Human Activity Recognition

Kirtiraj Khandekar
School of Coumputing and Information
University of Pittsburgh
Pittsburgh, PA 15213
kik26@pitt.edu

Nikunj Goel
School of Coumputing and Information
University of Pittsburgh
Pittsburgh, PA 15213
nig54@pitt.edu

Sai Pradeep Peri
School of Coumputing and Information
University of Pittsburgh
Pittsburgh, PA 15213
sap187@pitt.edu

April 21, 2020

Abstract

Data mining is a good way to find the relationship between raw data and predict the target we want which is also widely used in different field nowadays. In this project, we implement a lots of technology and method in data mining to predict the activity of human based on the sensor data. We created a strong model to predict the activity of the user and the same can be translated to real life scenario with lesser data and same accuracy.

1 Introduction

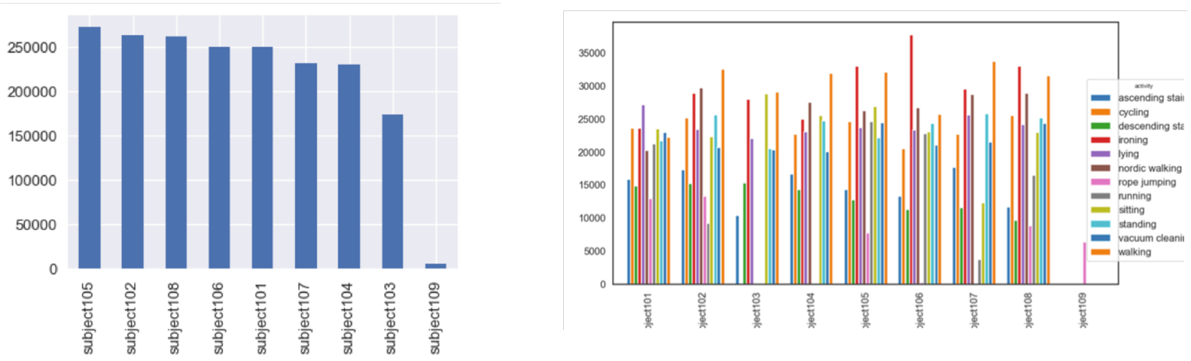
Activity recognition is an important task in several healthcare applications. By continuously monitoring and analyzing user activity it is possible to provide automated recommendations to both patients and doctors [1, 2]. There are also applications to consumer products such as data logging for smart watches health apps [3]. Common consumer devices such as smart phones and smart watches generally ship with IMUs (Inertial Measurement Unit), which are packaged accelerometer and gyroscope sensors. Through the information provided by these IMUs, machine learning techniques can then be used to train activity classifiers, giving users, doctors, and app developers access to an individual's lifestyle and activity choices. In this paper we examine what is the "best" classification technique and how well can it perform with less features. In particular, we examine the use of Logistic Regression, Random Forest and various boosted decision trees such as LightGBM and XGBoost.

2 Data Set and Features

We used the PAMAP2 Dataset from the UCI repository of machine learning data sets. [16, 15] This data set includes raw 9-axis IMU data streams as well as heart rate data from nine different subjects performing various activities. This data is collected using 3 Colibri wireless IMUs and a BM-CS5SR heart rate monitor. The IMUs are placed on the dominant hand, chest and ankle of the participant. Data was collected for a total of 10 hours from each participant. Each IMU provides temperature, 3-axis acceleration, 3-axis angular velocity, and 3-axis magnetometer data at a rate of 2000 Hz which records one record every **0.2** second. In total there are 1.9 million data points, each containing 52 features. In the data set, each time-step is labeled with an activity ID, one of 12 different activities that the subjects were engaged in. The 12 activities are the following: ironing, walking, lying, standing, sitting, Nordic walking, vacuum cleaning, cycling, ascending stairs, descending stairs, running, rope jumping. We combined each subjects data into a single matrix and divided that into a training and test set. We used 70% of our data for training and 30% for testing. To avoid over-fitting on only a certain subset of classes, we randomly split the data between training and testing. To train our models, we performed 5-fold cross-validation using our training set, to tune the hyper-parameters for the given model. We then used this optimized model and analyzed its performance on the test set.

