Activity Analyzer with Wearable Sensors

Thunradee Tangsupakij, XinYi Wang, Yu-Chieh Wang

Cpt_S 415 Big Data, Fall 2019
Washington State University

December 15, 2019

1 Introduction

The World Health Organization (WHO) recommends people ages 5 to 17 years old to have at least one hour of exercise daily to improve their health and reduce risks of having depression, cardiovascular diseases and obesity[1]. According to [2], more than 80% of young generations have physical activities less than an hour a day. Technologies have changed people's lifestyle, support us to have easier lives but also lazier lives. Today we have Amazon that delivers goods right in front of our doors, and Uber Eats that delivers foods to our places. However, health effects of technologies are not always bad. In the past decade, wearable sensors have been developing widely. As they've become smaller, lighter and more stylish, they've gained more popularity in daily usage. Combining big data techniques with machine learning to handle the future's health-focused data is essential to help with making healthy decisions.

In 2012, researchers in the Netherlands compared performance of three open source databases in regards to sensor data storage [3]. One of the three databases was SQL database (PostgreSQL) and the other two were NoSQL databases (Cassandra and MongoDB). The result showed that Cassandra had best performance in large critical sensor applications, MongoDB had best performance in small to medium size and non-critical sensor applications, and PostgreSQL has best performance in reading and flexible queries. Each of these aspects are important for machine learning algorithms to be used in conjunction with them.

Many machine learning algorithms such as Naive Bayes, Decision Tree , KNN, SVM, etc., have been used to classify human activity [4]. In 2015, researchers from the National University of Sciences Technology in Pakistan used Naive Bayes, Decision Tree, KNN and SVM to classify human activity from smartphone sensors data[5]. Their work showed that decision trees gave the highest recall, precision and accuracy which are 0.942, 0.944 and 0.985 respectively.

In this project, we aim to recognize human activity by analyzing wearable sensors data using machine learning. Section 2 describes the working process which includes data collection, data storage and management, data processing and machine leaning. Section 3 presents results of our model. Section 4 discusses some problems that happened in the project. Section 5 concludes the project and provides directions for future works. Finally, section 6 summarizes project contributions of each team member.

2 Methods

2.1 Data Collection

We obtained The University of Sussex-Huawei Locomotion-Transportation (SHL) Dataset from shl-dataset.org [6][7]. The SHL dataset was collected from sensors in four mobile phones placed on each participant. One phone each were located in their hand, on their chest, hip and finally one in their backpack or side bag. In the dataset, these locations are referred to as Hand, Torso, Hips and Bag respectively. The SHL dataset came in the form of text files, containing a wide range of features, including a file containing all of the feature labels. In this project, we only used four motion files which are Hand_motion.txt, Torso_motion.txt, Hips_motion.txt and Bag_motion.txt while using only the following features:

- Accelerometer: x, y, z in m/s^2
- Gyroscope: x, y, z in rad/s
- Magnetometer: x, y, z in μT

The SHL dataset contains 8 different activities as follows:

- Still: standing or sitting; inside or outside a building
- Walking: inside of outside
- Run
- Bike
- Car: as driver, or as passenger
- Bus: standing or sitting; lower deck or upper deck
- Train: standing or sitting
- Subway: standing or sitting

The activities were collected over the span of three days.

