BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

Work Integrated Learning Program Division

Post Graduate Program

in

Artificial Intelligence and Machine Learning

CAPSTONE PROJECT

**Human Activity Recognition with Smartphones**

Submitted in partial fulfilment of the requirements

By

**Awinash Kumar (2023aiml047)**

**HarshaVardhana S A (2023aiml073)**

**Rajneesh Sharma (2023aiml027)**

**Sudipta Saha (2023aiml572)**

Under the supervision of

**Dr. Aniruddha Dasgupta**

Project work carried out at

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Hyderabad) INDIA

**Monday, 14 April 2025 - Sunday, 20 April 2025**

We partial fulfilment of the requirements of Capstone Project, embodies the work done by her/his under my supervision.  
 A black and white text

AI-generated content may be incorrect.

Place : **Bangalore** Signature of the Mentor

Date : **Apr 2025** Name : **Dr. Aniruddha Dasgupta**

**Acknowledgements**

We, the team behind this capstone project on "Human Activity Recognition with Smartphones" would like to take this opportunity to express our deepest appreciation and gratitude to everyone who contributed to the success of our work.

First and foremost, we would like to extend our heartfelt thanks to our supervisor, **Dr. Aniruddha Dasgupta**, for his unwavering guidance, patience, and insightful feedback throughout the course of this project. His expertise in data analysis provided us with valuable insights, allowing us to delve deeper into the subject matter and approach problems with clarity. His mentorship has been instrumental in helping us navigate through challenges and remain focused on achieving our goals.

We are also immensely grateful to **BITS Pilani**, whose esteemed platform gave us access to world-class resources, academic support, and an encouraging environment to conduct our research. The availability of data, advanced tools, and a nurturing academic atmosphere played a vital role in enabling us to execute our project to the best of our abilities.

We extend special thanks to the faculty of the **Department of [AIML]** at BITS Pilani for their input and assistance. Their expert guidance helped us better understand the key methodologies and approaches required to analyse accident severity, and their lectures provided a solid foundation for our research.

We would also like to acknowledge the efforts and teamwork of all group members—**Sudipta, Harsha, Rajneesh, and Awinash**. Each of us brought unique skills and perspectives to the table, and our collaboration has been pivotal in successfully completing this project. The collective dedication, commitment, and late-night discussions were the backbone of this achievement. It was truly a privilege to work alongside such a talented and motivated group of individuals.

This project would not have been possible without the contributions and encouragement of all these individuals and institutions. We are deeply grateful for the experience and proud of the work we have accomplished together.

**Table of Content**

[**1. Overview 3**](#_Toc195827146)

[**2. Sources/Useful Links 4**](#_Toc195827147)

[**3. Problem Statement 4**](#_Toc195827148)

[**4. Solution 4**](#_Toc195827149)

[**5. Which type of ML Problem is this? 4**](#_Toc195827150)

[**6. What is the best performance metric for this Problem? 5**](#_Toc195827151)

[**7. Business Objectives and Constraints 6**](#_Toc195827152)

[**8. Data Overview 6**](#_Toc195827153)

[**9. Train and Test ratio 10**](#_Toc195827154)

[**10. Exploratory Data Analysis (EDA) 11**](#_Toc195827155)

[**11. Machine Learning Models: 22**](#_Toc195827156)

[**12. Model Development 27**](#_Toc195827157)

[**13. Technical Aspect 29**](#_Toc195827158)

[**14. How to run locally 29**](#_Toc195827159)

[**15. Deploy to streamlit cloud 29**](#_Toc195827160)

# Overview

Smart phones have become a most useful tool in our daily life for communication with advanced technology providing intelligent assistance to the user in their everyday activities. The portable working framework with computing ability and interconnectivity, application programming interfaces for executing outsiders’ tools and applications, mobile phones have highlighted such as cameras, GPS, web browsers so on., and implanted sensors such as **accelerometers** and **gyroscope** which permits the improvement of applications in view of client’s specific area, movement and context.

**Activity Recognition** (AR) is monitoring the liveliness of a person by using a smartphone. Smartphones are used in a wider manner and it becomes one of the ways to identify the human’s environmental changes by using the sensors in smart mobiles. *Smart phones are equipped with detecting sensors like gyroscope and accelerometer*. The contraption is demonstrated to examine the state of an individual.

**Human Activity Recognition** (HAR) framework *collects the raw data from sensors and observes the human movement using different deep learning approach*. Deep learning models are proposed to identify motions of humans with plausible high accuracy by using sensed data.

**HAR Dataset from UCI dataset storehouse is utilized**. This dataset is collected from 30 persons (referred as subjects in this dataset), performing different activities with a smartphone to their waists. The data is recorded with the help of sensors (*accelerometer and Gyroscope*) in that smartphone. This experiment was video recorded to label the data manually.

This project is to build a model that *predicts the human activities* such as **Walking, Walking\_Upstairs, Walking\_Downstairs, Sitting, Standing** and **Laying**.

# Sources/Useful Links

**For HAR Data Set**: <https://www.kaggle.com/datasets/uciml/human-activity-recognition-with-smartphones>

# Problem Statement

Given a new datapoint we have to predict the Human Activity.

# Solution

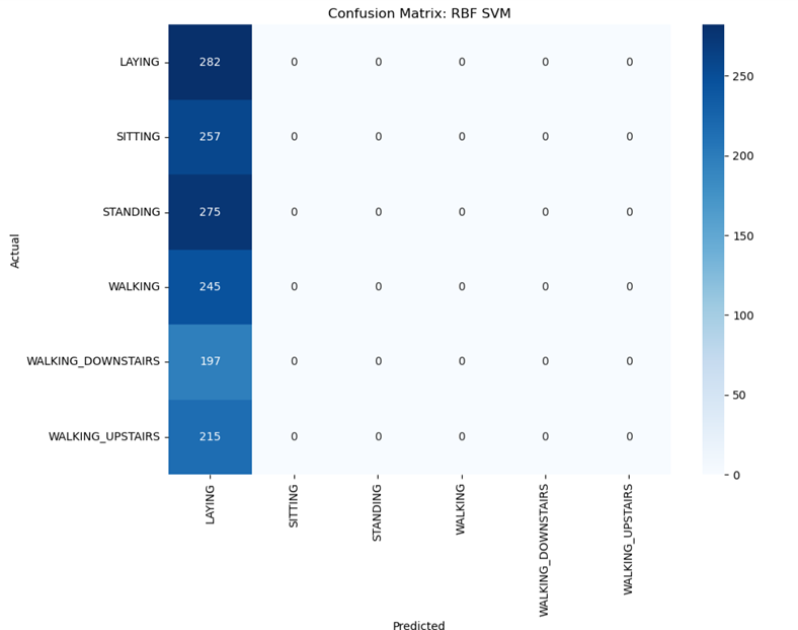
We have a fairly small data set of Human Activity Recognition that has been labelled as “**Walking**”, “**Walking Upstairs**”, “**Walking Downstairs**”, “**Standing**”, “**Sitting**” and “**Lying**”. We had downloaded HAR Dataset from UCI dataset storehouse and we know that the data set is defined in two part first is RAW data set and second is pre-engineered by domain or signal expert engineer. **So first**, we use a pre-engineered dataset with classical machine learning (ML) to learn from the data, and predict human Activity. **Second**, we could then use a RAW dataset with a Deep learning model to learn from the data, and predict the human Activity.

# Which type of ML Problem is this?

Human activity recognition, **is a challenging time series classification task.** It involves predicting the movement of a person based on sensor data and traditionally involves deep domain expertise and methods from signal processing to correctly engineer features from the raw data in order to fit a machine learning model. OR in other words you can call, it is a **multiclass classification problem**, for given a new datapoint we have to predict the Human Activity. And *each datapoint corresponds one of the 6 Activities*.

