HUMAN ACTIVITY RECOGNITION

V.G.S. Sai Kiran Reddy	Praveen Kumar. G	Sanjay Chandra. K
S20210020331	S20210020313	S20210020318
IIIT Sri-City	IIIT Sri-City	IIIT Sri-City

Abstract- In this machine learning project on Human Activity Recognition, a dataset derived from 30 participants, aged 19-48, was utilized. Each individual performed six activities while wearing a waist-mounted smartphone equipped with inertial sensors capturing 3-axial linear acceleration and angular velocity at 50Hz. The goal is to classify activities such as walking, walking upstairs, walking downstairs, sitting, standing, and lying down. The dataset was randomly split into training (70%) and test (30%) sets. The sensor signals underwent preprocessing, including noise filtering and sampling in fixed-width sliding windows with a 2.56-second duration and 50% overlap. Further, the acceleration signal was separated into body and gravitational components, and feature vectors were extracted from each window, encompassing both time and frequency domain variables. The challenge is to develop a robust machine learning model capable of accurately classifying human activities based on these features, thereby enabling the recognition and categorization of daily activities from sensor data.

Introduction

The features used for human activity recognition in this project are derived from both time and frequency domains. From each 2.56-second window with 50% overlap, various variables are calculated to form a feature vector. Time domain features may include statistical measures such as mean, standard deviation, minimum, and maximum values of the sensor signals for both linear acceleration and angular velocity. Additionally, features like skewness and kurtosis may provide insights into the distribution of the data. Frequency domain features are obtained through techniques such as Fast Fourier transform (FFT) applied to the sensor signals, yielding information about the frequency components present. These features could encompass dominant frequencies, energy distribution, and spectral entropy. The combination of time and frequency domain features allows the model to capture both the temporal patterns and frequency characteristics of human activities, providing a comprehensive set of information for effective classification.

Feature Description

Features using Accelerometer

tBodyAcc-mean():- Refers to the body accelerometer data. Accelerometer data measures the acceleration experienced by the body in different directions. In time-series analysis, data is often divided into segments, and statistical measures like mean are computed within each segment.
tBodyAccJerk-energy() :- Refers to the jerk component of the body's accelerometer data. Jerk is the rate at which acceleration changes over time, derived from accelerometer readings.
energy():-Denotes the energy calculation for the signal. In signal processing, energy is often computed as the sum of the squares of the values in a given signal or segment.
tBodyAccJerk-iqr() :- Refers to the jerk component of the body's accelerometer data. Jerk signifies the rate at which acceleration changes over time, typically derived from accelerometer readings.
iqr():-Denotes the calculation of the interquartile range. The interquartile range is a measure of statistical dispersion, representing the range between the first quartile (25th percentile) and the third quartile (75th percentile) of a dataset
tBodyAccJerk-mean():- This refers to the jerk component of the body accelerometer data. Jerk represents the rate at which acceleration changes over time, derived from the accelerometer readings.
mean():- Denotes that the value represents the mean (average) of the body acceleration jerk.