2.2 Data Storage and Management

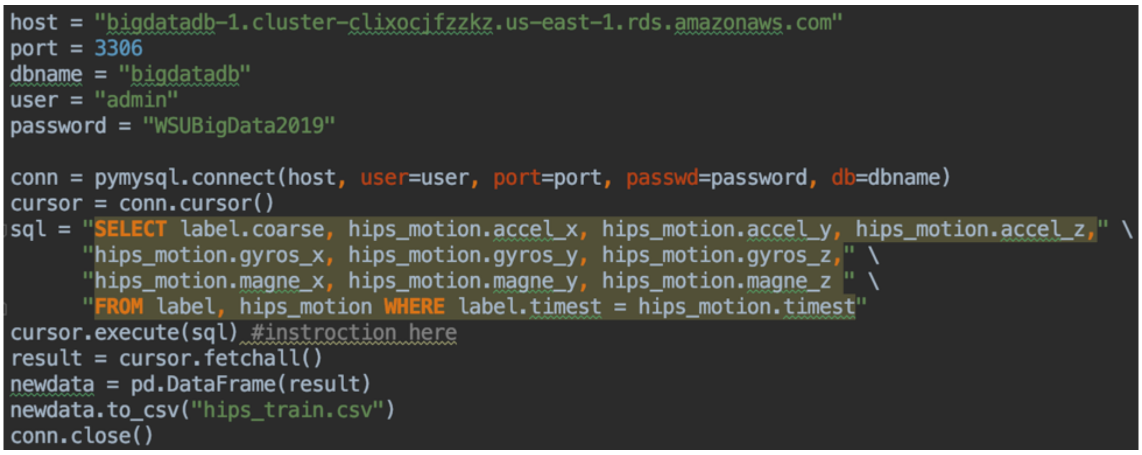
In this project we attempted to work with cloud services by using Amazon Web Service (AWS). Since our data is well structured and not gathered in real-time, we decided to go with a relation based database system (RDS): Amazon Aurora MySQL.

As we mentioned in the previous section, we used only four motion files and one label file. Each of the four motion files is approximately 1.09GB, and the size of the label file is 134.8MB. The total size of our data is about 4.49GB, leading us to select a db.r5.large as our database instance. According to the AWS website[],the db.r5.large database instance had 2 vCPU, 10 ECU, 16 GiB memory, up to 3,500 Mbps bandwidth and up to 10 Gbps network performance. However, we had encountered financial issues during the project's process which will be addressed in section 4. As a result, we had to change our database instance to the smallest one which was db.t2.medium. The db.t2.medium database instance had 2 vCPU, ECU available, 4 GiB memory, no information in terms of bandwidth and moderate network performance.

We spent more than 5 hours just attempting to read from db.t2.medium. This made us decide to store the data into a csv file and drop the database.

2.3 Data Processing

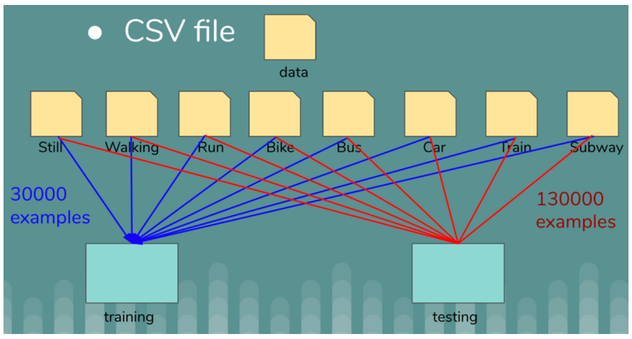
First of all, we write a Python code to access our database by SQL, download the data we need, and make it as a CSV file.



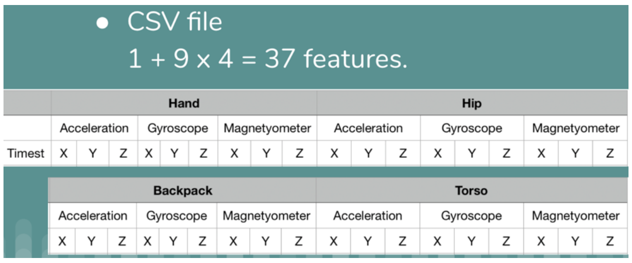
```
host = "bigdatadb-1.cluster-clixocjfzzkz.us-east-1.rds.amazonaws.com"
port = 3306
dbname = "bigdatadb"
user = "admin"
password = "WSUBigData2019"

conn = pymysql.connect(host, user=user, port=port, passwd=password, db=dbname)
cursor = conn.cursor()
sql = "SELECT label.coarse, hips_motion.accel_x, hips_motion.accel_y, hips_motion.accel_z," \
      "hips_motion.gyros_x, hips_motion.gyros_y, hips_motion.gyros_z," \
      "hips_motion.magne_x, hips_motion.magne_y, hips_motion.magne_z " \
      "FROM label, hips_motion WHERE label.timest = hips_motion.timest"
cursor.execute(sql) #instroction here
result = cursor.fetchall()
newdata = pd.DataFrame(result)
newdata.to_csv("hips_train.csv")
conn.close()
```

