2020.9315826

[6] T. Dinh Le and C. Van Nguyen, "Human activity recognition by smartphone," 2015 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS), 2015, pp. 219-224, doi: 10.1109/NICS.2015.7302194.