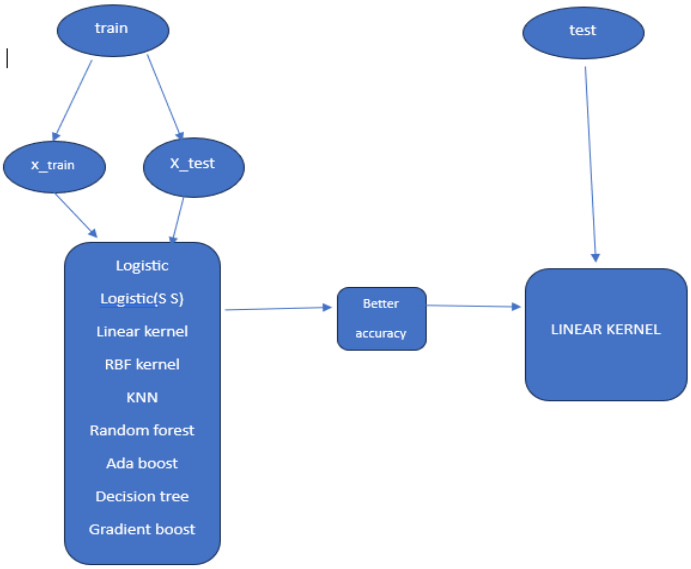
tGravityAcc-mean():- This likely refers to the gravity component of the accelerometer data. Accelerometer data measures acceleration along different axes (X, Y, Z) and can be separated into components caused by gravity and components caused by motion or external forces.
tGravityAcc-arCoeff() :- This refers to the gravity component within the accelerometer data, as discussed earlier. It represents the acceleration due to gravity isolated from other movements or forces acting on the device or individual.
arCoeff(): Stands for Auto-Regressive Coefficients. Auto-regressive models are used in time series analysis to model the relationship between a variable and its lagged values.
tBodyAccJerk-correlation() : Refers to the jerk component of the body's accelerometer data, representing the rate of change of acceleration over time.
correlation(): Denotes the calculation of correlation. Correlation measures the strength and direction of the relationship between two variables. In this case, it computes the correlation coefficient between the X-axis and Z-axis components of the jerk signal.
tBodyAccJerk-min(): Refers to the jerk component of the body's accelerometer data, which measures the rate of change of acceleration over time.
min(): Denotes that the value represents the minimum value within a given time window or segment. In time-series analysis, this indicates the smallest recorded value.
tBodyAccJerk-max(): Refers to the jerk component of the body's accelerometer data, measuring the rate of change of acceleration over time.
max(): Denotes that the value represents the maximum value within a given time window or segment. In time-series analysis, this indicates the largest recorded value.
tBodyAccJerk-std(): Refers to the jerk component of the body's accelerometer data, which measures the rate of change of acceleration over time.
std(): Denotes that the value represents the standard deviation within a given time window or segment. In time-series analysis, standard deviation measures the amount of variation or dispersion of a dataset.

Features using Gyroscope

tBodyGyro-max() : Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
max(): Denotes that the value represents the maximum value within a given time window or segment. In time-series analysis, this indicates the highest recorded angular velocity.
tBodyGyro-min(): Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
min(): Denotes that the value represents the minimum value within a given time window or segment. In time-series analysis, this indicates the lowest recorded angular velocity.
tBodyGyro-mad(): Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
mad(): Denotes the calculation of the Median Absolute Deviation. MAD is a measure of statistical dispersion that indicates the median of the absolute deviations from the median value.
tBodyGyro-std(): Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
std(): Denotes that the value represents the standard deviation within a given time window or segment. In time-series analysis, the standard deviation measures the amount of variation or dispersion in a dataset.
tBodyGyro-energy(): Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
energy(): Denotes the energy calculation for the signal. In signal processing, energy is often computed as the sum of the squares of the values in a given signal or segment.
tBodyGyro-iqr(): Refers to the body's gyroscope data, which measures the rate of rotation or angular velocity.
iqr(): Denotes the calculation of the Interquartile Range
tBodyGyro-mean(): Refers to the body's gyroscope data. Gyroscopes measure angular velocity, which represents the rate of rotation or change in orientation of the body.
mean(): Denotes that the value represents the mean (average) of the angular velocity. In time-series analysis, data is often segmented, and statistical measures like the mean are computed within each segment.

tBodyGyroJerk-mad(): Refers to the jerk component of the body's gyroscope data, representing the rate of change of gyroscope readings over time.
mad(): Denotes the calculation of the Median Absolute Deviation. MAD is a measure of statistical dispersion that indicates the median of the absolute deviations from the median value.
tBodyGyroJerk-mean(): Refers to the jerk component of the body's gyroscope data, representing the rate of change of gyroscope readings over time.
mean(): Denotes that the value represents the mean (average) of the gyroscope jerk. In time-series analysis, data is often segmented into intervals, and statistical measures like the mean are computed within each segment.
tBodyGyroJerk: Refers to the jerk component of the body's gyroscope data, indicating the rate of change of readings over time
max(): Denotes that the value represents the maximum value within a given time window or segment. In time-series analysis, this indicates the highest recorded gyroscope jerk value.