Second, we separate this CSV file to 8 CSV files by 8 activities so that we can pick some examples as our training data and use the rest of the examples as our testing data. Third, we random the order of the examples of our training and testing data. Next, We added the training data to the decision number algorithm to obtain a training model, and then put the test data into the training model to obtain the prediction result. Finally, the prediction result is compared with the actual label to get the prediction accuracy.

To form a dataset, we extract pure data from the original dataset to form a pure dataset for each activity. Then we take 5 mins data(30000 examples pre-activity) as our training data and the rest(130000 examples pre-activity) as the testing data, depending on the frequency at which the sensor collects the data. For the missing data and null data, we just directly drop them off from the dataset in case it has a



bad influence on the pure dataset. The datasets we use for training and testing are consist of 37 labels which are label and 36 features from 4 sensors.



- CSV file
  1 + 9 x 4 = 37 features.

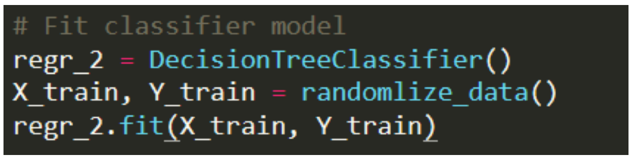
	Hand									Hip								
	Acceleration			Gyroscope			Magnetyometer			Acceleration			Gyroscope			Magnetyometer		
Timest	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z

Backpack									Torso								
Acceleration			Gyroscope			Magnetyometer			Acceleration			Gyroscope			Magnetyometer		
X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z

2.4 Machine Learning

First, we remodeling the raw data into the format that we want by using a Python data processing tool. Then we randomize the data but keep the label-features relation fixed. Finally, we put the features set and label set into the algorithm to fit it. In order to see the output result for the prediction accuracy, we use a specific command line from the scikit-learn decision tree which called accuracy_score. We can also visualize the instances that are correctly classified and not correctly classified through printing out the confusion matrix.

For an algorithm, We decide to use the DecisionTreeClassifier algorithm from the library of scikit-learn.

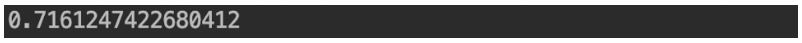


```
# Fit classifier model
regr_2 = DecisionTreeClassifier()
X_train, Y_train = randomlize_data()
regr_2.fit(X_train, Y_train)
```

Method: Here we want to point out some important methods that may impact the final result. First is the way we manage the raw data. We pick the data for individual activity from the pure data segment, which means we pick one-day data and filter out the target label we want with the features along with so that we can make sure the data we picked is especially for specific one activity. Secondly, it includes a mechanism inside of the algorithm that converts the continuous values to discrete values. The true method for it is, the algorithm finds the maximal and minimal value for one feature as an interval, then divides it into several small intervals with same space between, so that the algorithm can automatically classify multiple continuous values into several small labels(here the of small labels are equal to the of small intervals with same space). For example, if the maximal and minimal value is +10 and -10, then the interval can be [-10, +10], and if it can be divided into 2 intervals then it will be [-10, 0] and [0, +10]. If one of the values is 5 in one feature value set, it will be classified as label 1. If one of the values is -2 in one feature value set, it will be classified as label 0.

