

**Report on**  
**Human Activity Recognition**  
**Smartphone Sensor Data**

# Summary and Dataset

## *Description of the Experiment*

In this experiment, data was collected from a group of 30 volunteers, aged between 19 and 48 years. Each participant performed six different activities while wearing a smartphone on their waist. The activities included:

- **WALKING**
- **WALKING UPSTAIRS**
- **WALKING DOWNSTAIRS**
- **SITTING**
- **STANDING**
- **LAYING**

The smartphone was equipped with an **accelerometer** and a **gyroscope**, which captured the following data at a constant sampling rate of 50Hz:

- 3-axial linear acceleration (from the accelerometer)
- 3-axial angular velocity (from the gyroscope)

In addition to the sensor data, the experiments were video recorded to manually label the data. The dataset was randomly split into two sets:

**70%** of the data was used for training the machine learning models.

**30%** of the data was used for testing the model's performance.

## Data Preprocessing

Before using the raw data for analysis, several preprocessing steps were applied to ensure high-quality input data for model training:

### Feature Extraction

For each window of data, a feature vector was created by computing various statistical variables from both the **time domain** and **frequency domain**. These features were then used to train machine learning models to classify the different activities.

### Dataset Variables

Each record in the dataset contains the following data attributes:

- **Triaxial Acceleration:** The accelerometer readings representing the total acceleration (which includes both body acceleration and gravitational components) as well as the estimated body acceleration.
- **Triaxial Angular Velocity:** The gyroscope readings that represent the angular velocity along the three axes (X, Y, and Z).

These features form the basis for classifying the activities performed by the participants.

## Training and Evaluating Models

Once the dataset is pre-processed and the features are extracted, the next step is to train machine learning models to classify the different activities based on the sensor data.

## Splitting the Data:

The first step in training the model is to divide the dataset into **features (X)** and **labels (y)**. Features represent the sensor data, and labels represent the activities to be classified.

- **Training Set (70% of the dataset):** This set is used to train the machine learning model.
- **Testing Set (30% of the dataset):** This set is used to evaluate the model's performance after training.

## Machine Learning Models Used

The following models were used for classification:

Logistic Regression

K-Nearest Neighbors (KNN)

Support Vector Classifier (SVC) with RBF Kernel

Support Vector Classifier (SVC) with Linear Kernel

Random Forest Classifier

Decision Tree Classifier

## Model Training and Evaluation

To train and evaluate the models, the following steps were followed:

**Splitting Data:** The data was split into training and testing sets. The training set contained 70% of the data, while the test set

contained 30% of the data. This split ensures that we can assess how well the model generalizes to new, unseen data.

**Model Initialization:** Each model was instantiated, trained, and evaluated. The training process involves fitting the model to the training data ( $X_{\text{train}}$  and  $y_{\text{train}}$ ), and performance was evaluated using the test data ( $X_{\text{test}}$  and  $y_{\text{test}}$ ).

**Model Evaluation:** After training the models, the performance of each model is evaluated using metrics such as **accuracy**, **precision**, **recall**, and **F1-score**. These metrics help assess how well each model performs on the test data

## Results Interpretation

1. **Logistic Regression:** Logistic regression performed well with linear boundaries and is computationally efficient. However, it may struggle with more complex datasets or non-linear relationships.
  - Excellent **generalization performance** with a **small gap (~3.3%)** between training and test accuracy.
  - Performs **better than KNN, Decision Tree, and Random Forest**.
  - Slightly **below Linear SVC**, but still among the **top-performing models**.
2. **K-Nearest Neighbour (KNN):** KNN performed well if the data is not high-dimensional. However, it is computationally expensive as it needs to calculate distances to all other data

points during prediction. Its performance depends heavily on the choice of  $k$  and the data distribution.

- **Good training accuracy**, but the **gap** (~6%) between training and testing indicates **mild overfitting**.
- As a **lazy learner**, KNN relies heavily on the training data and can struggle with high-dimensional or noisy features.
- Performance is **lower than SVCs** and Random Forest, but still acceptable

### 3. Support Vector Machine (SVC):

- a. **SVC with RBF Kernel**: This model captures complex, non-linear boundaries and often works well for high-dimensional data.

**RBF SVC** (Support Vector Classifier with Radial Basis Function kernel) also delivers **strong performance**, though not quite as high as the Linear SVC

- lightly lower performance than Linear SVC on both training and test sets.
- RBF kernel introduces **non-linear decision boundaries**, which is helpful when data isn't perfectly linearly separable.
- Still generalizes very well — **only ~2.86% gap** between training and test accuracy.
- Compared to Random Forest and Decision Tree, it's more balanced and less prone to overfitting

- b. **SVC with Linear Kernel**: This model is efficient and works well for linearly separable data. It has lower complexity compared to the RBF kernel version but may not perform well on non-linear data.