3 Data Exploration

- Data was in the form of 9 .DAT files of all 9 participants. We appended all the files into 1 single data frame to make data processing easy and efficient.
- As described by the data author in the report that at times there was data drop because of wireless sensors so we removed all the rows with NAN records.
- We removed all the records with activity **0** because it's a **transient activity**. Transient activity is any activity between 2 activities.
- We removed all the Orientation column which are 4 per IMU as they do not provide valid data.



(a) Frequency of the activities performed by a particular subject

(b) Distinct activities performed by a particular subject

Figure 1: Graph plots which indicates that subject 9 performed least number of activities

- Subject 9 was removed from the analysis of the data.

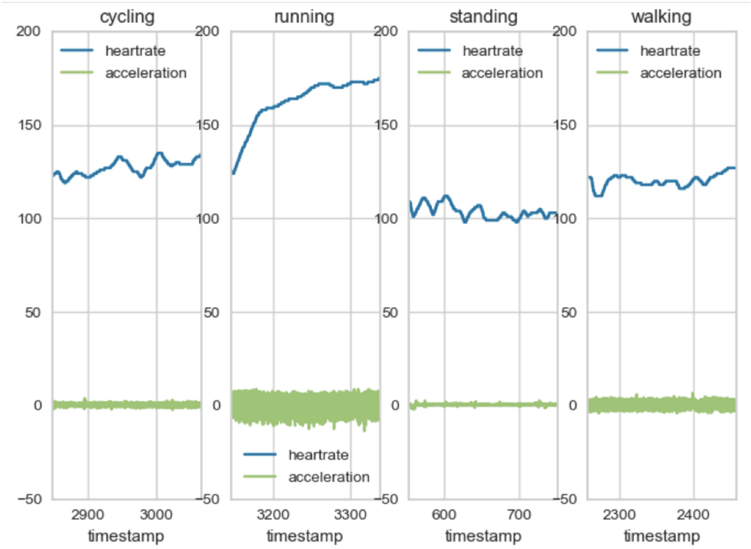


Figure 2: Raw Acceleration(from chest IMU) and heart rate data

Figure 2 addresses the mystery question of whether the activities are separable given the PAMAP dataset. It clearly shows that activities in the data are clearly and neatly separable by taking into account just chest IMU and heart rate.

- **As the data was taken in every 250 ms it ensures that the data is no too duplicated to give vague results. A normal resting subject heart beat is approximately 1 beat per 600 ms and while running is 1 beat per 200-250 ms.**
- **Also we are not distinguishing between a chest exercise or biceps exercise rather we are distinguishing between running, cycling, standing and ironing. The problem listed sounds intimidating but provided the dataset with humongous number of features given by a particular subject it looks accomplish-able.**
- **Nowadays FANG comanies main resarch area is to distuinguish perfectly within a cluster of exercises in gym to manufacture better smart watches. They are spending millions to generate this data and will be published soon.**

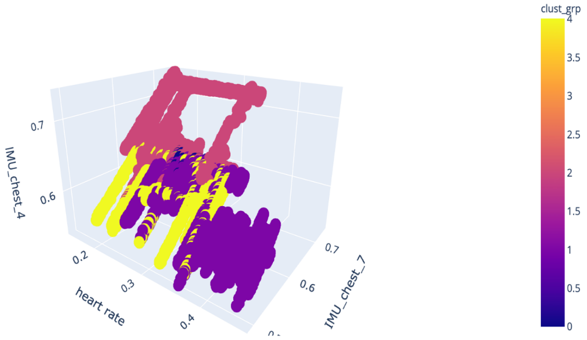


Figure 3: 3D view of 5 clusters after K-means clustering on all the features.

Above plot also shows that the data is separable as we can see distinct boundaries between different clusters.