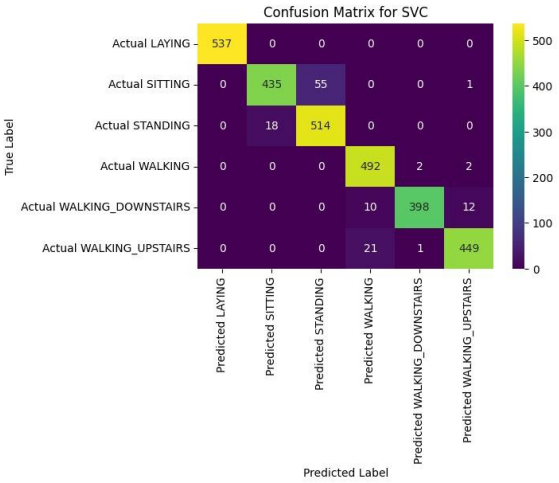
Block Diagram



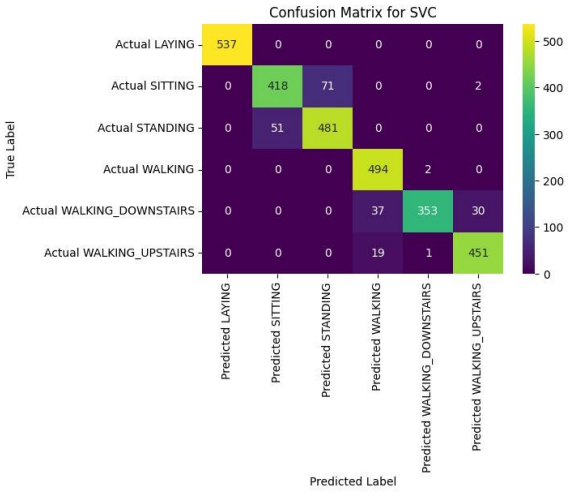
Methodology

We have the pre-processed dataset, training data and test data. We now split the training data into two parts, namely x_train data and x_test data. Now we use the above-mentioned models and fit our x_train data and predict the x_test data accuracy for each model. The model which gave the higher accuracy for the x_test is used for the original test data set for predicting.