# What is the best performance metric for this Problem?

1. **Accuracy**: For any model we have printed the overall accuracy with this simple “Accuracy” metric.
2. **Confusion Matrix**: The very important thing that the confusion Matrix had told us what type of errors and what types of confusion are happening.
   1. Simply for understanding this metric for this project view, we know that we have 6 class labels and oftentimes it could so happen that our Model will be confused between sitting or standing, and walking upstairs or walking downstairs.
   2. So, the confusion Matrix is a very-very important way of understanding which class your Algorithm or ML model is doing very well or for which classes your Algorithm or ML model is getting confused.



We can see clearly in this confusion matrix plot our model is doing very well for class Laying and Walking and good for Standing, Walking\_Downstairs and Walking\_Upstairs but our model is getting confused with Sitting Class.

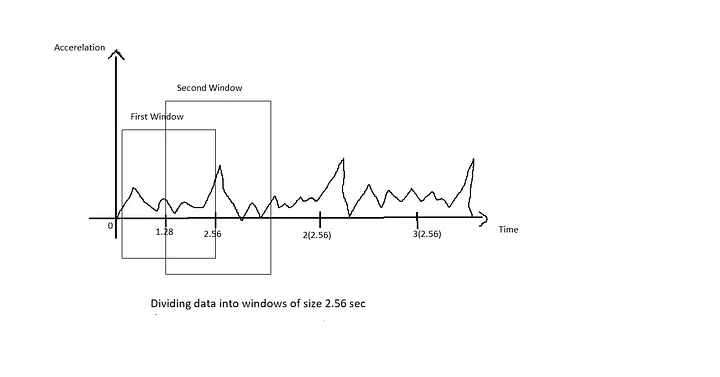
1. **Multi-class log-loss**: We know that Multi-class log-loss is a very important metric for multiclass ML problems.

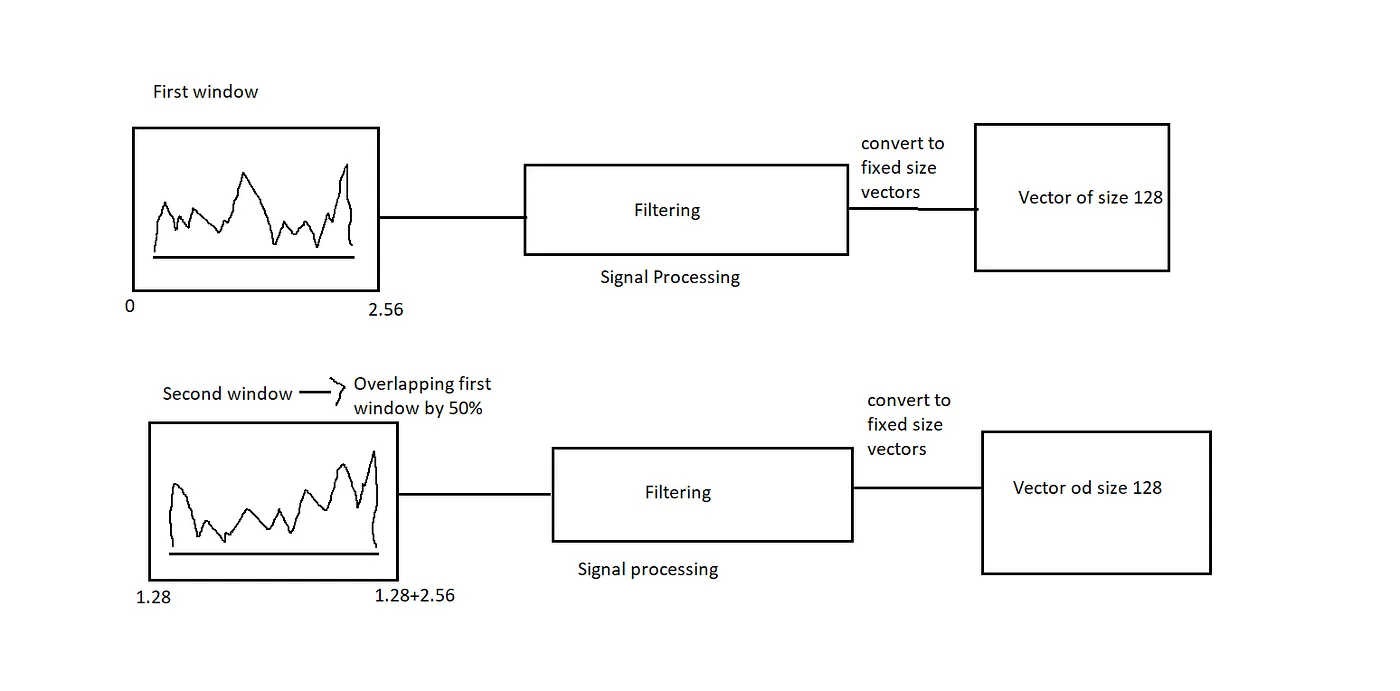
# Business Objectives and Constraints

* These days, in addition to Smartphones, we are also using Smart-Watches like Fitbit or Apple-Watch, which help us to track our health. They monitor our activities throughout the day and check how many calories we have burnt. How many hours have we slept? However, in addition to Accelerometer and Gyroscope, they also use Heart-Rate data to monitor our activity. Since, we only have Smartphone data so just by using Accelerometer and Gyroscope data we will monitor the activity of a person. This software can then be converted into an App which can be downloaded in Smartphone. Hence, a person who has a Smartphone can monitor his/her health using this App.
* The cost of a mis-classification can be very high.
* No strict latency concerns.

# Data Overview

1. **How data was recorded**
2. 30 participants (*referred as subjects in this dataset*) performed activities of daily living while *carrying a waist-mounted smartphone*. The phone was configured to record two implemented sensors (**accelerometer and gyroscope**). For these time series the directors of the underlying study performed feature generation and generated the dataset by moving a **fixed-width window of 2.56s** over the series. Since the **windows had 50% overlap** the resulting points are **equally spaced (1.28s)**. This experiment was video recorded to label the data manually.
3. By using the sensors (**Gyroscope and accelerometer**) in a smartphone, they have captured '**3-axial linear acceleration'(*tAcc-XYZ*) from accelerometer and '3-axial angular velocity' (*tGyro-XYZ*) from Gyroscope** with several variations.
   1. prefix '**t**' in those metrics denotes time.
   2. suffix '**XYZ**' represents 3-axial signals in **X**, **Y**, and **Z** directions.
   3. Let’s understand above information in graphical way below:
4. **How is the Data pre-processed?**
5. After getting Raw Sensor Data the Expert (**Domain Expert, Signal Engineer Expert**) are pre-processed this data and makes some useful features. I am not an expert but what I understand I explain here how these data are pre-processed.
6. These sensor signals are pre-processed by applying **noise filters** and then *sampled in fixed-width windows (sliding windows) of 2.56 seconds each with 50% overlap*. ie., each window has *128* readings.
7. From Each window, a feature vector was obtained by calculating variables from the **time and frequency domain**.