4 Problem Statement : Why is it interesting?

The goals of this project in physical activity monitoring is to estimate the performed activities and to identify basic or recommended activities and postures. The PAMAP dataset with different data processing methods and classification algorithms, will be used to create a benchmark of physical activity classification problems.

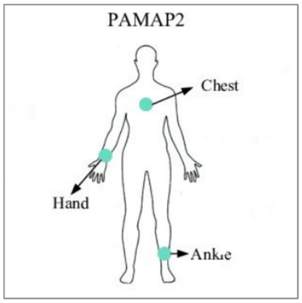


Figure 4: Walking? Running? Sleeping? Driving?

4.1 Why it is interesting?

4.1.1 Healthcare and lifestyle

It can be useful for elderly assistance.

4.1.2 Sports and fitness

Provide the user with feedback and motivation.

4.1.3 Industrial

It can be effectively used in safety, daily analysis and training.

4.1.4 Robotics

Interaction with the objects and the people.

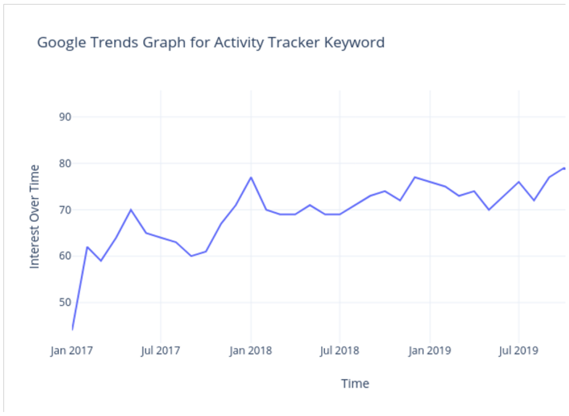
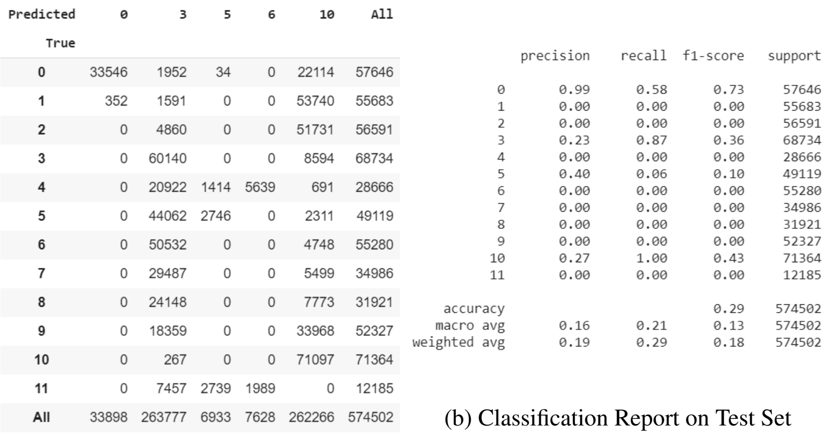


Figure 5: Google trends for the keyword Activity Tracker

5 Methods

5.1 Logistic Regression

As a baseline measure, we incorporated a standard logistic regression model for multiclass classification. Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. We have used MultinomialNB classifier as provided in scikit learn package. It implements the naive Bayes algorithm for multinomially distributed data. The distribution is parametrized by vectors $\theta y = (\theta y1, \ldots, \theta yn)$ for each class y, where n is the number of features and θyi is the probability $P(xi|y)$ of feature i appearing in a sample belonging to class y. Using this model, we achieved a multi class macro AUC score of 0.886 on test set. A confusion matrix and classification report for the model is shown below.