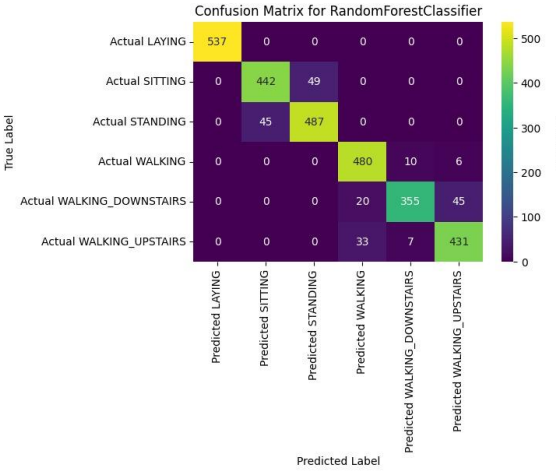
Analysis

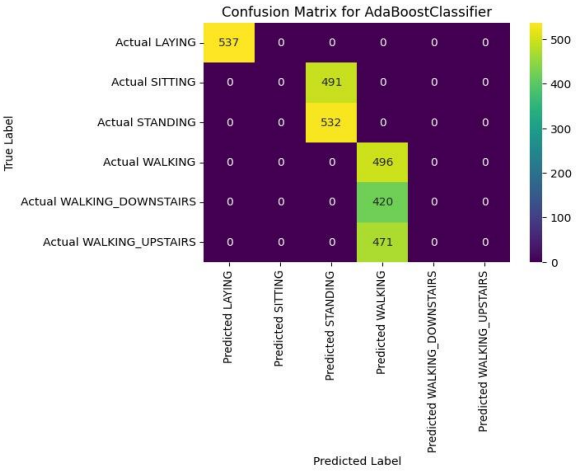


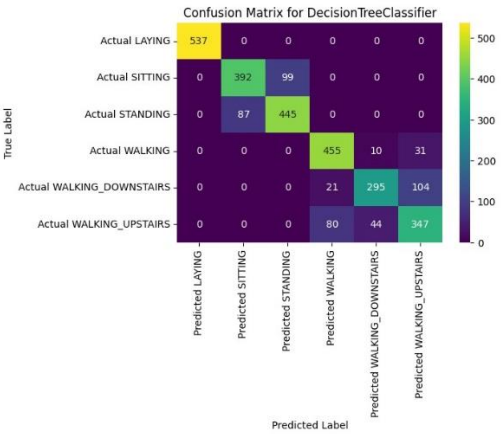
Linear SVC

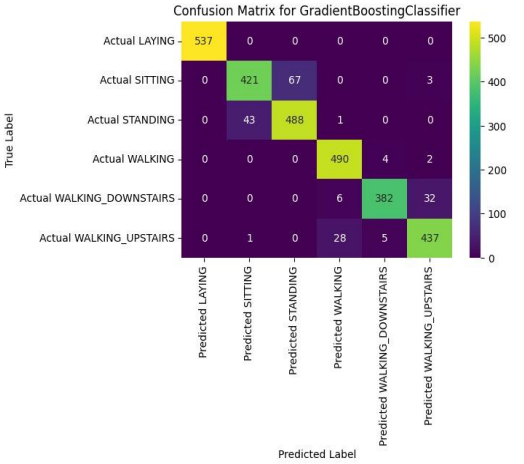


RBF SVC

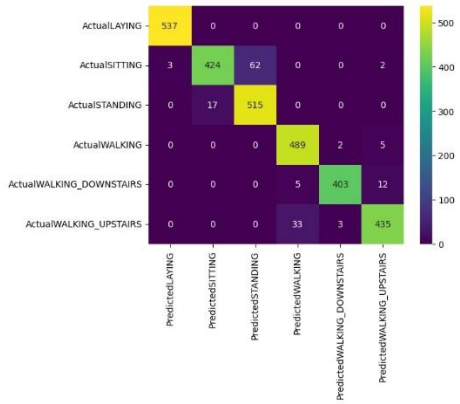




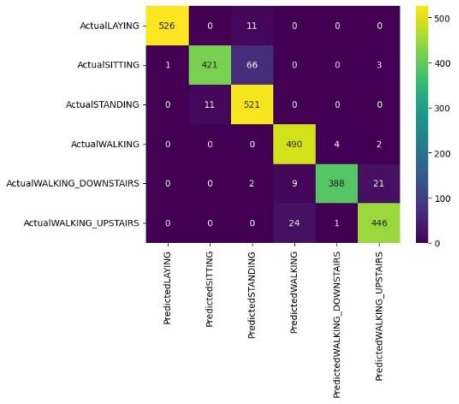




Logistic Regression



Pipeline(Standard scaler, Logistic Regression)



Results

Models	Accuracy
SVM.SVC(KERNEL='LINEAR')	0.9877634262406526
SVM.SVC(KERNEL='RBF')	0.9551325628823929
RANDOMFORESTCLASSIFIER	0.9830047586675731
ADABOOSTCLASSIFIER	0.5322909585316111
DECISIONTREECLASSIFIER	0.947654656696125
GRADIENTBOOSTINGCLASSIFIER	0.9850441876274643
SVC	0.9789259007477906
LOGISTIC REGRESSION	0.9836845683208701
KNN	0.9802855200543847
LOGISTIC REGRESSION(PIPELINING)-	0.982324949014276

Models	F1_scores
SVM.SVC (Kernel='Linear')	0.9585249783661883
SVM.SVC (Kernel='RBF')	0.9272623705257748
RandomForestClassifier	0.9267949307264421
AdaBoostClassifier	0.39440793478364267
DecisionTreeClassifier	0.837518667569274
GradientBoostingClassifier	0.9346721323123757
Logistic Regression	0.9509232260558226
KNN	0.787868108311999
Logistic Regression(Pipelining)	0.9473998241377237

NOTE

We have used SFS for feature selection , but the server crashed after 2 hours. We even tried by reducing the dataset , still the server crashed , so we haven't implemented SFS.

Conclusion

It was observed that for linear kernel in SVM had the highest accuracy for the training data divided into train data and test data. So we have used this model for our original test data with all features and with some features selected using correlation .