**Linear SVC (LBasedImpl)** model shows excellent performance with **very high accuracy, precision, and recall** on both the training and test sets

- The model generalizes very well with only a **2.8% gap** between training and test scores.
- High **recall** suggests it's correctly identifying almost all activity classes.
- Minimal **overfitting** compared to Decision Tree or Random Forest.
- Linear SVC works especially well with high-dimensional, linearly separable data — consistent with pre-processed feature-rich dataset.

4. **Random Forest:** Random Forest is an ensemble method that combines several decision trees, providing robust performance even with complex data. It is less likely to overfit than a single decision tree.

**Perfect scores on training:** Indicates the model has learned the training data very well.

**High test performance:** A test accuracy and precision of **92.5%** is excellent and shows that the model generalizes well to unseen data — though not perfectly.

5. **Decision Tree:** A single decision tree be prone to overfitting, but it is easy to interpret. It's best used when the data has clear decision boundaries.

**DecisionTreeClassifier** shows **perfect training performance** but significantly **lower test performance**, indicating **strong overfitting**

- **100% training scores** indicate the model memorized the training data.
- **14% drop in test accuracy/precision/recall** is a strong signal of overfitting.
- Decision Trees tend to create overly complex models if not pruned or regularized.

### ***Training and Testing Performance Plots***

Once the models have been trained and evaluated, plotting the results for **accuracy**, **precision**, **recall**, and **F1-score** helps visualize their performance.

### ***Hyperparameter Tuning and Cross-Validation***

- **Grid Search:** Hyperparameters like the number of neighbours for KNN, the C and gamma parameters for SVC, or the number of trees in Random Forest can be tuned to achieve better performance.
- **Cross-Validation:** Cross-validation can be used to further evaluate each model's stability by splitting the data into multiple folds, training the model on each fold, and evaluating it on the remaining fold.



After training and evaluating multiple models, the one that yields the highest performance (based on accuracy, precision, recall, and F1-score) on the test set can be considered the best model for this activity classification task.

- **Best Model:** Chosen the model with the highest overall metrics.
- **Model Tuning:** Further hyperparameter tuning and cross-validation can improve model performance.
- **Real-Time Application:** Depending on the desired use case (e.g., real-time classification), models like Random Forest and SVC are generally more suited for deployment due to their balance between performance and computation.

By comparing multiple models and their performance, I have ensured that the select most suitable model for recognizing human activities using sensor data.

## 1. Logistic Regression:

- **Simple and Fast:** Logistic Regression is computationally efficient and works well for binary and multi-class classification problems.
- **Interpretable:** The model provides interpretable coefficients, which can be useful for understanding the relationship between features and predicted classes.
- **Good for Linearly Separable Data:** It performs well when the decision boundaries between classes are linear.

### Performance Insights:

- If the data is mostly linearly separable (or approximately so), logistic regression might perform well with a fast-training time.

## 2. Linear SVC (Support Vector Classifier with Linear Kernel):

- **Efficient for Linearly Separable Data:** Linear SVC works well when the data is linearly separable or nearly so.
- **Good Generalization:** By maximizing the margin between classes, SVC tends to generalize well even in high-dimensional spaces.
- **Effective for High Dimensionality:** Linear SVC can handle many features well, making it suitable for complex datasets.

### Performance Insights:

- Linear SVC is a great choice if the data has a linear structure or if have a lot of features that are not inherently highly non-linear.

## 3. RBF SVC (Support Vector Classifier with Radial Basis Function Kernel):

- **Handles Non-Linear Boundaries:** The RBF kernel can map data into higher-dimensional spaces, making it effective for handling non-linear relationships between features.
- **Good Performance in Complex Datasets:** RBF SVC often performs better when the data is complex and not linearly separable.

- **Flexible Kernel:** The RBF kernel is highly flexible and can model complex decision boundaries.

### Performance Insights:




- The RBF kernel can yield higher accuracy than logistic regression or linear SVC for datasets with non-linear patterns. It is worth experimenting with different values of C and gamma to find the best configuration



## Conclusion

The primary goal of this project was to **classify human physical activities** using sensor data collected from a smartphone's **accelerometer and gyroscope**. The classification involved six activities:

- **Dynamic:** WALKING, WALKING UPSTAIRS, WALKING DOWNSTAIRS
- **Static:** SITTING, STANDING, LAYING

By applying various machine learning models, we aimed to determine which algorithm best distinguishes between these activities using extracted time and frequency domain features.

-  **Linear SVC** emerged as the best performer with the highest accuracy and excellent generalization (low overfitting).
-  **Logistic Regression** showed comparable performance, making it a strong baseline model due to its simplicity and speed.
-  **RBF SVC** also performed well, capturing nonlinear relationships in the data.

-  **Random Forest** and **Decision Tree** had perfect training accuracy but lower test accuracy, indicating signs of **overfitting**.
-  **KNN** showed decent performance but might struggle with larger datasets due to its **lazy learning** nature and computational cost at inference time.