Physical Activity Classified by Tri-Axial Accelerometer

Dominique Barnes
Data Science Institute, Brown University
December 10th, 2023
GitHub [3]

1. Introduction

Complex motion capture systems are commonly used in the field of biomechanics. These motion capture systems combine motion capture beads and high speed cameras to determine and classify different activities. The average cost of motion capture systems can range between $25,000 to $500,000 [1]. The WIreless Sensor Data Mining (WISDM) lab at Fordham University utilizes data mining to answer research related questions from powerful devices such as cellular phones to build useful applications [2]. Mobile devices have begun to incorporate powerful sensors including GPS sensors and acceleration sensors (i.e. accelerometers). Although the use of advanced motion capture systems can have certain benefits such as higher precision and accuracy in classifying different motions, the systems are expensive and can be time consuming to run. The use of the accelerometers in the mobile devices can be beneficial in classifying activities in a more time and cost effective manner. The objective of this project was to train four machine learning algorithms to classify a user's motion based on the phone's triaxial accelerometer's readings.

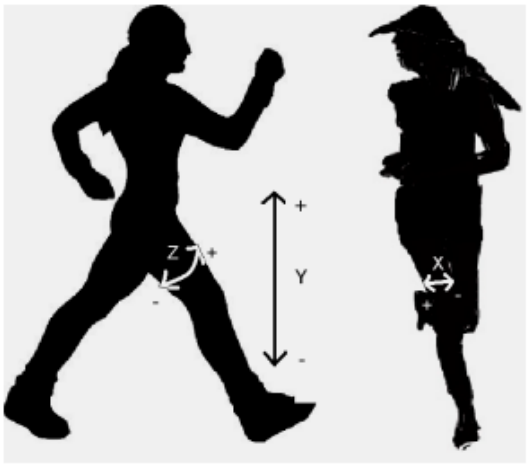

Figure 1 Axes of motion [2]

The dataset was collected by the WISDM lab in a controlled laboratory setting. They chose an Android-based cell phone due to the free, open-source, and easy-to-use operating system. Accelerometer data from twenty-nine users as they performed the activities were collected using the Android-phone's tri-axial accelerometer (Figure 1).

The users carried the Android phone in their front pants leg pocket and were asked to complete the set of six activities for specific periods of time (Figure 2). It is important to note that certain activities contain fewer example than other, because the users were not (i.e. jogging, climbing stairs) for very long. Certain activities, like standing and sitting, were added after the study had begun and was not collected for some users. Accelerometer data was collected every 50ms to have 20 samples per second. The data used for this project was transformed by the WISDM lab by dividing the data into 10-second segments and generating features based on the 200 readings contained within each segment. Forty-three summary features were generated which were variants of six basic features were used to predict the target variable (Table I).

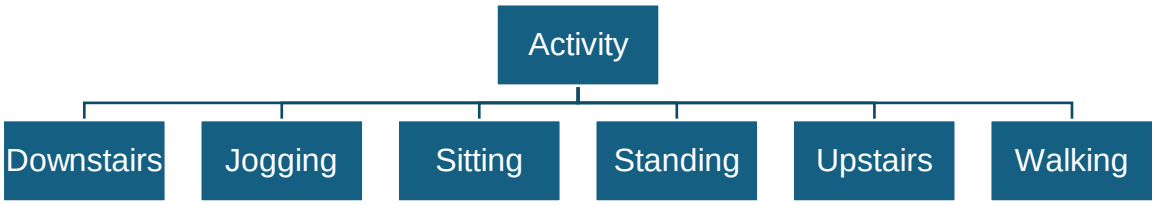

Figure 2 Activity classifications

Table I: The six basic features

Basic Feature	**Feature Description**
Average	Average acceleration for each axis
Standard Deviation	Standard deviation for each axis
Average Absolute Deviation	Average absolute deviation between the value of each of the 200 readings with the segment and the mean value over those 200 values for each axis
Average Resultant Acceleration	Average of the square roots of the sum of the values of each axis squared over the segment $\sqrt{(x_i^2 + y_i^2 + z_i^2)}$
Time Between Peaks	Time in milliseconds between peaks in the sinusoidal waves associated with most activities for each axis
Binned Distribution	Determined the range of values for each axis (maximum – minimum), divided the range into 10 equal sized bins, then recorded the fraction of the 200 values within each bin
Target Variable	Activity Class: Downstairs, jogging, sitting, standing, upstairs, walking

2. EDA

With the transformed data, an exploratory data analysis (EDA) was completed in order to have a better understanding of the underlying class imbalance and the relations between features and the target variable. The number of times each class had been recorded was plotted below (Figure 3). From this plot it is evident there is a slight imbalance in the target variable. Walking and jogging were recorded more times than the other activity classes recorded. Walking was recorded 38.4% and jogging was recorded 29.9% while the other classes were recorded less than 10% of the time.