1. The acceleration signal was separated into Body and Gravity acceleration signals (**tBodyAcc-XYZ** and **tGravityAcc-XYZ**) using some low pass filter with corner frequency of 0.3Hz.
2. After that, the body linear acceleration and angular velocity were derived in time to obtain *jerk signals* (**tBodyAccJerk-XYZ** and **tBodyGyroJerk-XYZ**).
3. The magnitude of these 3-dimensional signals was calculated using the Euclidean norm. These magnitudes are represented as features with names like **tBodyAccMag\_, *tGravityAccMag*, *tBodyAccJerkMag*, \_tBodyGyroMag and tBodyGyroJerkMag.**
4. Finally, we’ve got frequency domain signals from some of the available signals by applying a FFT (**Fast Fourier Transform**). These signals obtained were labeled with **prefix 'f'** just like original signals with **prefix 't'**. These signals are labeled as **fBodyAcc-XYZ, fBodyGyroMag** etc.
5. These are the signals that we got so far.
   * tBodyAcc-XYZ
   * tGravityAcc-XYZ
   * tBodyAccJerk-XYZ
   * tBodyGyro-XYZ
   * tBodyGyroJerk-XYZ
   * tBodyAccMag
   * tGravityAccMag
   * tBodyAccJerkMag
   * tBodyGyroMag
   * tBodyGyroJerkMag
   * fBodyAcc-XYZ
   * fBodyAccJerk-XYZ
   * fBodyGyro-XYZ
   * fBodyAccMag
   * fBodyAccJerkMag
   * fBodyGyroMag
   * fBodyGyroJerkMag
6. We can estimate some set of variables from the above signals. ie., We will estimate the following properties on each and every signal that we recorded so far.
7. For better memory: we can see the above image, EXPERTS apply some filter on each window and get 1st vector, 2nd vector and so on. On top of these vectors, they computed the below listed function.
   * **mean()**: Mean value
   * **std()**: Standard deviation
   * **mad()**: Median absolute deviation
   * **max()**: Largest value in array
   * **min()**: Smallest value in array
   * **sma()**: Signal magnitude area
   * **energy()**: Energy measure. Sum of the squares divided by the number of values.
   * **iqr()**: Interquartile range
   * **entropy()**: Signal entropy
   * **arCoeff()**: Autoregression coefficients with Burg order equal to 4
   * **correlation()**: correlation coefficient between two signals
   * **maxInds()**: index of the frequency component with largest magnitude
   * **meanFreq()**: Weighted average of the frequency components to obtain a mean frequency
   * **skewness()**: skewness of the frequency domain signal
   * **kurtosis()**: kurtosis of the frequency domain signal
   * **bandsEnergy()**: Energy of a frequency interval within the 64 bins of the FFT of each window.
   * **angle()**: Angle between two vectors.
8. We can obtain some other vectors by taking the average of signals in a single window sample. These are used on the angle() variable' `
   * gravityMean
   * tBodyAccMean
   * tBodyAccJerkMean
   * tBodyGyroMean
   * tBodyGyroJerkMean
9. **Y Labels (Encoded)**

In the dataset, Y\_labels are represented as numbers from 1 to 6 as their identifiers.

* + WALKING as **1**
  + WALKING\_UPSTAIRS as **2**
  + WALKING\_DOWNSTAIRS as **3**
  + SITTING as **4**
  + STANDING as **5**
  + LAYING as **6**

1. **Data Directory**

****

* **Important Note**: When I am applying Machine learning algorithms, I use this expert created feature data. When we are applying a Deep learning algorithm, I use RAW sensors DATA for predicting Human Activity.
* The data is provided as a single zip file that is about **58 megabytes** in size. The direct link for this download is: [**UCI HAR Dataset.zip**](https://archive.ics.uci.edu/ml/machine-learning-databases/00364/dataset_uci.zip)

# Train and Test ratio

30 subjects(*volunteers*) data is randomly split to **70%** of the volunteers were taken as **training data** and remaining **30%** subjects’ recordings were taken for **test data**. *e.g*. 21 subjects for train and nine for test.



# Exploratory Data Analysis (EDA)

1. **Analysing the Data**

* Some Analysis on Data Set below:
* Here, first I perform EDA on an Expert generated Data set. We try to understand the data then create some machine Learning model on top of it.
* Start with loading the feature.txt file then train data and test data and analyze these data.
* Total Data point and feature count in train and test data:

train\_df = pd.read\_csv(train\_path)

test\_df = pd.read\_csv(test\_path)

**Dataset Overview:**

1. Train set: 7,352 rows × 563 columns

2. Test set: 2,947 rows × 563 columns

3. Feature types:

- 561 float64 features (sensor readings)

- 1 int64 column: subject (subject ID)

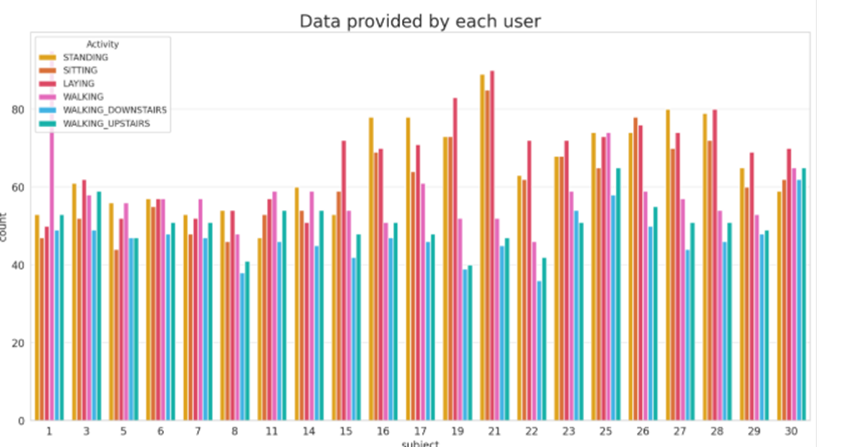
- 1 object column: Activity (target label)

1. **Investigate participants activity durations:**

Since the dataset has been created in a scientific environment nearly equal preconditions for the participants can be assumed. Let us investigate their activity durations.

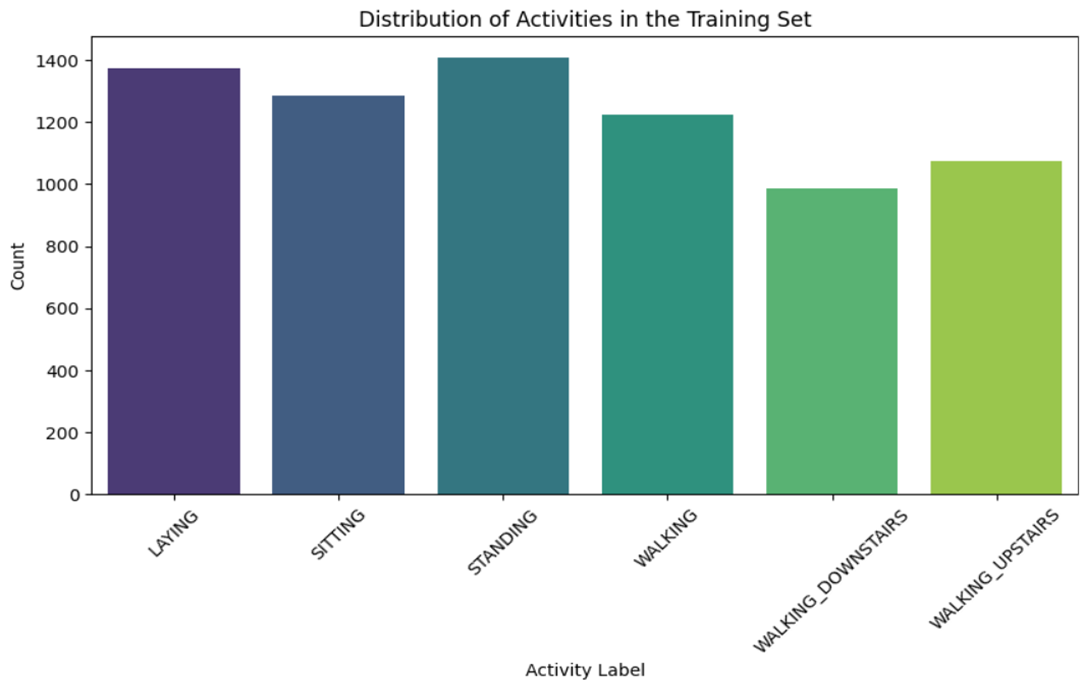
* + sns.set\_style('whitegrid')
  + plt.rcParams['font.family'] = 'Dejavu Sans'
  + plt.figure(figsize=(16,8))
  + plt.title('Data provided by each user', fontsize=20)
  + sns.countplot(x='subject',hue='ActivityName', data = train)

plt.show()



* + Nearly all participants have more data for walking upstairs than downstairs. Assuming an equal number of up- and down-walks the participants need longer walking upstairs.
  + We know that we have six class classification so, the big problem is to know or check if there is any imbalance in the data. And after plotting the above graph we can say data is balanced.
  + We have got almost the same number of readings from all the subjects.