Predicted	0	3	5	6	10	All
True						
0	33546	1952	34	0	22114	57646
1	352	1591	0	0	53740	55683
2	0	4860	0	0	51731	56591
3	0	60140	0	0	8594	68734
4	0	20922	1414	5639	691	28666
5	0	44062	2746	0	2311	49119
6	0	50532	0	0	4748	55280
7	0	29487	0	0	5499	34986
8	0	24148	0	0	7773	31921
9	0	18359	0	0	33968	52327
10	0	267	0	0	71097	71364
11	0	7457	2739	1989	0	12185
All	33898	263777	6933	7628	262266	574502

(a) Confusion Matrix

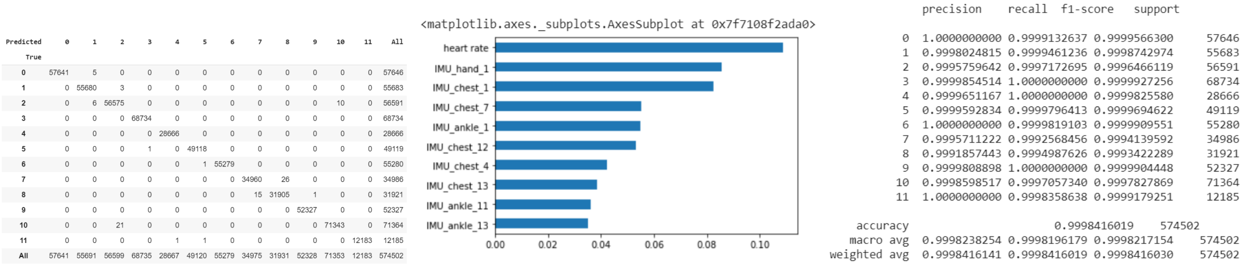
```
              precision    recall  f1-score   support

           0       0.99      0.58      0.73     57646
           1       0.00      0.00      0.00     55683
           2       0.00      0.00      0.00     56591
           3       0.23      0.87      0.36     68734
           4       0.00      0.00      0.00     28666
           5       0.40      0.06      0.10     49119
           6       0.00      0.00      0.00     55280
           7       0.00      0.00      0.00     34986
           8       0.00      0.00      0.00     31921
           9       0.00      0.00      0.00     52327
          10       0.27      1.00      0.43     71364
          11       0.00      0.00      0.00     12185

    accuracy                           0.29    574502
   macro avg       0.16      0.21      0.13    574502
weighted avg       0.19      0.29      0.18    574502
```

(b) Classification Report on Test Set

Figure 6: Plots for Multinomial Logistic Regression

5.2 Random Forest

One of the ensemble methods for decision trees, for the reason of improving forecast precision, is random forest. Random Forest is a form of bagging (bootstrap aggregation), which involves examining with substitution from the original population to diminish variance (at the expense of an increase in bias, increased computational cost, and decreased interpretability of the trees). For a random forest, a huge number of decision trees are generated, and the bias is further decreased (by decorrelating the trees) by only considering a subset of the total number of features at each split in the decision tree [4]. We decided to include 100 trees in our random forest model, because 100 was a well-performing trade-off of time and accuracy (for random forests, increasing the number of trees will only serve to decrease the variance, and will not increase the likelihood of overfitting). We used the default option of only considering a random subset of the square-root of the total number of features for each split. Using this model, we achieved a multi class macro AUC score of 0.99999 on test set. A confusion matrix, variable importance and classification report for the model is shown below.



Predicted True	0	1	2	3	4	5	6	7	8	9	10	11	All
0	57641	5	0	0	0	0	0	0	0	0	0	0	57646
1	0	55680	3	0	0	0	0	0	0	0	0	0	55683
2	0	6	56575	0	0	0	0	0	0	0	10	0	56591
3	0	0	0	68734	0	0	0	0	0	0	0	0	68734
4	0	0	0	0	28666	0	0	0	0	0	0	0	28666
5	0	0	0	1	0	49118	0	0	0	0	0	0	49119
6	0	0	0	0	0	1	55279	0	0	0	0	0	55280
7	0	0	0	0	0	0	0	34960	26	0	0	0	34986
8	0	0	0	0	0	0	0	15	31905	1	0	0	31921
9	0	0	0	0	0	0	0	0	0	52327	0	0	52327
10	0	0	21	0	0	0	0	0	0	0	71343	0	71364
11	0	0	0	0	1	1	0	0	0	0	0	12183	12185
All	57641	55691	56599	68735	28667	49120	55279	34975	31931	52328	71353	12183	574502