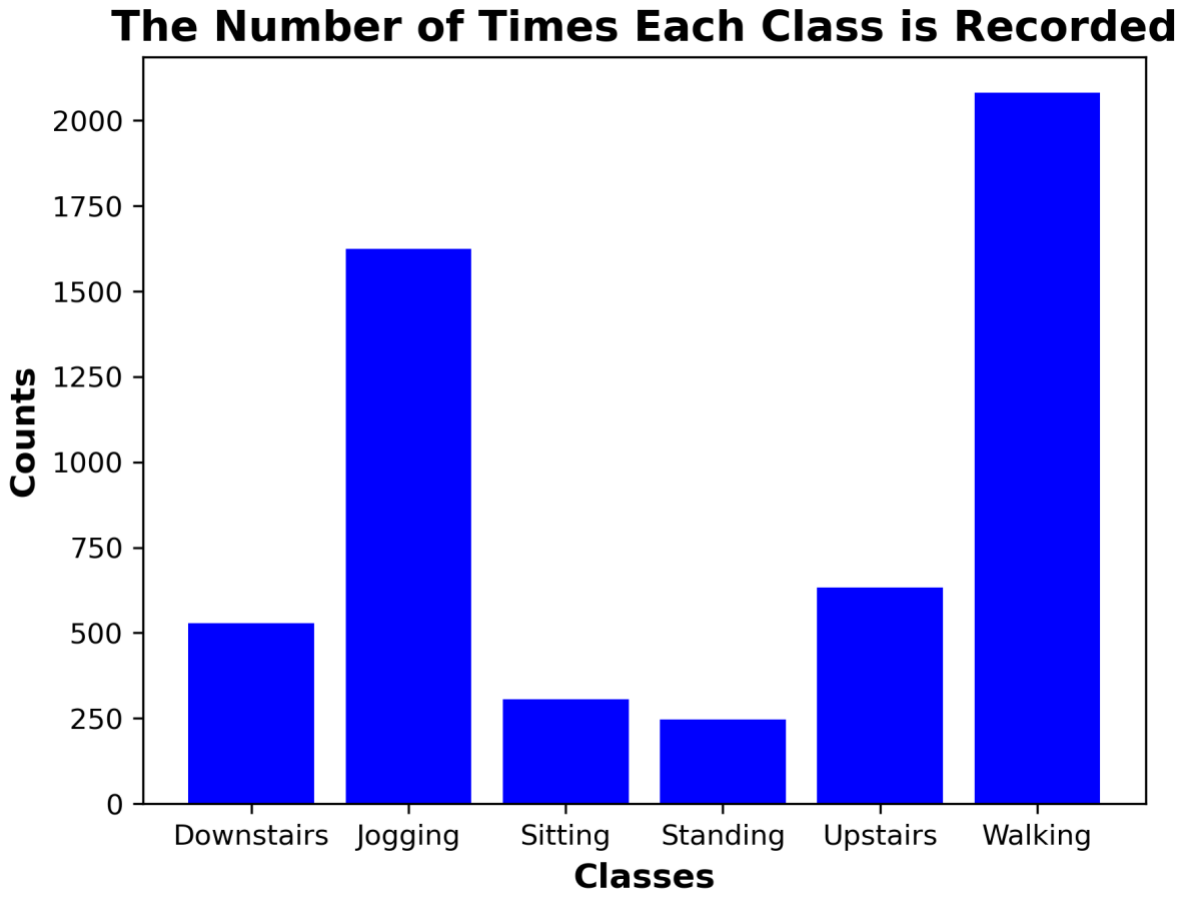

Figure 3 A bar chart of the physical activity classes recorded

In order to better understand how the machine learning algorithms may train, and therefore classify each class the average accelerations were plotted by each class. The X axis did not collect very useful data as it was only measuring how the leg moved left-to-right (Figure 1). Therefore, only the Y and Z axes' average accelerations were plotted. The Y axis indicated the rate at which the leg moved up and down during the specific activity classes (Figure 4). While the Z axis will indicated the rate at which the leg swung back-to-forth (Figure 5).

