INSTITUTE FOR ADVANCED

COMPUTING AND

SOFTWARE

DEVELOPNMENT

AKURDI, PUNE

Documentation On

“Human Activity Recognition”

PG-DBDA SEP 2022

Submitted By:

Group No: 25

Vijay Pawar (229352)

Mandar Deokar (229319)

Dr. Shantanu Pathak
Project Guide

Mr. Rohit Puranik
Centre Coordinator

Contents:

Introduction

Problem Statement:

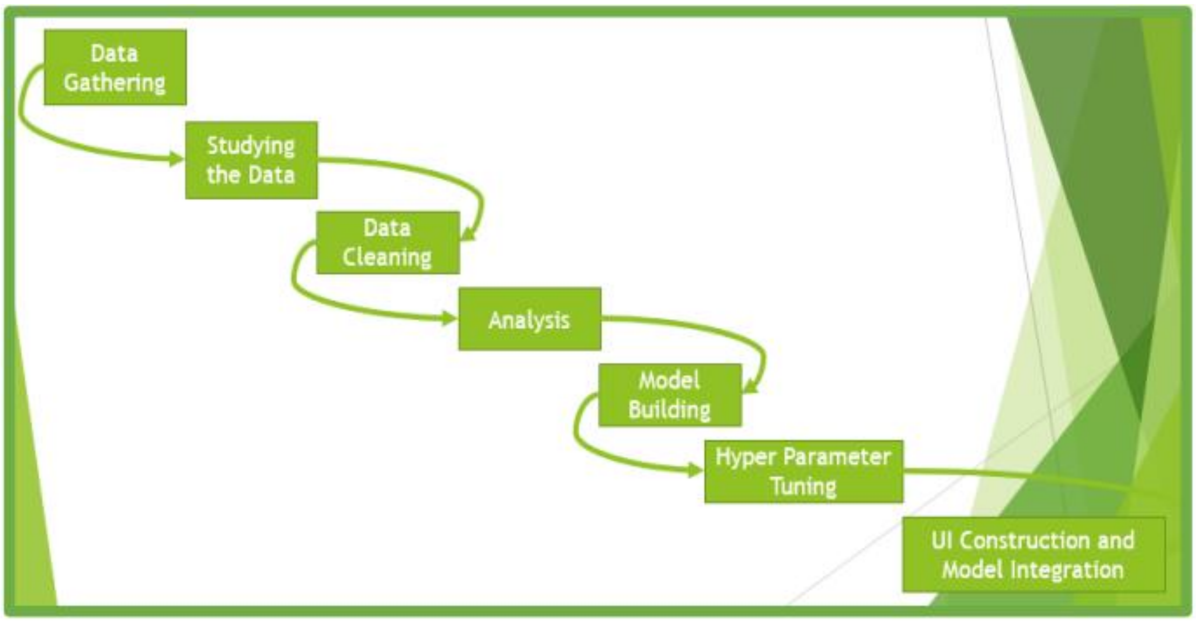
Recognition of Human Activity on the basis of sensor data obtained from smartphone and smart watch.

Abstract:

Members of the WISDM (Wireless Sensor Data Mining) Lab in the Department of Computer and Information Science of Fordham University collected data from the accelerometer and gyroscope sensors of a smartphone and smart watch as 51 subjects performed 18 diverse activities of daily living. Each activity was performed for 3 minutes, so that each subject contributed 54 minutes of data. These activities include basic ambulation-related activities (e.g., walking, jogging, climbing stairs), hand-based activities of daily living (e.g., brushing teeth, folding clothes), and various eating activities (eating pasta, easting chips). The data set contains the low level time-series sensor data from the phone's accelerometer, phone's gyroscope, watches' accelerometer, and watches' gyroscope. All of the time-series data is tagged not only with the activity that was being performed, but with a subject identifier, which means that the data be used for building and evaluating biometrics models, as well activity recognition models. Researcher in the WISDM Lab subsequently used a sliding window approach to transform the time-series data into labeled examples, and the scripts for performing the transformation, as well as the transformed data, are also provided with the data set. The data set is available from the UCI Machine Learning Repository as the "WISDM Smartphone and Smart watch Activity and Biometrics Dataset."