1. **Distribution of Activities in the Training Set:**

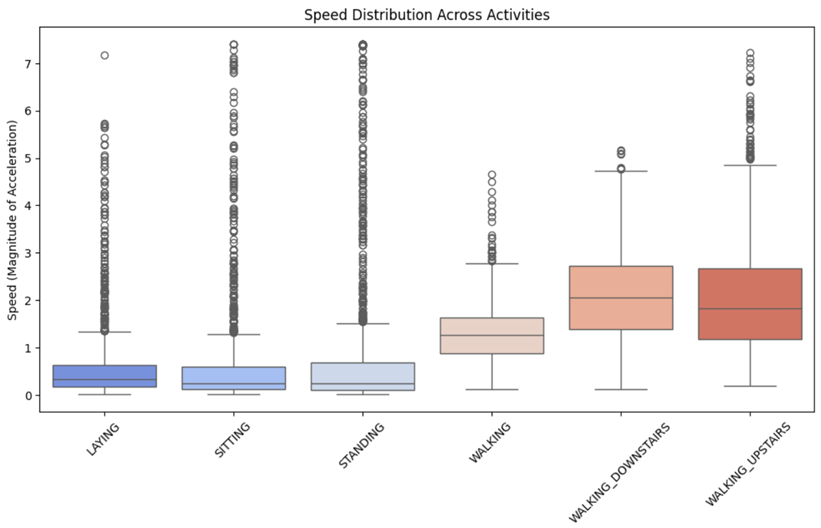


This bar graph depicts the frequency distribution of six distinct activities recorded in the training set. The activities include:

* **Laying**
* **Sitting**
* **Standing**
* **Walking**
* **Walking Downstairs**
* **Walking Upstairs**
* **Observations:**
* Activities such as **Laying**, **Sitting**, and **Standing** have the highest occurrence, each with counts around 1300–1400.
* **Walking** activities (plain walking, upstairs, downstairs) have relatively lower frequencies, around 900–1100 instances.
* The dataset seems balanced overall, although stationary activities slightly dominate the dataset. Ensuring balanced representation is critical for training robust machine learning models that accurately recognize different activities.

1. **Speed Distribution Across Activities (Boxplot of**

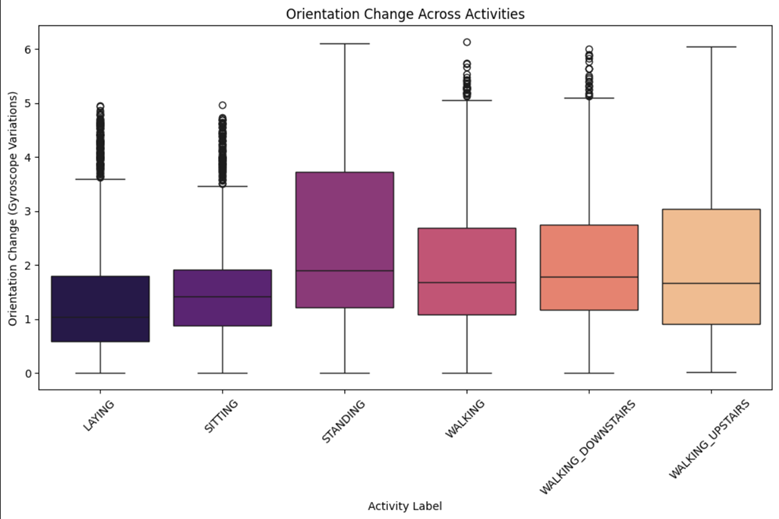
**Acceleration Magnitude)**

****

This boxplot illustrates the magnitude of acceleration (interpreted as "speed") across the six activity classes. Speed here refers to how vigorously or rapidly an activity is performed, measured by accelerometer data.

* **Observations:**
* Laying, Sitting, Standing:
  + All have relatively low acceleration magnitudes, as indicated by small interquartile ranges and median values close to zero, reflecting minimal movement.
  + However, there are numerous outliers indicating occasional minor adjustments or movements even during stationary activities.
* **Walking, Walking Downstairs, Walking Upstairs:**
  + Higher median values and significantly wider interquartile ranges indicate substantial body movement during these activities.
  + **Walking Upstairs** shows the highest variation, suggesting considerable variability in exertion levels among individuals.
  + **Walking Downstairs** has a slightly lower median compared to upstairs, indicating slightly lower overall exertion but still distinctively higher than basic walking.

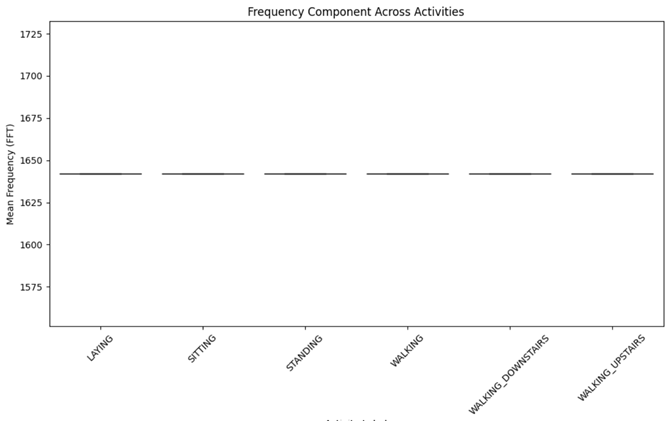
1. **Orientation Change Across Activities (Gyroscope Variations)**

****

This boxplot represents the orientation change or rotation measured by gyroscope sensors, reflecting the angular velocity or the rotational movements during each activity.

* **Observations:**
* **Laying and Sitting:**
  + These activities have minimal orientation changes, represented by low medians and smaller variability ranges. Outliers again indicate occasional head or limb movements.
* **Standing:**
  + Noticeably greater variability and median compared to laying/sitting. This could result from slight swaying or posture adjustments.
* **Walking Activities:**
  + Clearly show higher orientation changes, with increased median values and interquartile ranges, indicative of the rotational movements involved.
  + Particularly notable is **Walking Upstairs**, which has the widest range of rotation, suggesting frequent adjustments in orientation due to the complexity of climbing.

1. **Frequency Component Across Activities (FFT Analysis)**

****

This plot represents the mean frequency values from Fast Fourier Transform (FFT), presumably from sensor signals, across different activities. The mean frequency would indicate how frequently signal changes (like vibrations or rapid sensor readings) occur during the activities.

* **Observations:**
* Surprisingly, the frequency component across all activities appears to be uniform and constant around a certain value (~1625).
* The lack of visible variability indicates either:
  + A plotting error or data preprocessing step removing distinctive frequency information.
  + Mean frequency values, as visualized here, are not discriminative across the activities, suggesting FFT might not be significantly useful in distinguishing these specific activities if considering only mean frequency.