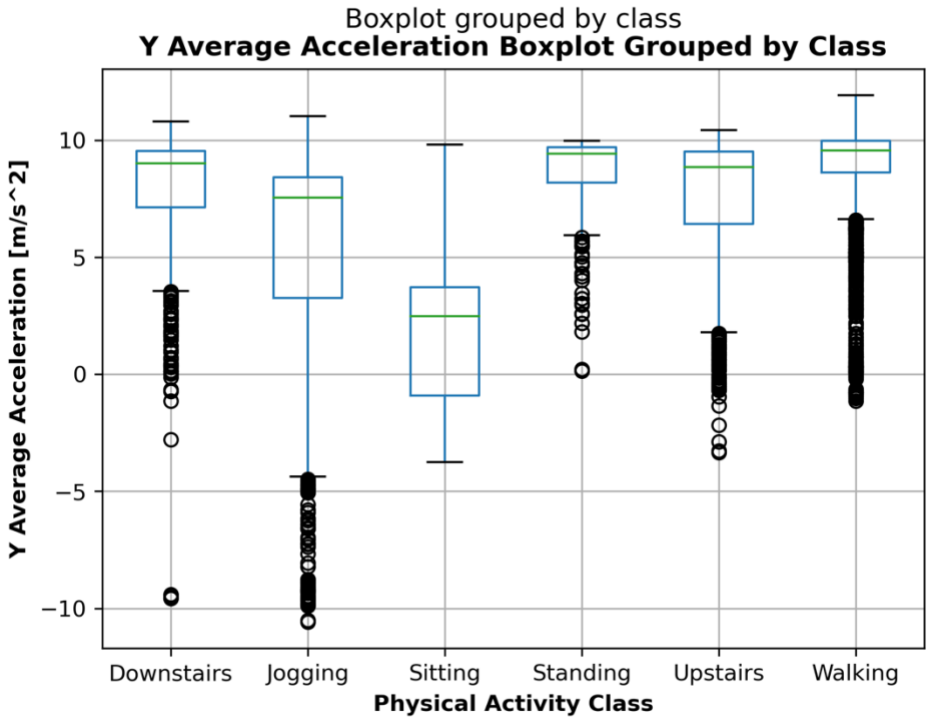

Figure 4 Boxplot of the Y-axis average acceleration grouped by classes

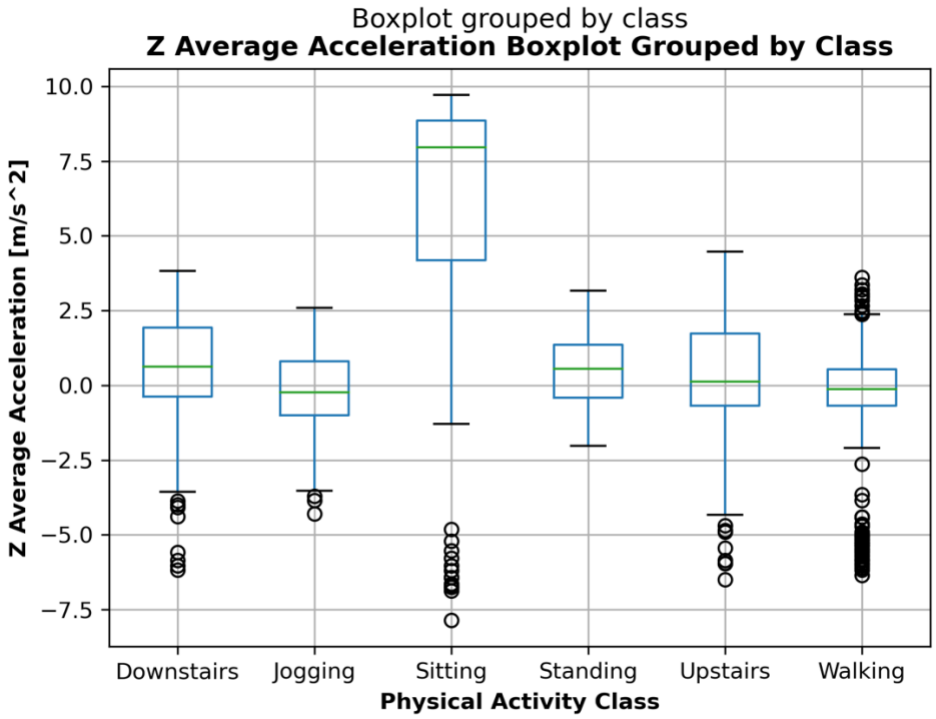

Figure 5 Boxplot of the Z-axis average acceleration grouped by classes

3. Methodology

To begin the model development process, the data was split into train-validation-test splits. All of the features in this dataset are continuous values, therefore, for the preprocessing step the *StandardScaler* method was used. There was a small amount of values (9.3%) missing from the features ['XPEAK','YPEAK','ZPEAK']. The next step was to train and evaluate the performance of four models – XGBoost Classifier, Logistic Regression with L2 Regularization, Random Forest Classifier, Support Vector Machine Classifier (SVC). Four evaluation metrics were measured: Accuracy, precision, recall, and F1 beta score. The main evaluation metric used for interpretation was accuracy.