Workflow of Project:

The diagram below shows the workflow of this project.



OVERVIEW

The "WISDM Smartphone and Smart watch Activity and Biometrics Dataset" includes data collected from 51 subjects, each of whom were asked to perform 18 tasks for 3 minutes each. Each subject had a smart-watch placed on his/her dominant hand and a smartphone in their pocket. The data collection was controlled by a custom-made app that ran on the smartphone and smart watch. The sensor data that was collected was from the accelerometer *and* gyroscope on both the smartphone *and* smart watch, yielding four total sensors. The sensor data was collected at a rate of 20 Hz (i.e., every 50ms). The smartphone was either the Google Nexus 5/5X or Samsung Galaxy S5 running Android 6.0 . The smart watch was the LG G Watch running Android Wear 1.5. The general characteristics of the data and data collection process are summarized in

TABLE SUMMARY INFORMATION FOR THE DATASETS

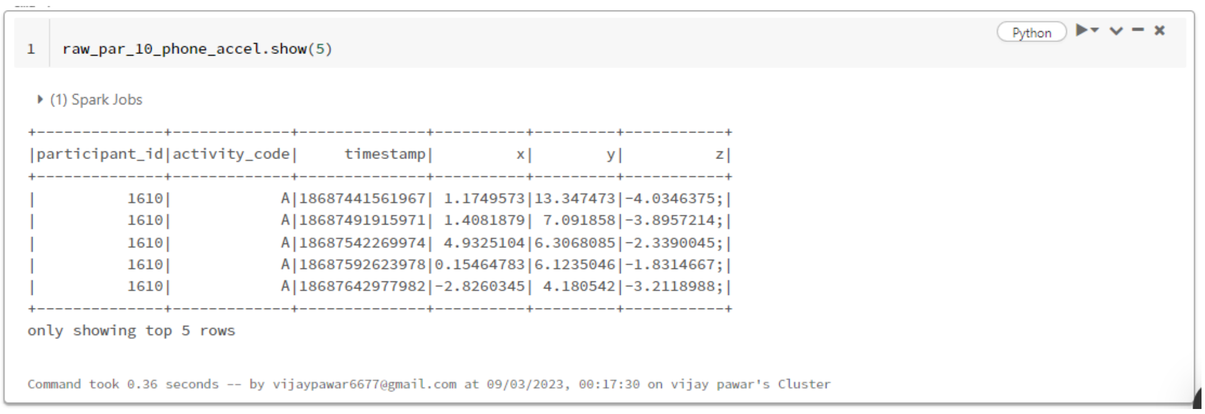
Number of subjects	51
Number of activities	18
Minutes collected per activity	3
Sensor polling rate	20Hz
Smartphone used	Google Nexus 5/5xor Samsung Galaxy S5
Smart watch used	LG G Watch
Number raw measurements	15,630,426

- **Exploratory Data Analysis:**

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

Following are some plots we used to extract some useful information.