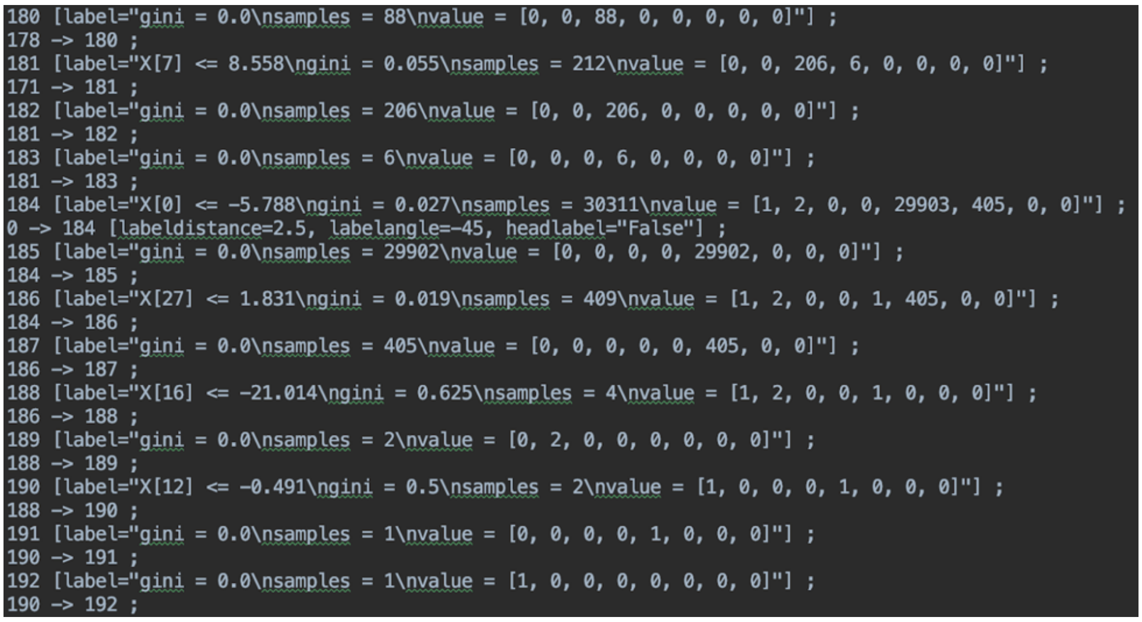
3 Results

The result shows that the decision tree algorithm works and gets the accuracy which is up to 71.61