3.1 GridSearchCV and Model Pipeline

Splitting and Preprocessing

The first step of the pipeline was to split the data. The data is non-iid because there was group structure based on the user. The test set was created using the *GroupShuffleSplit* to split the data in 20% of the 36 users. The other 80% of the data was then split into validation and test set using *GroupKFold* with 5 folds. From the remaining 80%, there was a split of 60% for the training set and 20% for the validation set. No columns were added after preprocessing because all of the features were continuous values and used *StandardScaler* to preprocess.

Hyperparameter Tuning

In order to optimize a model's performance, the model should be tuned through several parameter configurations. However, due to the computational time it will take to iterate through every parameter combination, only two parameters were tuned per model (Table II). Once the optimal parameter configuration is determines, the model refits the test set and is evaluated. A summary of the pipeline is presented below (Figure 6).

Table II Machine learning algorithms and hyperparameters

Model	**Tuned Hyperparameter**
XGBoost	N_Estimators: [50, 100, 500] Max_Depth: [1, 3, 10, 30]
Logistic Regression L2 Regularization	C: [0.001, 0.1, 1, 10, 100] Max_Iter: [1000, 5000]
Random Forest	Max_Depth: [1, 3, 10] Max_Features: [0.5, 0.75, 1.0]
SVC	C: [0.1, 0, 1] Gamma: [0.001, 0.1, 1, 1000]

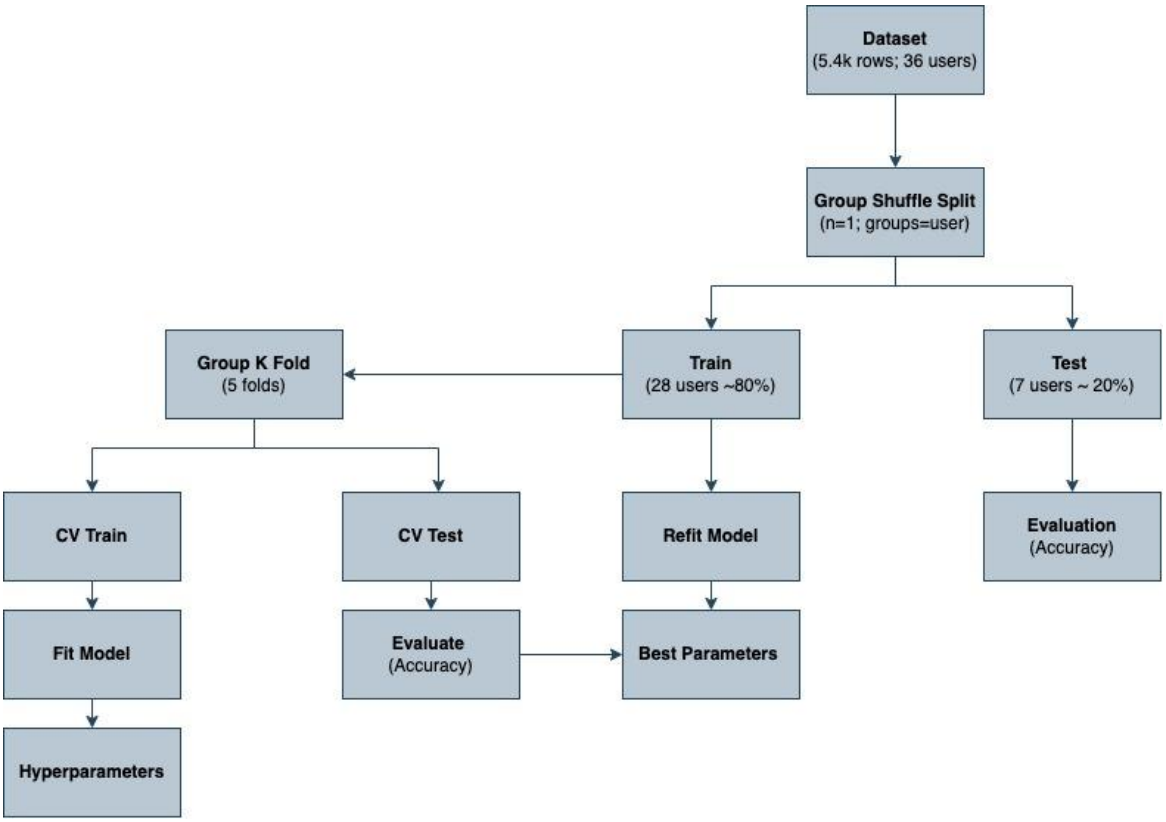

Figure 6 Flow chart of model development

4. Results

In order to determine how well the model performed with different train-validation-test set combinations, 10 random states were generated and ran through the pipeline describe in section 3. The mean accuracy score was calculated along with the standard deviation and plotted with the baseline model's recorded accuracy score (Figure 7).