The Dataset we have contain More than 70 lakh rows and 6 columns



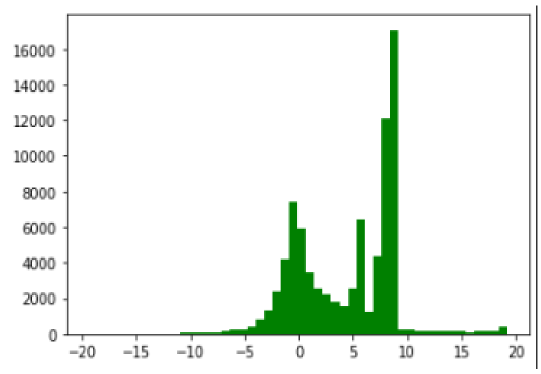
```
raw_par_10_phone_accel.show(5)
```

▸ (1) Spark Jobs

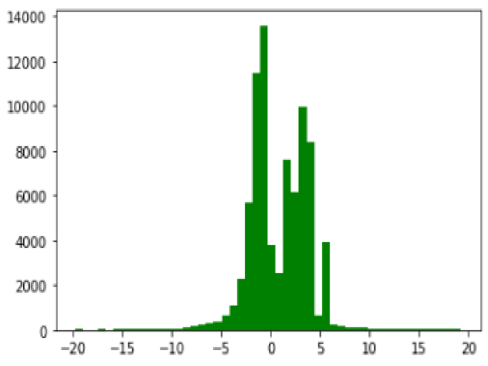
```
+--------------+-------------+--------------+----------+---------+-----------+
|participant_id|activity_code|     timestamp|         x|        y|          z|
+--------------+-------------+--------------+----------+---------+-----------+
|          1610|            A|18687441561967| 1.1749573|13.347473|-4.0346375;|
|          1610|            A|18687491915971| 1.4081879| 7.091858|-3.8957214;|
|          1610|            A|18687542269974| 4.9325104|6.3068085|-2.3390045;|
|          1610|            A|18687592623978|0.15464783|6.1235046|-1.8314667;|
|          1610|            A|18687642977982|-2.8260345| 4.180542|-3.2118988;|
+--------------+-------------+--------------+----------+---------+-----------+
only showing top 5 rows
```

Command took 0.36 seconds -- by vijaypawar6677@gmail.com at 09/03/2023, 00:17:30 on vijay pawar's Cluster

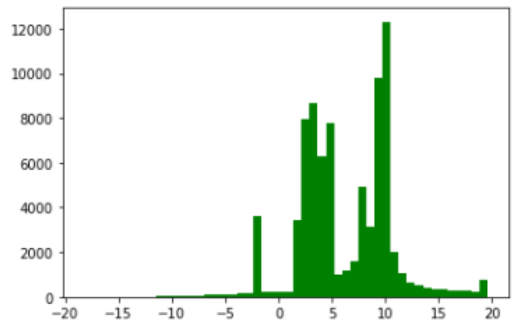
Histogram (*Accelometer*) :



(fig:distribution of x)

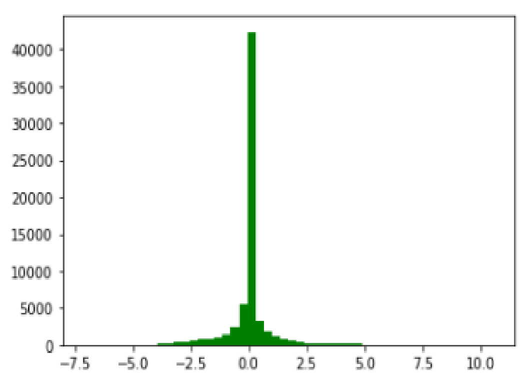


(fig:distribution of z)

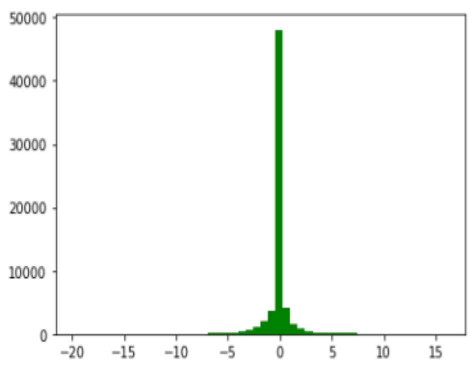


(fig:distribution of y)