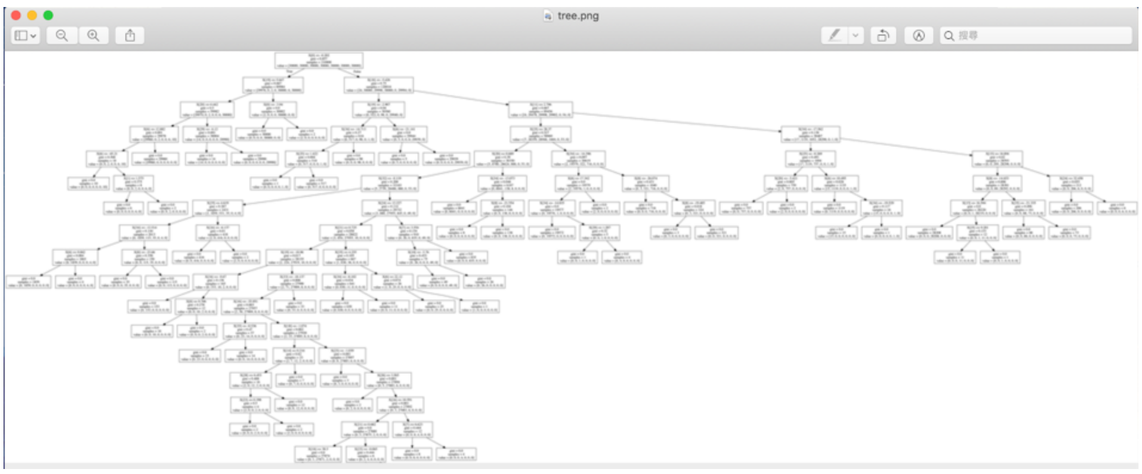
```
0.7161247422680412
```

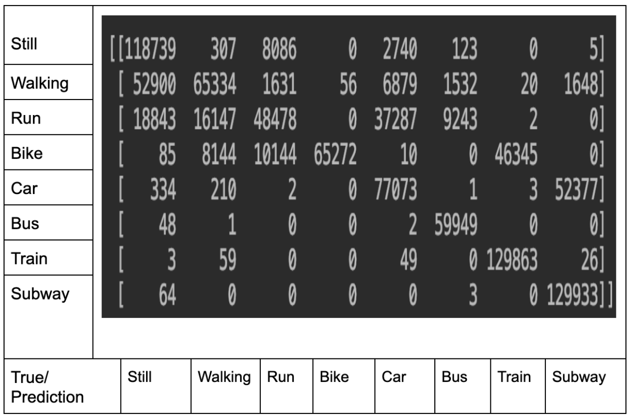
The following picture show there are a total of 190 dots in our decision tree model.



```
180 [label="gini = 0.0\nsamples = 88\nvalue = [0, 0, 88, 0, 0, 0, 0, 0]"] ;
178 -> 180 ;
181 [label="X[7] <= 8.558\ngini = 0.055\nsamples = 212\nvalue = [0, 0, 206, 6, 0, 0, 0, 0]"] ;
171 -> 181 ;
182 [label="gini = 0.0\nsamples = 206\nvalue = [0, 0, 206, 0, 0, 0, 0, 0]"] ;
181 -> 182 ;
183 [label="gini = 0.0\nsamples = 6\nvalue = [0, 0, 0, 6, 0, 0, 0, 0]"] ;
181 -> 183 ;
184 [label="X[0] <= -5.788\ngini = 0.027\nsamples = 30311\nvalue = [1, 2, 0, 0, 29903, 405, 0, 0]"] ;
0 -> 184 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
185 [label="gini = 0.0\nsamples = 29902\nvalue = [0, 0, 0, 0, 29902, 0, 0, 0]"] ;
184 -> 185 ;
186 [label="X[27] <= 1.831\ngini = 0.019\nsamples = 409\nvalue = [1, 2, 0, 0, 1, 405, 0, 0]"] ;
184 -> 186 ;
187 [label="gini = 0.0\nsamples = 405\nvalue = [0, 0, 0, 0, 0, 405, 0, 0]"] ;
186 -> 187 ;
188 [label="X[16] <= -21.014\ngini = 0.625\nsamples = 4\nvalue = [1, 2, 0, 0, 1, 0, 0, 0]"] ;
186 -> 188 ;
189 [label="gini = 0.0\nsamples = 2\nvalue = [0, 2, 0, 0, 0, 0, 0, 0]"] ;
188 -> 189 ;
190 [label="X[12] <= -0.491\ngini = 0.5\nsamples = 2\nvalue = [1, 0, 0, 0, 1, 0, 0, 0]"] ;
188 -> 190 ;
191 [label="gini = 0.0\nsamples = 1\nvalue = [0, 0, 0, 0, 1, 0, 0, 0]"] ;
190 -> 191 ;
192 [label="gini = 0.0\nsamples = 1\nvalue = [1, 0, 0, 0, 0, 0, 0, 0]"] ;
190 -> 192 ;
```

Then, we draw these points as an entity decision tree as following:





True/ Prediction	Still	Walking	Run	Bike	Car	Bus	Train	Subway
Still	118739	307	8086	0	2740	123	0	5
Walking	52900	65334	1631	56	6879	1532	20	1648
Run	18843	16147	48478	0	37287	9243	2	0
Bike	85	8144	10144	65272	10	0	46345	0
Car	334	210	2	0	77073	1	3	52377
Bus	48	1	0	0	2	59949	0	0
Train	3	59	0	0	49	0	129863	26
Subway	64	0	0	0	0	3	0	129933

Finally, we make a confusion matrix so that we can see which label has higher accuracy and which label is hard to be classified.