(a) Confusion Matrix

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7108f2ada0>
```

(b) Variable Importance

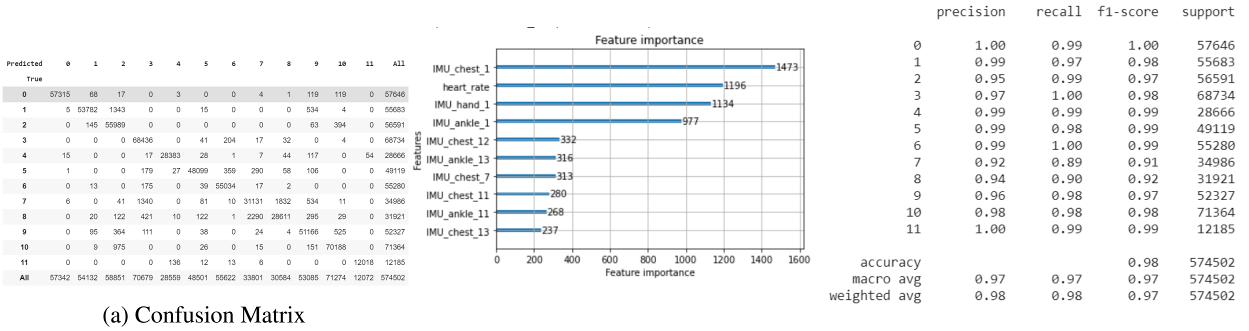
```
             precision    recall  f1-score   support

           0 1.0000000000 0.9999132637 0.9999566300     57646
           1 0.9998024815 0.9999461236 0.9998742974     55683
           2 0.9995759642 0.9997172695 0.9996466119     56591
           3 0.9999854514 1.0000000000 0.9999927256     68734
           4 0.9999651167 1.0000000000 0.9999825580     28666
           5 0.9999592834 0.9999796413 0.9999694622     49119
           6 1.0000000000 0.9999819103 0.9999909551     55280
           7 0.9995711222 0.9992568456 0.9994139592     34986
           8 0.9991857443 0.9994987626 0.9993422289     31921
           9 0.9999808898 1.0000000000 0.9999904448     52327
          10 0.9998598517 0.9997057340 0.9997827869     71364
          11 1.0000000000 0.9998358638 0.9999179251     12185

    accuracy                         0.9998416019    574502
   macro avg 0.9998238254 0.9998196179 0.9998217154    574502
weighted avg 0.9998416141 0.9998416019 0.9998416030    574502
```

(c) Classification Report on Test Set

Figure 7: Plots for Random Forest

5.3 LightGBM

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks. Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise. So when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm. Light GBM is prefixed as 'Light' because of its high speed. Light GBM can handle the large size of data and takes lower memory to run. Since our data set is huge, we choose to use LightGBM too. For this model, we have set objective as multi-class with $num_classes$ 12 since we have 12 classes in labels. The metric is multi log-loss which is used for multi class classification problems. We have set max_depth to 3 to curtail over-fitting with default learning rate. Using this model, we achieved a multi class macro AUC score of 0.99946 on test set. A confusion matrix, variable importance and classification report for the model is shown below.



Predicted True	0	1	2	3	4	5	6	7	8	9	10	11	All
0	57315	68	17	0	3	0	0	4	1	119	119	0	57646
1	5	53782	1343	0	0	15	0	0	0	534	4	0	55683
2	0	145	55989	0	0	0	0	0	0	63	394	0	56591
3	0	0	0	68436	0	41	204	17	32	0	4	0	68734
4	15	0	0	17	28383	28	1	7	44	117	0	54	28666
5	1	0	0	179	27	48099	359	290	58	106	0	0	49119
6	0	13	0	175	0	39	55034	17	2	0	0	0	55280
7	6	0	41	1340	0	81	10	31131	1832	534	11	0	34986
8	0	20	122	421	10	122	1	2290	28611	295	29	0	31921
9	0	95	364	111	0	38	0	24	4	51166	525	0	52327
10	0	9	975	0	0	26	0	15	0	151	70188	0	71364
11	0	0	0	0	136	12	13	6	0	0	0	12018	12185
All	57342	54132	58851	70679	28559	48501	55622	33801	30584	53085	71274	12072	574502

(a) Confusion Matrix

(b) Variable Importance