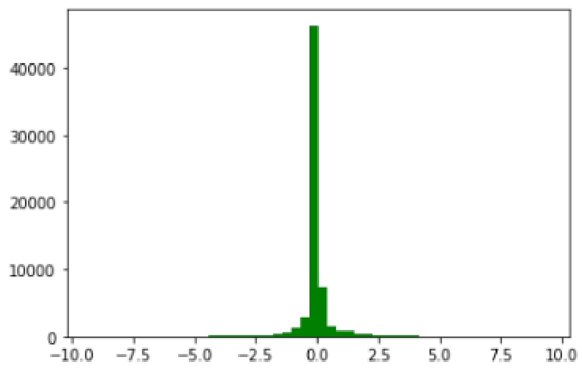
Histogram (*gyroscope*):



(fig:distribution of x)

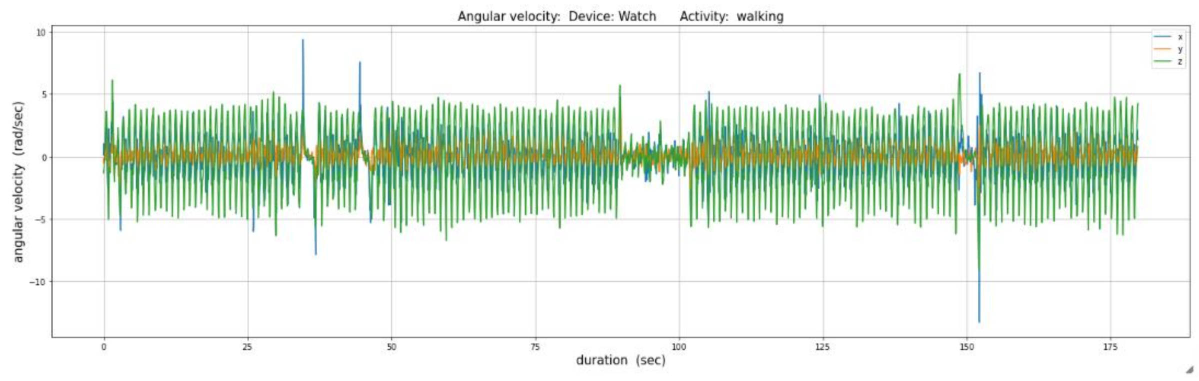


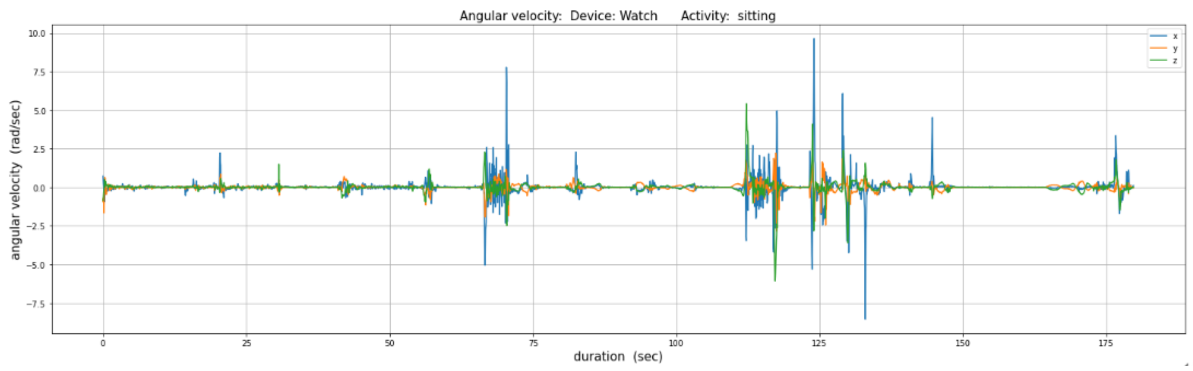
(fig:distribution of y)