According to the plot, we can see that each activity(row) has 130000 examples. In addition, we find some interesting things in this plot. First of all, Subway has the highest accuracy which means people shaking on the subway are very consistent. Second, when people walk, they don't walk all the time, but stop and go. As a result, we can see some activity examples are misclassified, but some of the activities have high accuracies.

4 Discussion

We encountered numerous issues while conducting this project. The first major issue involves the storage method: We ran out of money before we could finish the project. This was because of a bad misunderstanding we had about cloud services. For the future, we will plan to do the whole process on cloud services as opposed to repeatedly downloading the data.

The second issue involves how we selected samples randomly. When we first obtain the dataset, we shuffle the dataset, planing to then take samples for cross validation. However, we misunderstood the properties of the dataset, and realize now that we should maintain an original copy of the dataset.

5 Conclusions and Future Works

Despite the issues in our project, we believe we were able to test the foundation of using big data methods for health base data. We found that using db.t2.medium is insufficient for this task; it seemed to work far too slow for our purposes. In term of classification, we could also improve our model in several ways. We leave it to future works to try methods such as random forests to better classify human activities.

6 Project Contributions

Thunradee Tangsupakij: First major task was to set up the database. This involves selecting a database type and analyzing features such as the cost, computation time and overall effectiveness. For the report, Thunradee wrote the abstract, sections

2.1, 2.2, Dicussion and Conclusion. She also edited the introduction and finalized the report.

XinYi Wang: Design prediction model and implement in Python language, plot the result and do with the output analysis.

Yu-Chieh Wang: Analysis of the topic and design the method of processing the dataset. Implement of extracting pure data of each activity from the original database and reshape the dataset chose to the data we need. Do an analysis of the dataset.

References

[1] Anon. 2017. Physical Activity. (May 2017). Retrieved December 15, 2019 from https://www.who.int/dietphysicalactivity/pa/en/

[2] Reuters. 2019. Teens too busy with their smartphones: Lack of exercise is making them obese prone to heart disease. (November 2019). Retrieved December 15, 2019 from https://economictimes.indiatimes.com/magazines/panache/teens-too-busy-with-their-smartphones-lack-of-exercise-is-making-them-obese-prone-to-heart-disease/articleshow/72178631.cms

[3] Jan Sipke Van Der Veen, Bram Van Der Waaij, and Robert J. Meijer. 2012. Sensor Data Storage Performance: SQL or NoSQL, Physical or Virtual. 2012 IEEE Fifth International Conference on Cloud Computing (2012). DOI:http://dx.doi.org/10.1109/cloud.2012.18

[4] Oscar D. Lara and Miguel A. Labrador. 2013. A Survey on Human Activity Recognition using Wearable Sensors. IEEE Communications Surveys Tutorials 15, 3 (2013), 1192–1209. DOI:http://dx.doi.org/10.1109/surv.2012.110112.00192

[5] A. Anjum and M.U. Ilyas. 2013. Activity recognition using smartphone sensors. 2013 IEEE 10th Consumer Communications and Networking Conference (CCNC) (2013). DOI:http://dx.doi.org/10.1109/ccnc.2013.6488584

[6] Hristijan Gjoreski et al. 2018. The University of Sussex-Huawei Locomotion and Transportation Dataset for Multimodal Analytics With Mobile Devices. IEEE Access 6 (2018), 42592–42604. DOI:http://dx.doi.org/10.1109/access.2018.2858933

[7] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. 2019. Enabling Reproducible Research in Sensor-Based Transportation Mode Recognition With the Sussex-Huawei Dataset. IEEE Access 7 (2019), 10870–10891. DOI:http://dx.doi.org/10.1109/access.2019.2890793

[8] Anon. Choosing the DB Instance Class. Retrieved December 14, 2019 from https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html