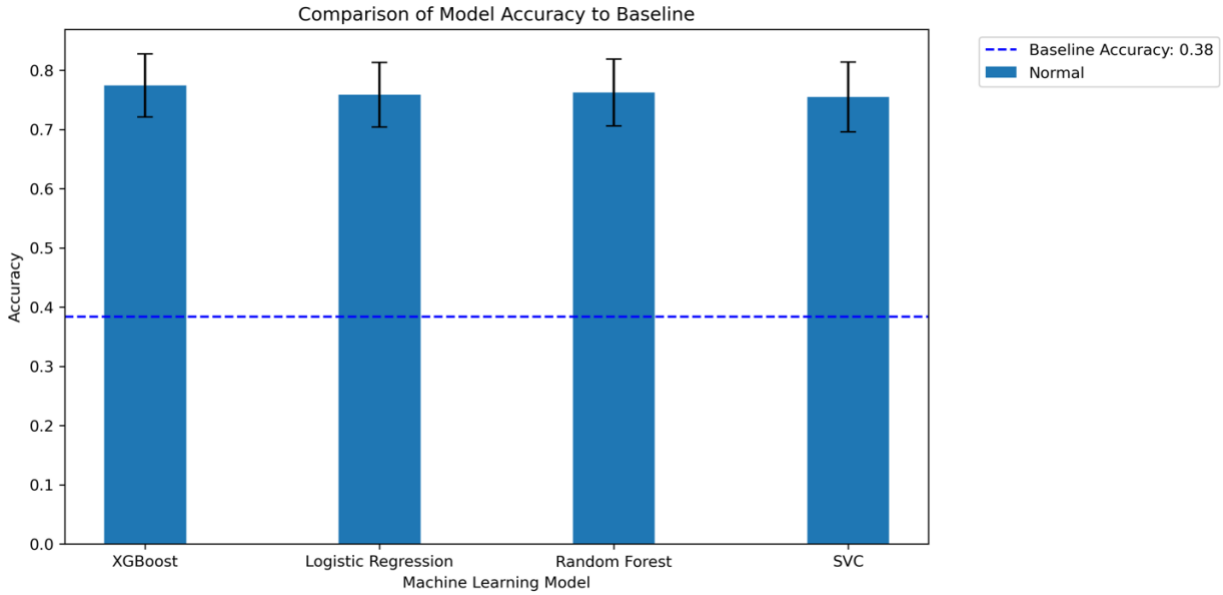

Figure 7 Average accuracy of each model compared to baseline accuracy score

Every model performed better than the baseline model at the classification task. Although each model did well at completing the classification task, it is hard to determine just exactly how well each model did at classifying the specific tasks. Therefore, the evaluation metrics values were reported (Table III) and a normalized confusion matrix was created and normalized to the true labels (Figure 8).

Table III Average evaluation metrics for each model

	XGBoost Mean +/-StDev	**Logistic Regression L2** Mean +/-StDev	**Random Forest** Mean +/-StDev	**SVC** Mean +/-StDev
Accuracy	0.774 +/-0.053	0.759 +/-0.054	0.766 +/-0.050	0.755 +/-0.059
Precision	0.755 +/-0.050	0.725 +/-0.067	0.768 +/-0.057	0.717 +/-0.064
Recall	0.774 +/-0.053	0.759 +/-0.054	0.766 +/-0.050	0.755 +/-0.059
F1 Beta Score	0.751 +/-0.053	0.718 +/-0.060	0.748 +/-0.061	0.708 +/-0.059