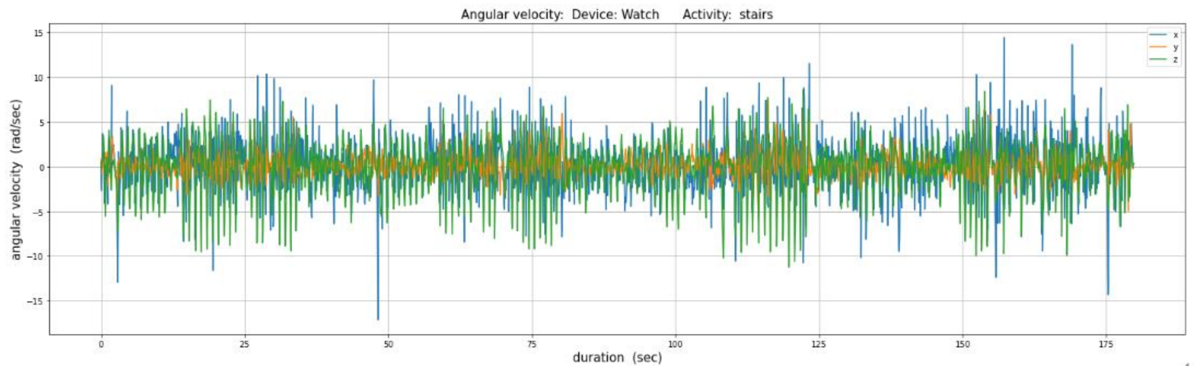


(fig:distribution of z)

Line Charts by Activity :







THE ACTIVITIES

Table 2 lists the 18 activities that were performed. The actual data files specify the activities using the code from Table 2. Similar activities are not necessarily grouped together (e.g., eating activities are not all together). This mapping can also be found at the top level of the data directory in *activity_key.txt*. In some of our research articles [3, 4] we partition these activities into 3 groupings to faciliate analysis, as follows:

Non-hand-oriented activities:
{walking, jogging, stairs, standing, kicking}

Hand-oriented activities (General):
{dribbling, playing catch, typing, writing, clapping, brushing teeth, folding clothes}

Hand-oriented activities (eating):
{eating-pasta, eating soup, eating sandwich, eating chips, drinking}

TABLE 2

Activity	**Code**
Walking	A
Jogging	B
Stairs	C
Sitting	D
Standing	E
Typing	F
Brushing Teeth	G
Eating Soup	H
Eating Chips	I
Eating Pasta	J
Drinking from Cup	K
Eating Sandwich	L
Kicking (Soccer Ball)	M
Playing Catch w/Tennis Ball	O

Dribblinlg (Basketball)	P
Writing	Q
Clapping	R
Folding Clothes	S

THE RAW TIME-SERIES SENSOR DATA

The raw time-series sensor data is recorded by the accelerometrer and gyrsocope on both the phone and watch at a rate of 20Hz. These sampling rates are just suggestions to the operating systems and polling of the sensors can be delayed if the processor is busy. Each sensor measurement that were cord is for one sensor on one device. Thus there are effectively four sensors, which we abbreviate as: phone-accel, phone-gyro, watch-accel, and watch-gyro. The data for each of the four sensors are recorded in different subdirectories, as shown visually in Figure 1.

Under the main data directory is a subdirectory called raw that contains all of the raw sensor data. The phone and watch sensor data are stored in separate subdirectories, and the accelerometer and gyroscope data for each device are stored in the corresponding two subdirectories.
Within each subdirectory there is a file per subject, so there will be 51 files in each of the four subdirectories. The files are all named using a standard naming convention with four components. Sample filenames are:
data_1600_accel_phone.txt
data_1634_gyro_watch.txt