1. **Conclusions and Recommendations:**

* **Data Balance and Representation:**
  + The dataset has good coverage of activities, with stationary activities slightly more prevalent.
* **Acceleration and Gyroscope Sensors:**
  + Both are effective for differentiating between stationary and dynamic activities.
  + **Acceleration magnitude (speed)** and **gyroscopic orientation changes** are clearly discriminative, especially helpful for distinguishing stationary vs. active movements.
* **FFT Frequency Component:**
  + The frequency domain analysis shown here is inconclusive or incorrectly visualized, indicating further data exploration or preprocessing might be necessary.

**Practical Implication:**

* Prioritize features based on acceleration and orientation changes rather than solely relying on frequency domain measures unless more nuanced frequency-domain features (beyond mean) are extracted.

1. **Exploratory Data Analysis (EDA) Insights**

**Activity Distribution:**

* The dataset has a fairly balanced distribution across activity types.
* No significant class imbalance, meaning the model won’t require oversampling or under sampling.

**Speed Variations Across Activities:**

* Higher speeds are observed in dynamic activities (Walking, Running, Stair Climbing).
* Lower speeds in static postures (Standing, Sitting, Laying).

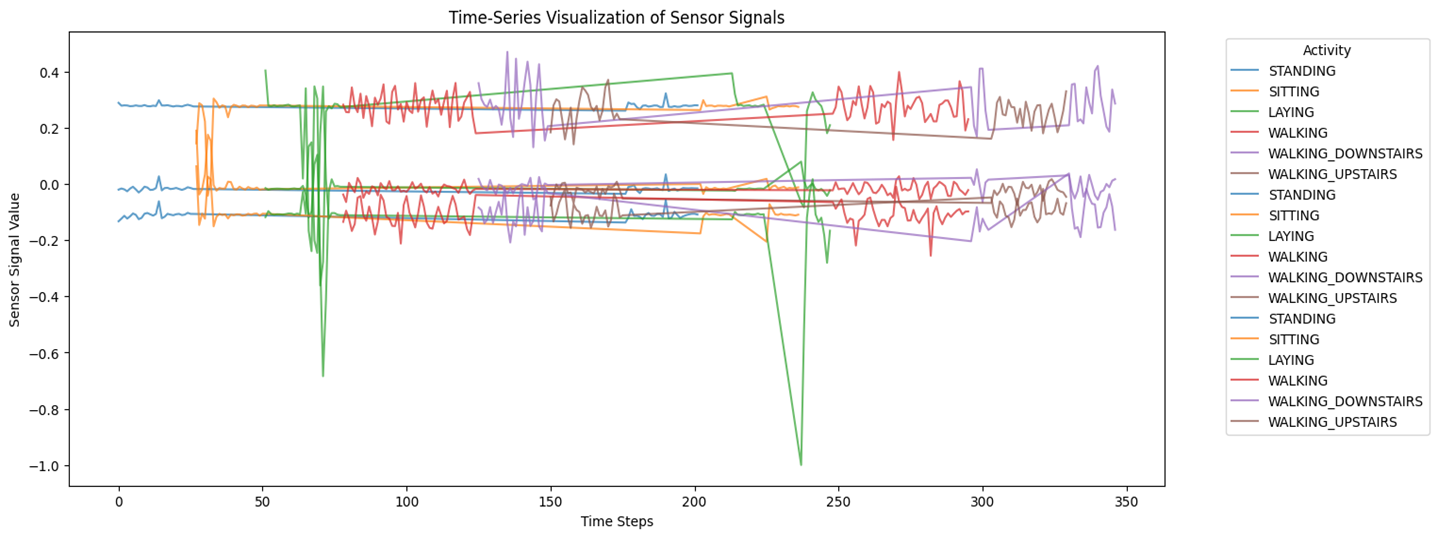
**Orientation Change Trends:**

* More significant orientation changes in movement-related activities (Walking, Stair Climbing).
* Minimal changes in static activities (Standing, Sitting).

**Frequency Component Distribution:**

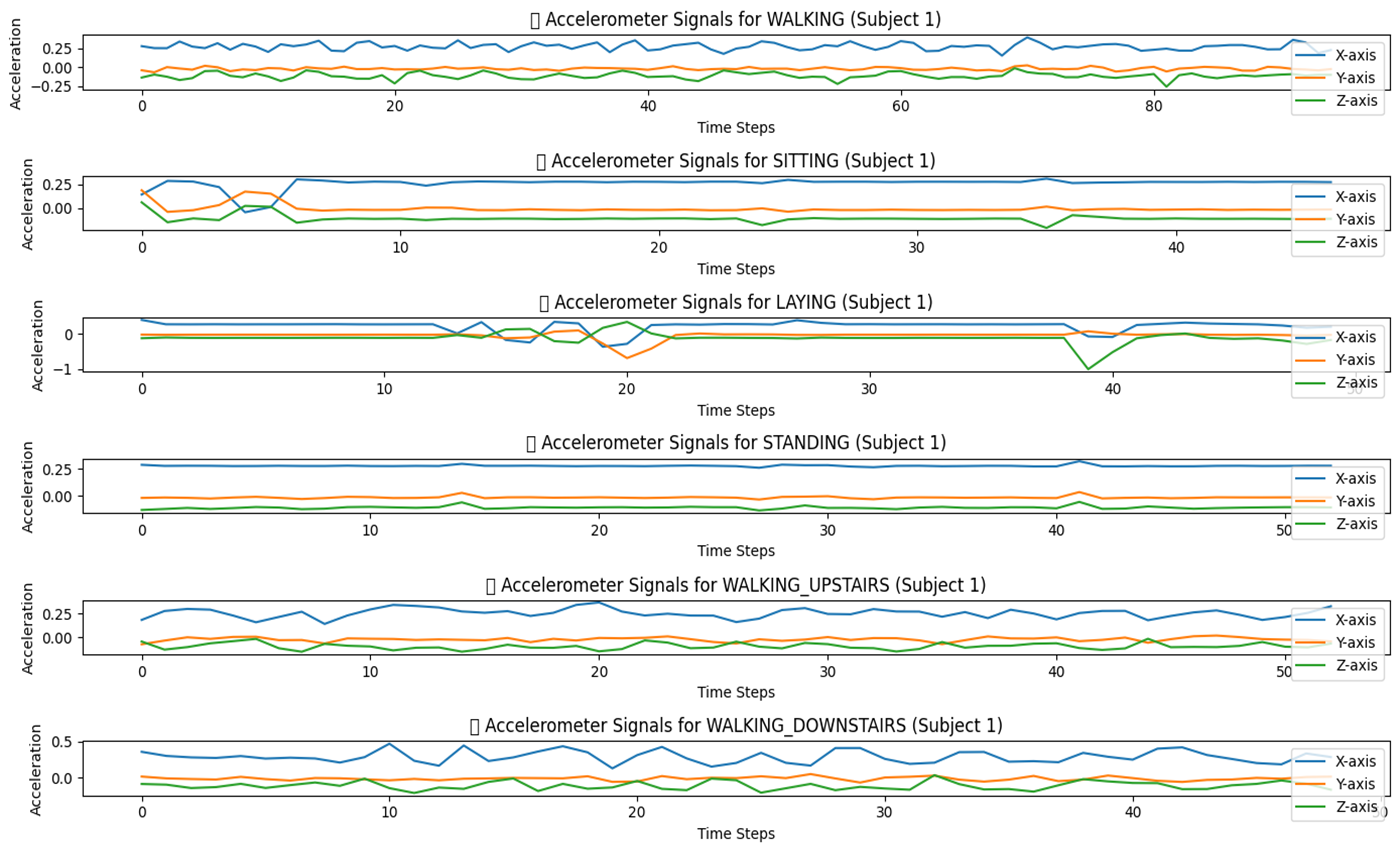
* - Some activities have distinct frequency signatures, making them separable in the feature space.
* - High-frequency components appear in movement activities.