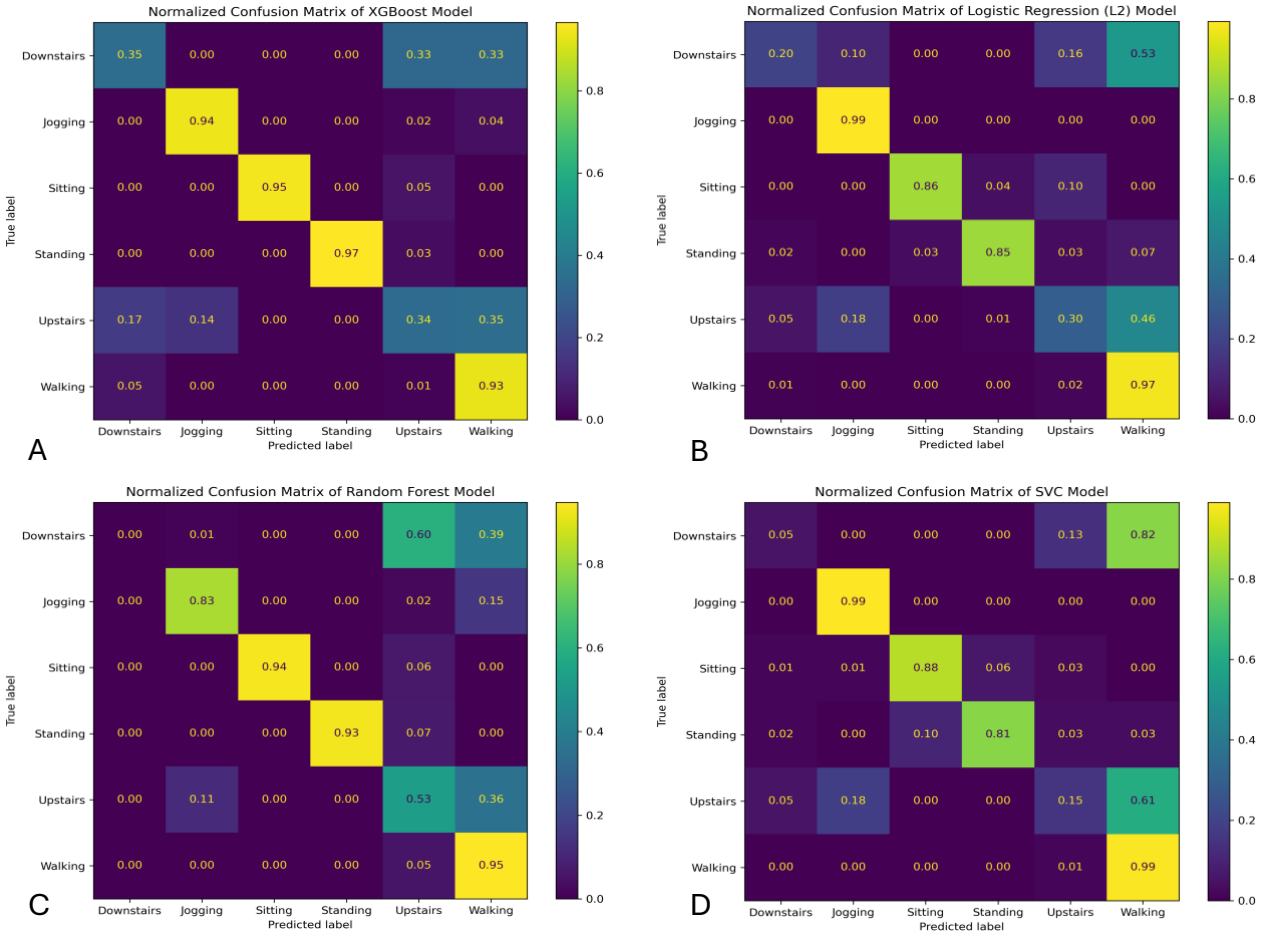

Figure 8 Normalized confusion matrices: A) XGBoost, B) Logistic Regression L2 Regularization, C) Random Forest, D) Support Vector Machine Classifier (SVC)

The average evaluation metrics show that the XGBoost had the highest accuracy, precision, and F1 beta score. While the Random Forest model had the highest recall score. The normalized confusion matrices confirm what was to be expected, XGBoost and Random Forest models did best at classifying each activity class (Figure 8 A and B). The XGBoost normalized confusion matrix show slightly better performance than the Random Forest model at classifying specifically difficult classes such as downstairs and upstairs. The confusion matrix from the XGBoost shows that the downstairs activity was commonly misclassified. This can have been for various of reasons. It may have been due to the data imbalance in the target variables (Figure 2), so the models may have not been able to properly learn the specific activity of walking downstairs. It may also have been due to the common speed or motion at which the users walked down the stairs that may have been similar to walking up the stairs that could have led to the misclassification of the downstairs activity class.

4.1 Global Feature Importance

Global feature importance allows for a better understanding as to why certain classes may have gotten misclassified as mentioned previously. Three types of importance metrics were used for the best model which was determined to be XGBoost: 1) Feature Importance, 2) XGBoost Total Gain Importance, 3) SHAP Feature Importance. Feature importance is used to estimate predictive power based on the dropping of model performance due to dropping of a feature (Figure 9). XGBoost gain importance estimated the predictive power based on the improvement in accuracy based on the addition of a feature(Figure 10). SHAP used Shapely values from game theory(Figure 11).