The first component is fixed and is always "data_". The second component identifies the subject and var-ies from 1600 to 1650 (i.e., there are 51 subjects and labeling starts at 1600). The third component is either "accel" or "gyro" to identify the sensor, and the fourth component is "phone" or "watch" to identify the de-vice on which the sensor resides. All of these files end with ".txt" since they are text files. Note that technically the third and fourth components of the filenames

are not needed since the device and sensor type is implied by the directory structure. However it is convenient to have all of the necessary information en-coded in the filename.
Within each time-series sensor file is one sensor reading per line. The format of this line is identical across both devices and both types of sensors. The data on each line is comma separated and each line ends with a semicolon. The format of each line is as follows: *Subject-id, Activity Code, Timestamp, x, y, z;*
Table 3 defines each of the six features. The actual sensor values have different units for the different sensors. For the accelerometer sensor, the units are m/s2, while for the gyroscope sensor the units are radians/s. Note that the force of gravity on Earth, which affects the accelerometer readings, is 9.8m/s2.

Field name	
Subject-id	*Type:Symbolic numericidentifier.*Uniquely identifyiesthe subject. Range:1600-1650.
Activity code	*Type: Symbolic single letter.*Identifies aspecific activity as listed in Table 2. Range:A-S (no “N”value)
Timestamp	*Type: Integer.*Linux time
x	*Type Numeric: real.* Sensor value for x axis. May be positive or negative.
y	Same as x but for y axis
z	Same as x but for z axis

The actual number of lines of data per subdirectory is as follows:
- raw/phone/accel: 4,804,403
- raw/phone/gyro: 3,608,635
- raw/watch/accel: 3,777,046
- raw/watch/gyro: 3,440,342

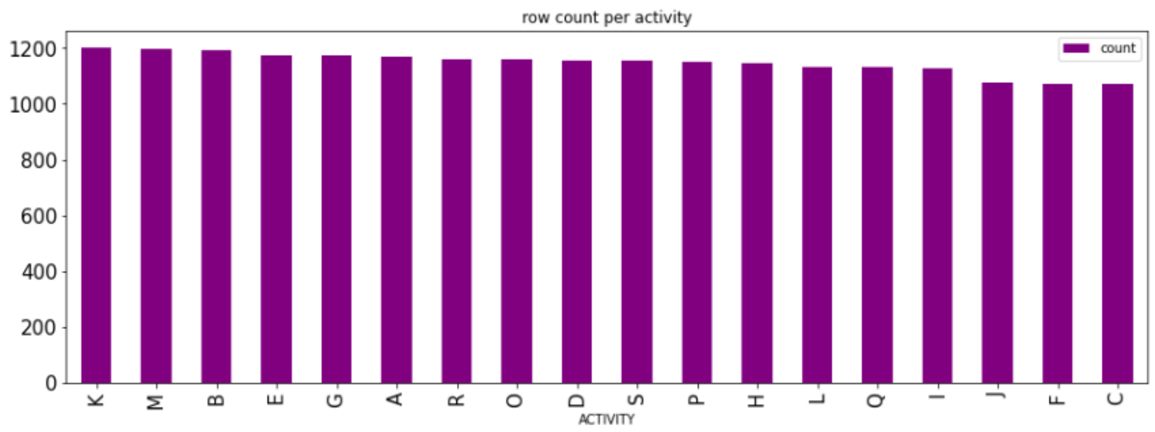
We are unsure exactly why there are so many more sensor readings for the phone accelerometer. Next, Table 4 shows the distribution of sensor readings by activity.

This command extracts the second field in the comma delimited files (the activity code), then sorts it, and then counts the number of entries for each unique value/code. The "Total" field in Table 4 is the sum of the values preceding it in each row, and the "Class %" is computed as this total value divided by 15,630,426, the total number of sensor readings. Thus "Class %" represents the class distribution for the activity class. Table 4 shows that the class distribution over these values is close to the ideal amount, which would be 5.55%(100/18).