```
              precision    recall  f1-score   support

           0       1.00      0.99      1.00     57646
           1       0.99      0.97      0.98     55683
           2       0.95      0.99      0.97     56591
           3       0.97      1.00      0.98     68734
           4       0.99      0.99      0.99     28666
           5       0.99      0.98      0.99     49119
           6       0.99      1.00      0.99     55280
           7       0.92      0.89      0.91     34986
           8       0.94      0.90      0.92     31921
           9       0.96      0.98      0.97     52327
          10       0.98      0.98      0.98     71364
          11       1.00      0.99      0.99     12185

    accuracy                           0.98    574502
   macro avg       0.97      0.97      0.97    574502
weighted avg       0.98      0.98      0.97    574502
```

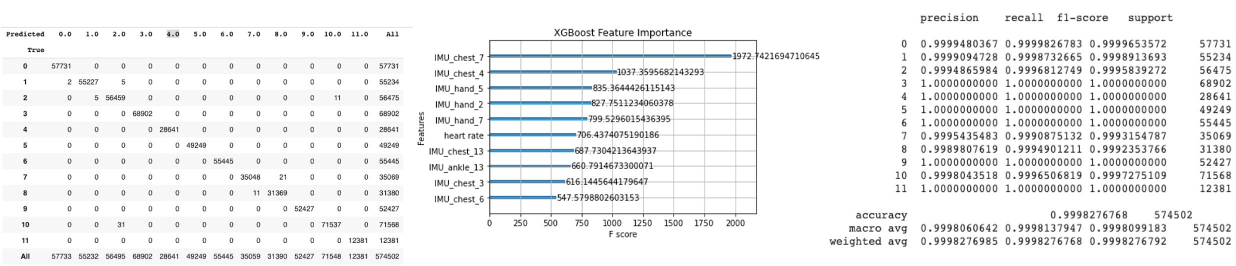
(c) Classification Report on Test Set

Figure 8: Plots for LightGBM

5.4 XGBoost

One form of ensembling is using ”boosting,” which combines many ”weak learners” (simple decision trees) in order to reduce bias in the model (at the expense of increasing variance). Boosting these weak decision trees is done for the purpose of ideally improving accuracy, though it may be prone to overfitting [4]. One specific algorithm for boosting, is XGBoost which is used in this project.

For XGBoost, we used the default learning rate, and used manual tuning to tune the maximum depth of the base decision tree estimators, and the number of trees. Since the base estimator of boosting should be a ”weak learner,” we decided to limit our weak learners to having a maximum depth of less than or equal to 5. For the Dataset, we finally chose 500 trees of max-depth=5. Using these models, dataset scored an **Macro AUC(Computes the AUC of each class against the rest with multi-Class = ’ovr’)** of 0.999 on the test set. A confusion matrix,variable importance and classification report for the model is shown below.



Predicted	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	All
True													
0	57731	0	0	0	0	0	0	0	0	0	0	0	57731
1	2	55227	5	0	0	0	0	0	0	0	0	0	55234
2	0	5	56459	0	0	0	0	0	0	0	11	0	56475
3	0	0	0	68902	0	0	0	0	0	0	0	0	68902
4	0	0	0	0	28641	0	0	0	0	0	0	0	28641
5	0	0	0	0	0	49249	0	0	0	0	0	0	49249
6	0	0	0	0	0	0	55445	0	0	0	0	0	55445
7	0	0	0	0	0	0	0	35048	21	0	0	0	35069
8	0	0	0	0	0	0	0	11	31369	0	0	0	31380
9	0	0	0	0	0	0	0	0	0	52427	0	0	52427
10	0	0	31	0	0	0	0	0	0	0	71537	0	71568
11	0	0	0	0	0	0	0	0	0	0	0	12381	12381
All	57733	55232	56495	68902	28641	49249	55445	35059	31390	52427	71548	12381	574502

(a) Confusion Matrix

(b) Variable Importance