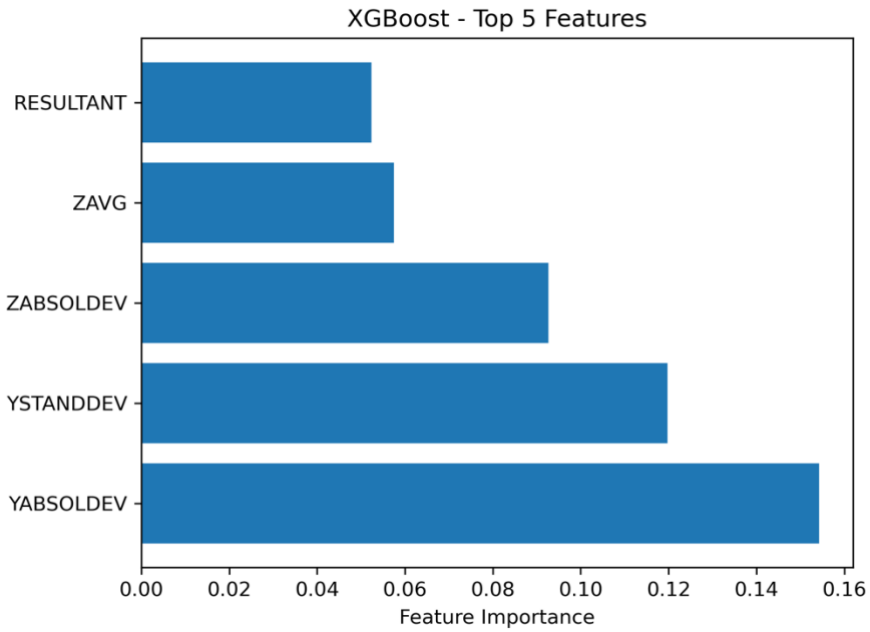

Figure 9 Top 5 important features based on XGBoost model

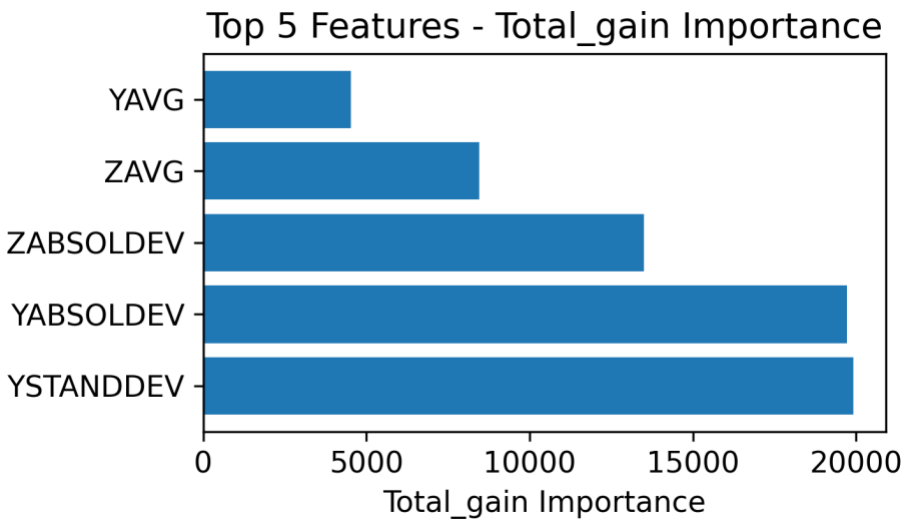

Figure 10 Top 5 features based on XGBoost total gain importance

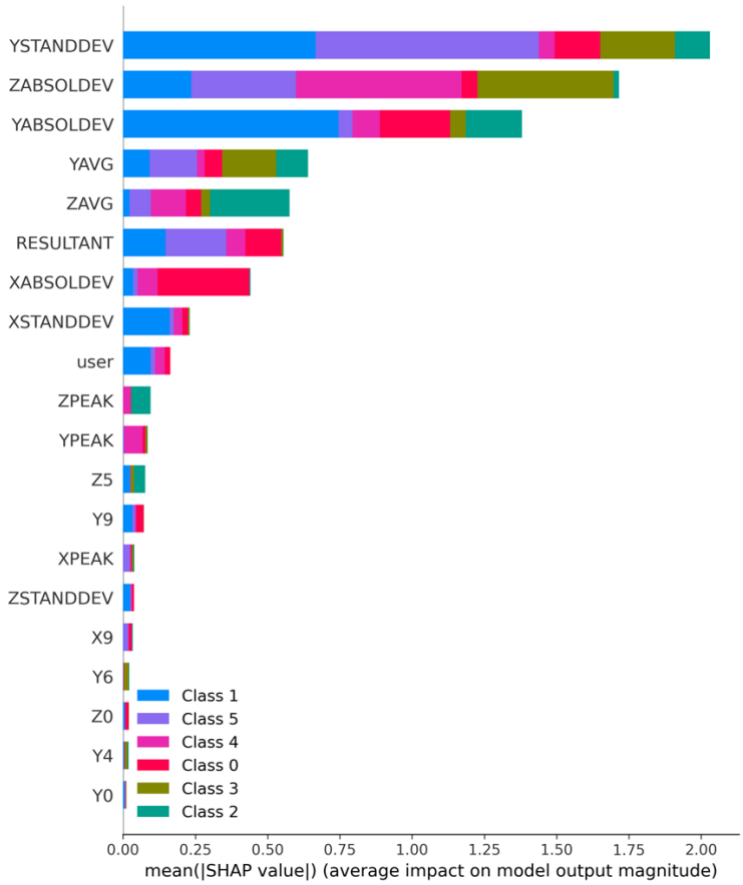

Figure 11 Top 20 features based on Shap summary

The Y Absolute Deviation was consistently in the top 3 features regardless of feature importance metric used. The Shap summary shows how certain features vary in importance depending on class. There were also more Y and Z features in the top ranks as compared to X features which was to be expected.

4.2 *Local Feature Importance*

The local feature importance metrics can provided a clearer insight on how the model determine which class each datapoint was decided to. The shap values for the XGBoost model were used to create the force plots below. Highlighted in this report are two instances, 1) the most correctly classified activity which was standing (Figure 12), classified correctly 97% of the time by the XGBoost model, and 2) the most misclassified activity which was walking downstairs, classified correctly 35% of the time by the XGBoost model and misclassified 65% (Figure 13).

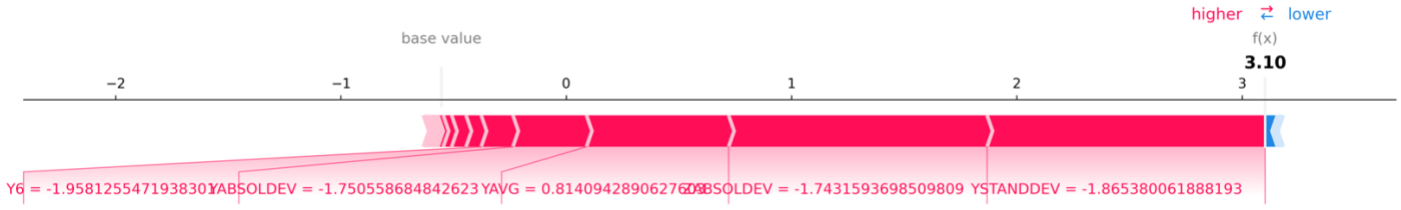

Figure 12 Shap force plot to determine the local feature importance for an indexed standing class

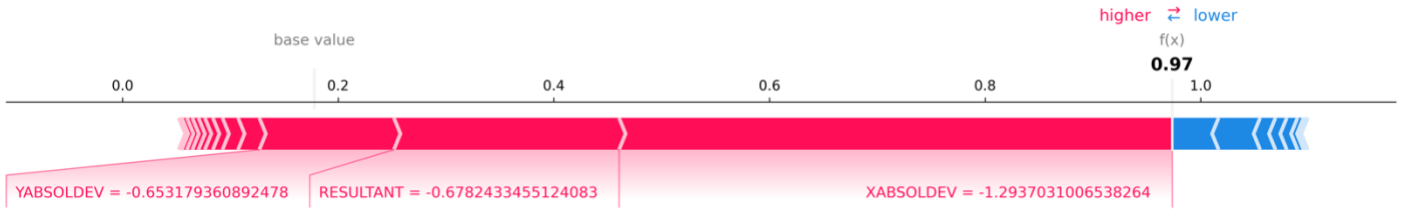

Figure 13 Shap force plot to determine the local feature importance for an indexed downstairs class

It was surprising to see that the X absolute deviation played an important role in the classification of walking downstairs, seeing that the X axis should not differentiate much when completing this activity class. On the contrary it is expected to see a Y axis feature be important in the classification of standing. Little to no deviation in the Y axis would correctly classify whether the user is standing or not.

5. Outlook

If given additional time and resources, I would have increased the sample size of this dataset to at least 100 users. I would have also ensures that the target variables were balanced to enhance the predicative power of the model. I would do this by making sure each user completed the same amount of tasks for the same amount of time. I would like to have explored other the use of deep learning models used by the WISDM lab such as multilayer perceptrons. I would also have liked to include other health metrics such as muscle activity and heart rates to allow for the classes with similar motions such as downstairs and upstairs to have more distinction between the classes.

References

[1] Rokoko. (2022, June 17). The Complete Guide to professional motion capture. Intuitive and affordable motion capture tools for character animation. https://www.rokoko.com/insights/the-complete-guide-to-professional-motion-capture

[2] Kwapisz, J. R., Weiss, G. M., & Moore, S. A. (2011). Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, *12*(2), 74-82.

[3] GitHub Repository - https://github.com/dbarnes16/DATA1030-Final