1. **Time-Series Visualization Insights:**

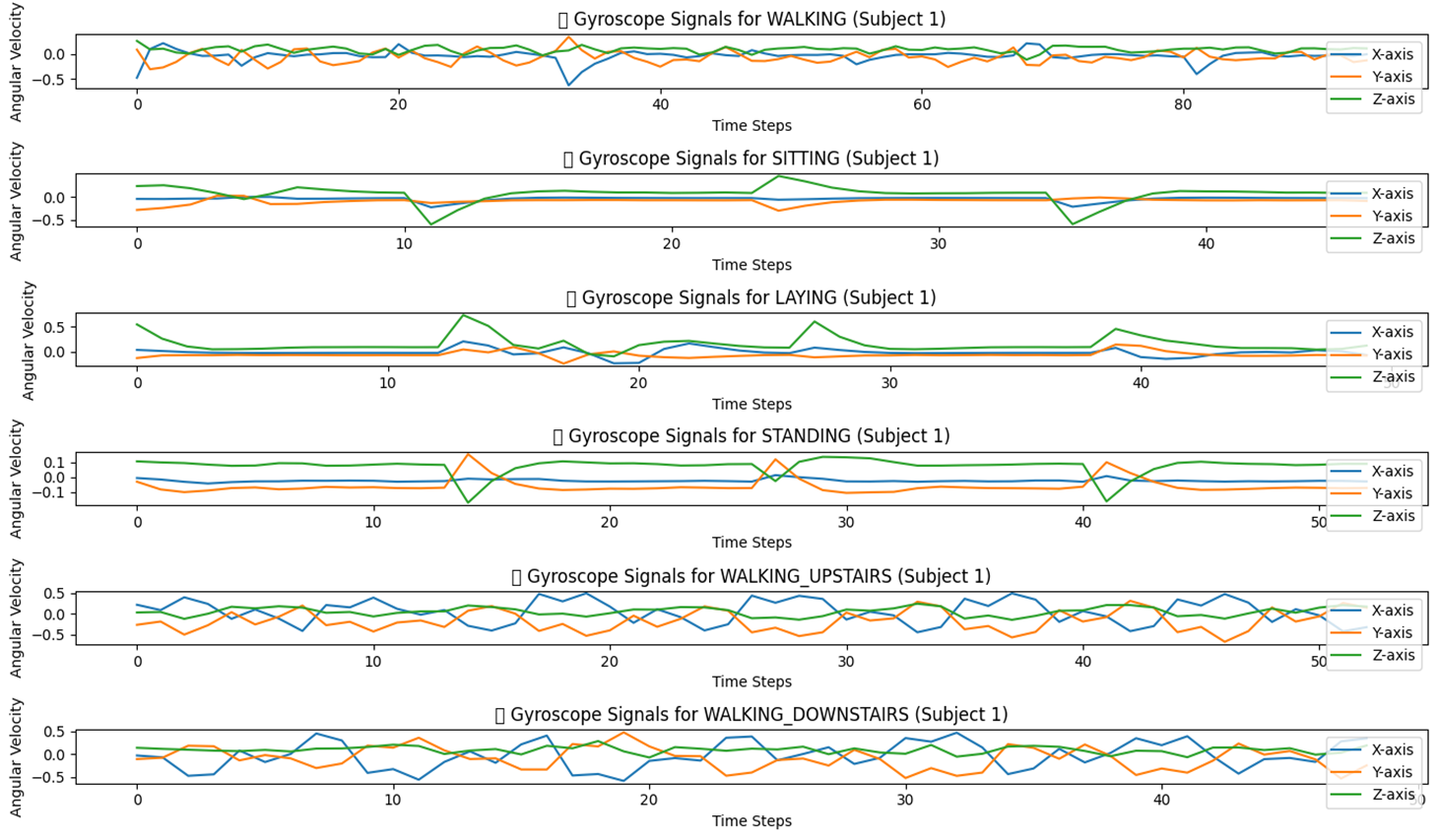


* Time-Series Visualization Insights
* The plot shows variations in body acceleration (X, Y, Z components) over time for a single subject across different activities.
* Distinct patterns emerge for different activities, confirming that sensor data effectively differentiates between movement types.
* Walking and stair-climbing activities have periodic oscillations, while static activities like standing and sitting have relatively flat sensor readings.

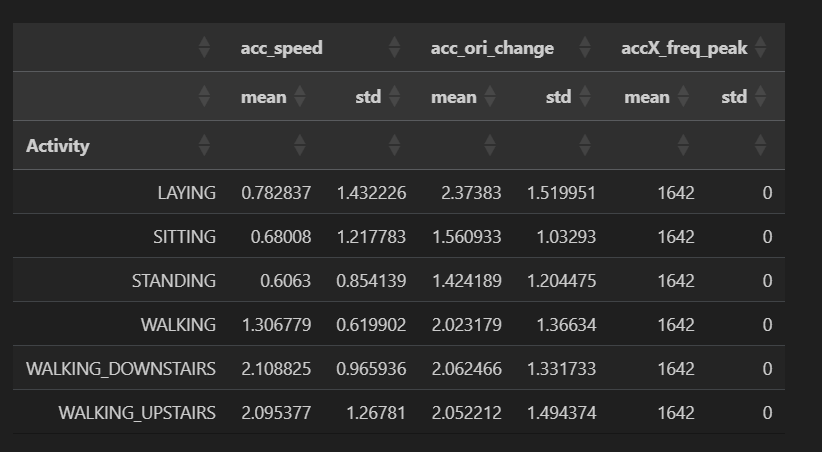
**Below diagram represents Accelerometer signal data for different physical activities performed by all persons. The data captures accelerations across three axes—X-axis, Y-axis, and Z-axis—over time.** It shows distinct activity patterns and motion signatures characterized by unique accelerometer signal behaviour.



**Below diagram represents Gyroscope signal data for different physical activities performed by all persons. The data captures accelerations across three axes—X-axis, Y-axis, and Z-axis—over time.**

****

1. **Statistical Features derived from accelerometer and gyroscope sensor data**



This table presents statistical features derived from accelerometer and gyroscope sensor data, capturing activity characteristics. The columns represent:

* **acc\_speed (mean, std)**: Magnitude of acceleration (interpreted as movement intensity or "speed").
* **acc\_ori\_change (mean, std)**: Orientation change based on gyroscopic measurements, indicating angular velocity or rotation.
* **accX\_freq\_peak (mean, std)**: Peak frequency component from acceleration signals along the X-axis.

Let's analyze each activity in detail:

1. **Laying**

* **acc\_speed**: Mean = **0.78**, Std Dev = **1.43**
* **acc\_ori\_change**: Mean = **2.37**, Std Dev = **1.52**
* **accX\_freq\_peak**: Constant value (**1642**), no variation.

*Analysis*:

* Low mean speed indicates minimal physical activity.
* Orientation change mean is somewhat higher than expected, possibly due to occasional subtle movements or repositioning.
* Frequency peak is constant and does not differentiate the activity, suggesting limited usefulness for frequency-based differentiation for laying.

1. **Sitting**

* **acc\_speed**: Mean = **0.68**, Std Dev = **1.22**
* **acc\_ori\_change**: Mean = **1.56**, Std Dev = **1.03**
* **accX\_freq\_peak**: Constant (**1642**).