```
                precision    recall  f1-score   support

           0  0.9999480367 0.9999826783 0.9999653572     57731
           1  0.9999094728 0.9998732665 0.9998913693     55234
           2  0.9994865984 0.9996812749 0.9995839272     56475
           3  1.0000000000 1.0000000000 1.0000000000     68902
           4  1.0000000000 1.0000000000 1.0000000000     28641
           5  1.0000000000 1.0000000000 1.0000000000     49249
           6  1.0000000000 1.0000000000 1.0000000000     55445
           7  0.9995435483 0.9990875132 0.9993154787     35069
           8  0.9989807619 0.9994901211 0.9992353766     31380
           9  1.0000000000 1.0000000000 1.0000000000     52427
          10  0.9998043518 0.9996506819 0.9997275109     71568
          11  1.0000000000 1.0000000000 1.0000000000     12381

    accuracy                          0.9998276768    574502
   macro avg  0.9998060642 0.9998137947 0.9998099183    574502
weighted avg  0.9998276985 0.9998276768 0.9998276792    574502
```

(c) Classification Report on Test Set

Figure 9: Plots for XGBoost

6 Results and Conclusions

In conclusion, we have used PAMAP2 dataset which comprises of 9 subjects performing 12 protocol activities over a period of 10 hours. First, we preprocessed the raw data by removing unnecessary data. Then, we extracted the features from time and frequency domain to use in the classifier models. Finally, we performed classification on the transformed data. Multiple performance metrics have been used for evaluating the classifier models and we have managed to achieve considerable accuracy in classification. Unsurprisingly, logistic regression performed the worst. Having the advantage of extremely low memory usage and speed for predictions, it can still be a viable method in low compute devices like micro-controllers. For decision tree methods, as expected, ensemble methods improved test performance over logistic regression for the data set. Boosted decision trees performed slightly better than random forest as can be seen from the test scores of XGBoost, which is promising because it is far less computationally intensive, and thus is a good candidate for a model to actually deploy in a smart device.

7 Future Work

In future, we would like to use deep learning models on this data set. As such, simple feed-forward neural nets would be sufficient for this problem. However, we would like to explore CNN’s (Convolutional Neural Networks) which could potentially give similar or improved performance while using substantially less memory. We would also like to test our models on limited feature set as in real life scenario, we might not have data from all the 3 IMU sensors and the heart rate monitor. Promising results on this limited data set would be good for actual deployment in smart devices. We could also test these models using real IMU’s. In particular, we would want to see if a low-compute embedded device could perform classifications with neural nets or computationally cheaper methods such as decision trees, in real-time. Heartbeat or other physiological data can be used to measure the intensity of the activity being performed. If we can manage to capture and recognize not only what activity has been performed but also how well it has been performed, then that could open new doors for various applications. Major impact will be in sports and fitness field, where athletes or individuals could monitor how well they are training or performing. It could also prove useful in health field with elderly living assistance and rehabilitation patients.

8 Code Base

https://github.com/nikunjgoel95/Human-Activity-Recognition.git

References

[1] S. M. M. S. C. B. H. R. B. A. P. Y. T. R. P. H. A. S. A. MATTHEWS, Charles E; GEORGE, "Amount of time spent in sedentary behaviors and cause-specific mortality in us adults," *The American journal of clinical nutrition*, 2012.

[2] P. T. Katzmarzyk and I.-M. Lee, "Sedentary behaviour and life expectancy in the usa: a cause-deleted life table analysis," *BMJ Open*, vol. 2, no. 4, 2012.

[3] X. Chai and Q. Yang, "Multiple-goal recognition from low-level signals.," pp. 3–8, 01 2005.

[4] N. Andrew, "Cs229 course notes decision trees," *CS229 online course notes.*