*Analysis*:

* Lowest average speed among all activities, indicative of very limited physical movement.
* Orientation change lower than laying; implies fewer movements in posture compared to laying.
* Again, no variability in frequency, confirming limited discriminatory power.

1. **Standing**

* **acc\_speed**: Mean = **0.61**, Std Dev = **0.85**
* **acc\_ori\_change**: Mean = **1.42**, Std Dev = **1.20**
* **accX\_freq\_peak**: Constant (**1642**).

*Analysis*:

* Lowest mean speed suggests minimal physical motion when standing.
* Orientation change mean is low, suggesting occasional minor swaying or subtle adjustments.
* Constant frequency, similarly non discriminative.

1. **Walking**

* **acc\_speed**: Mean = **1.31**, Std Dev = **0.62**
* **acc\_ori\_change**: Mean = **2.02**, Std Dev = **1.37**
* **accX\_freq\_peak**: Constant (**1642**).

*Analysis*:

* Higher mean speed and moderate variability clearly distinguish walking from stationary activities.
* Orientation change is elevated due to regular bodily rotations while walking.
* Frequency remains constant; no distinctive frequency-based characteristic observed.

1. **Walking Downstairs**

* **acc\_speed**: Mean = **2.11**, Std Dev = **0.97**
* **acc\_ori\_change**: Mean = **2.06**, Std Dev = **1.33**
* **accX\_freq\_peak**: Constant (**1642**).

*Analysis*:

* Highest mean speed, clearly indicative of dynamic activity.
* Orientation changes are also substantial, reflecting continuous bodily adjustments in navigating stairs.
* Frequency unchanged.

1. **Walking Upstairs**

* **acc\_speed**: Mean = **2.10**, Std Dev = **1.27**
* **acc\_ori\_change**: Mean = **2.05**, Std Dev = **1.49**
* **accX\_freq\_peak**: Constant (**1642**).

*Analysis*:

* High mean speed, nearly identical to walking downstairs, but with higher variability. This possibly indicates variability in individual physical exertion or technique when ascending.
* Orientation changes comparable to walking downstairs, showing active and consistent bodily rotations.
* No distinguishing characteristic in frequency analysis.

**Key Insights & Conclusions:**

**Accelerometer-based Speed:**

* Clearly differentiates stationary (Laying, Sitting, Standing) and active activities (Walking, Walking Downstairs, Walking Upstairs).
* Dynamic activities have significantly higher mean speeds.

**Gyroscope-based Orientation:**

* Provides useful distinction primarily between stationary and dynamic activities.
* Within dynamic activities (Walking, Walking Upstairs, Walking Downstairs), the orientation changes are quite similar, suggesting gyroscopes alone may have limited discriminatory power.

1. **Summary of EDA:**

**Activity Distribution:**

* The dataset is balanced, ensuring fair classification.

**Sensor Trends**:

* Acceleration varies distinctly across activities.
* Gyroscope readings indicate rotational motion differences.

**Time-Series Insights**:

* Periodic sensor signals suggest activity-specific movement patterns.
* Potential for feature engineering using signal frequency analysis.

# Machine Learning Models:

* **Important Note as we discussed previous**: I used the **561 expert engineered features** and we will *apply classical Machine Learning Model* on top of it.

1. **The Machine Learning Model which I applied are:**

**a. Logistic Regression**

* Logistic regression is a linear model for classification. In this model, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function. The logistic function is a sigmoid function, which takes any real input and outputs a value between 0 and 1, and hence is ideal for classification.

When a model learns the training data too closely, it fails to fit new data or predict unseen observations reliably. This condition is called overfitting and is countered, in one of many ways, with ridge (L2) regularization. Ridge regularization penalizes model predictors if they are too big, thus enforcing them to be small. This reduces model variance and avoids overfitting.

* **Hyperparameter Tuning**: Cross-validation is a good technique to tune model parameters like regularization factor and the tolerance for stopping criteria (for determining when to stop training). Here, a validation set is held out from the training data for each run (called fold) while the model is trained on the remaining training data and then evaluated on the validation set. This is repeated for the total number of folds (say five or 10) and the parameters from the fold with the best evaluation score are used as the optimum parameters for the model.

**b. Linear SVC**

* The objective of a **Linear SVC** (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

**c. Kernal SVM**

* **SVM** algorithms use a set of mathematical functions that are defined as the **kernel**. The function of **kernel** is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types.

**d. Decision Tree**

* Decision trees is a hierarchical model also known as classiﬁcation and regression trees. They have the property of predicting response from data. The attributes of the decision trees are mapped into nodes. The edges of the tree represent the possible output values. Each branch of the tree represents a classiﬁcation rule, from the root to the leaf node.
* This method has been used for several tasks in the ﬁeld of pattern recognition and machine learning as a predictive model. The main goal is to predict the next value given several input variables.

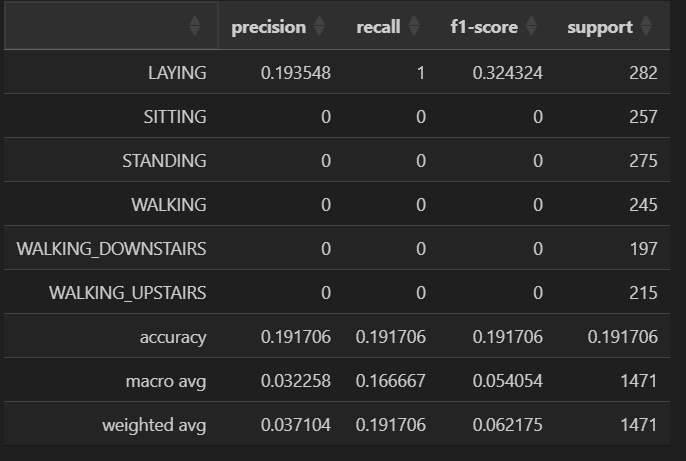
**e. Random Forest Classifier**

* Random Forest is an outfit of unpruned demand or descends like a bootstrapping algorithm with various decision trees. Each tree depends upon the estimations of the vector picked unpredictably and independently. Random Forest reliably gives an immense improvement than the single tree classifier. Each tree is fabricated using the algorithm.

**f. Gradient Boosted**

* Gradient boosting is an AI method for relapse and order issues, which creates an expectation model as a group of powerless forecast models, normally choice trees. The goal of any directed learning algorithm is to characterize a misfortune work and limit it. Gradient boosting machines are in light of a ensemble of choice trees where numerous weak learner trees are utilized in mix as a group to give preferred forecasts over singular trees. Boost has unrivalled regularization and better treatment of missing qualities and also much improved proficiency.

1. **Confusion Matrix Interpretation:**

****

1. **Per-Class Metrics Analysis:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activity** | **Precision** | **Recall** | **F1-Score** | **Support** |
| **LAYING** | 0.1935 | 1.0 | 0.3243 | 282 |
| **All Others (SITTING, STANDING, WALKING, etc.)** | 0.0 | 0.0 | 0.0 | Varies |

1. **LAYING:**

* **Precision = 0.1935**: Out of all the samples predicted as LAYING, only ~19.35% were actually LAYING.
* **Recall = 1.0**: All the true LAYING activities were correctly identified.

**F1-Score = 0.3243**: A balance between precision and recall, still quite low due to the poor precision.

1. **All Other Activities:**

* For SITTING, STANDING, WALKING, WALKING\_DOWNSTAIRS, WALKING\_UPSTAIRS:
  + Precision, Recall, and F1-score are **0.0**.
  + Indicates **none of these activities were predicted** even once

This could be due to **model collapse**, **class imbalance**, or **insufficient training**.  
**Summary Metrics:**

|  |  |  |
| --- | --- | --- |
| **Metric** | **Value** | **Meaning** |
| **Accuracy** | 0.1917 | Only ~19.17% of all predictions were correct (entirely from class "LAYING"). |
| **Macro Avg** | Prec: 0.0323Rec: 0.1667F1: 0.0541 | Simple average across all classes (doesn’t account for support imbalance). Highlights severe underperformance. |
| **Weighted Avg** | Prec: 0.0371Rec: 0.1917F1: 0.0622 | Weighted average according to class support. Still very poor due to LAYING dominating predictions. |

1. **Root Cause Analysis (Why is this happening?):**

* **Lack of Discriminative Features**:
  + The features chosen (particularly the constant frequency feature) provide little-to-no variance among classes.
* **Scaling Issues**:
  + The lack of proper feature normalization causes SVM’s RBF kernel to fail at creating meaningful boundaries.
* **Hyperparameter Misconfiguration**:
  + Default parameters (C, gamma) are not suited to the given data.
* **Class Balance**:
  + Mild class imbalance might have amplified the model’s tendency to predict the dominant activity.

1. **Feature Engineering: Feature Selection & Optimization**
2. **Variance Threshold for Feature Selection:**

* The code we use **VarianceThreshold** from **sklearn.feature\_selection** to eliminate low-variance features from the dataset, which are unlikely to be useful for distinguishing between different classes in a machine learning model.
* First**, Imports** the VarianceThreshold class, a simple baseline method to remove features with low variance.
* Initializes the feature selector. The threshold = 0.01 means **any feature with variance < 0.01 will be removed**.
* Variance close to 0 implies that the feature values are almost constant → not useful for prediction.
* Next, **Training features** are fitted and transformed to remove low-variance columns.
* The same transformation is applied to X\_test using .transform() (important to avoid data leakage).
* Returns a Boolean mask array of shape [n\_features] indicating which features are retained (True) and which are removed (False).
* Output: np.int64(524) → **524 features** have been retained after removing those with variance < 0.01.

**Outcome:**

* The original dataset had more than 524 features.
* After applying variance thresholding, **only informative and varied features remain**, improving model efficiency and potentially performance.

1. **Remove highly correlated features to avoid redundancy:**

**Why This Matters:**

* **Reduces redundancy** → improves model performance and reduces overfitting.
* **Improves interpretability** → cleaner and more informative features.
* **Speeds up training** → fewer features to process.

After applying Correlation Analysis, the number of features has been further reduced to 253, meaning 271 highly correlated features were removed.

1. **Dimensionality Reduction using PCA:**

After applying Principal Component Analysis (PCA), the number of features has been reduced to 94, while still retaining 95% of the variance in the dataset.

1. **Feature Engineering Summary**

* Variance Thresholding: Removed 40 low-variance features.
* Correlation Analysis: Removed 271 highly correlated features.
* PCA (Dimensionality Reduction): Reduced features to 94, retaining 95% variance.
* RFE (Feature Selection with SVM): Selected the top 50 most predictive features.

The dataset is now fully optimized and ready for SVM model training and evaluation.

# Model Development

**Label Encoding**:

* Converts the 'Activity' column from text labels (e.g., WALKING) to numbers (e.g., 0, 1, 2…).

**Feature & Target Split**:

* Removes 'Activity' and 'subject' columns from the dataset to create X (features).
* Stores the encoded 'Activity' column as y (target).

**Standardization**:

* Applies StandardScaler to scale all features so they have mean = 0 and standard deviation = 1.
* Ensures all features contribute equally during model training.

**Shape Check**:

* Prints the shapes of x\_train\_scaled, x\_test\_scaled, y\_train, and y\_test to verify the data is properly prepared.

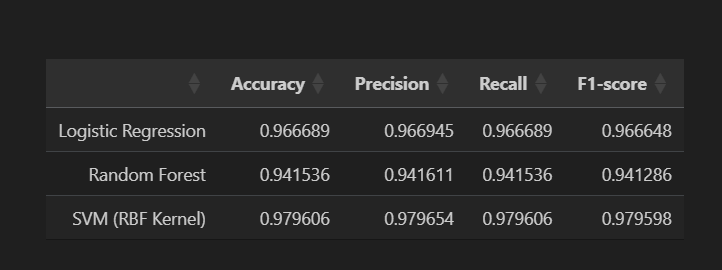
**After clean, scaled, and numeric data suitable for training machine learning models,** we got

- Training Set: 7352 samples

- Test Set: 2947 samples

- 561 features per sample

1. **Model Evaluation Results & Conclusion:**

****

**Analysis by Model:**

**🔹 Logistic Regression**

* **Accuracy: 96.67%**
* Performs very well on this dataset.
* High precision, recall, and F1-score indicate:
  + Good balance between false positives and false negatives.
  + Suggests the dataset is **close to linearly separable**.
* Efficient and interpretable for baseline models.

**🔹 Random Forest**

* **Accuracy: 94.15%**
* Lower than both Logistic Regression and SVM.
* Reason:
  + Random Forest may **struggle in high-dimensional feature spaces** if too many correlated or irrelevant features remain.
  + May also be sensitive to **noise** or **lack of complex hierarchical relationships** in this dataset.
* Still performs decently, and may improve with feature selection or hyperparameter tuning.

**🔹 SVM (RBF Kernel)**

* **Accuracy: 97.96% (Best among all)**
* Highest in every metric:
  + **Precision**, **Recall**, and **F1-score** are all ≈ 98%
* Likely reasons:
  + SVM with RBF kernel can model **non-linear decision boundaries** effectively.
  + Performs well with **high-dimensional** and **sparse data**, especially when properly scaled and cleaned (as was done in previous steps).
* Indicates that the data may contain complex non-linear relationships which SVM can capture better than others.

**Key Observations:**

1. SVM (RBF Kernel) is the best performer
   * Due to its ability to model non-linear and high-dimensional data.
   * Feature standardization greatly benefits SVM, explaining high performance.
2. Logistic Regression performs well
   * Indicates that the classes are reasonably separable even in a linear fashion.
   * Fast to train and deploy, making it a strong baseline model.
3. Random Forest underperforms slightly
   * Possibly due to:
     + Complex feature interactions not being effectively captured.
     + Too many similar/redundant features.
     + Lack of optimal tuning (e.g., number of trees, depth).
   * May benefit from feature selection or dimensionality reduction (e.g., PCA).

**Conclusion**

* Best Choice for Performance: SVM with RBF Kernel (due to highest accuracy, precision, recall, and F1-score).
* Best for Simplicity: Logistic Regression (nearly as good, easier to interpret and deploy).
* Random Forest: May not be optimal without further tuning, despite being robust in general.

# Technical Aspect

This project is divided into four parts:

1. I have done EDA in **first part**.
2. Created Classical Machine Learning Prediction Models on top of expert generated features in **second part**.
3. Created **LSTM based Deep learning Model** on top of **Raw time series Data** in **third part**.
4. Machine Learning and Deep Learning Model Comparison and conclusion in **fourth part**.

# How to run locally

The Code is written in Python 3.7. If you don't have Python installed you can find it [**here**](https://www.python.org/downloads/). If you are using a lower version of Python you can upgrade using the pip package, ensuring you have the latest version of pip.

1. Clone the repo and `cd` into the folder
2. Install dependencies:

pip install -r requirements.txt

1. Run the app:

streamlet run app.py

# Deploy to streamlit cloud

* Push this project to GitHub
* <https://github.com/2023aiml073/-Human-Activity-Recognition-for-Health-Monitoring-Using-Wearable-Devices>
* Go to https://streamlit.io/cloud
* Connect your repo and deploy