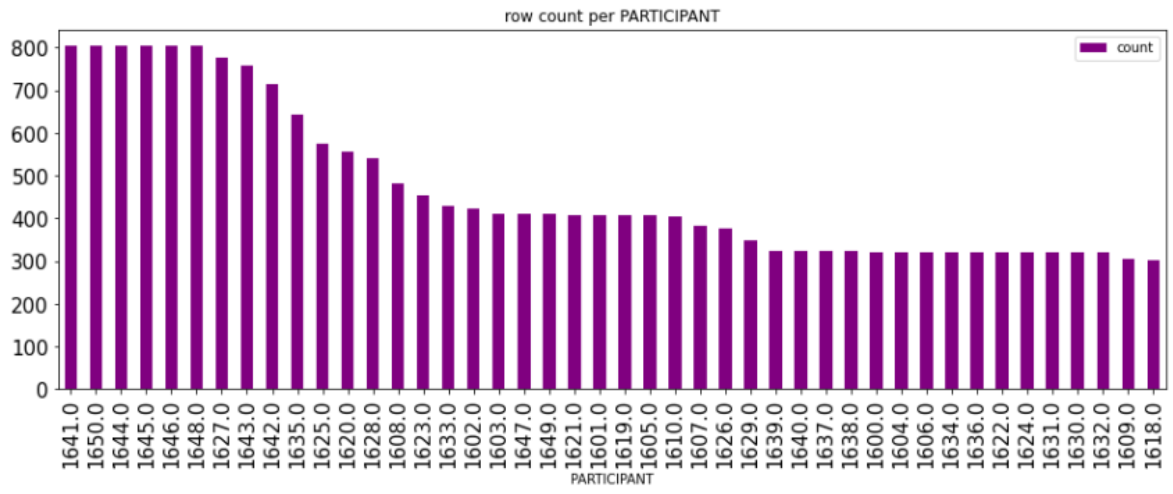
• THE TRANSFORMED ACTIVITY EXAMPLES

Activity recognition and biometric models can be built directly from the time-series data, but most ma-chine learning and data mining methods require the data be in the form of labelled examples and not labelled time series values. This data set includes labelled examples in addition to the raw time-series da-ta. There are many possible transformation processes that can be applied, and the process reflected in the examples associated with the data set reflects just one process. Therefore we view the raw time series data the main contribution of this data set, since re-searchers can use it to generate any higher level representation that they want.

The transformation process that was used to generate the labelled examples is described in Section 4.1, which also describes the features that are generated and the layout/format of each generated example. Section 4.2 then supplies some meta-data about the data, such as the number of examples generated and the distribution over the various activity values.



(Fig : No of Rows By Activity)



(Fig : No of Rows By participant)

- **The Data Transformation Process**

The basic data transformation process utilized to form the labelled examples provided with this data set has been used by our WISDM Lab since 2010 and has been used in many research papers--although normally on smaller data sets. The process was described in our first article on activity recognition [2], although the transformation process applied to generate the examples in this data set include some additional features.
The raw time series data for each sensor (per subject and per activity) is divided into 10-second non-overlapping segments and then high-

level features are generated based on the 200 (10s □ 20 readings/s) readings contained within each segment. A 10-second window was chosen because we felt that it provided sufficient time to capture several repetitions of those actions that involve reptious movements, and still is small enough to provide quick response (i.e., a pre-diction every 10 seconds).

The transformation process described in our prior research yields 43 features excluding the activity label, or 44 features with the activity label. However, the examples included in this data set have 93 features (with the label), because some experimentation was done and additional features were added, but never used in published research. For completeness we described the 93 features, but identify the 49 features that were not used in any published papers.

Our transformed examples are placed into ARFF (Attribute-Relation File Format) files, which is a file format specified by the WEKA suite of data mining tools [1]. ARFF files contain both formatting information for the data and then the data itself. The format-ting information specifies information about all of the attributes, so we will use the ARFF header to introduce and describe the features generated by the transformation process. The start of every ARFF file will have the same header information, with one exception to be

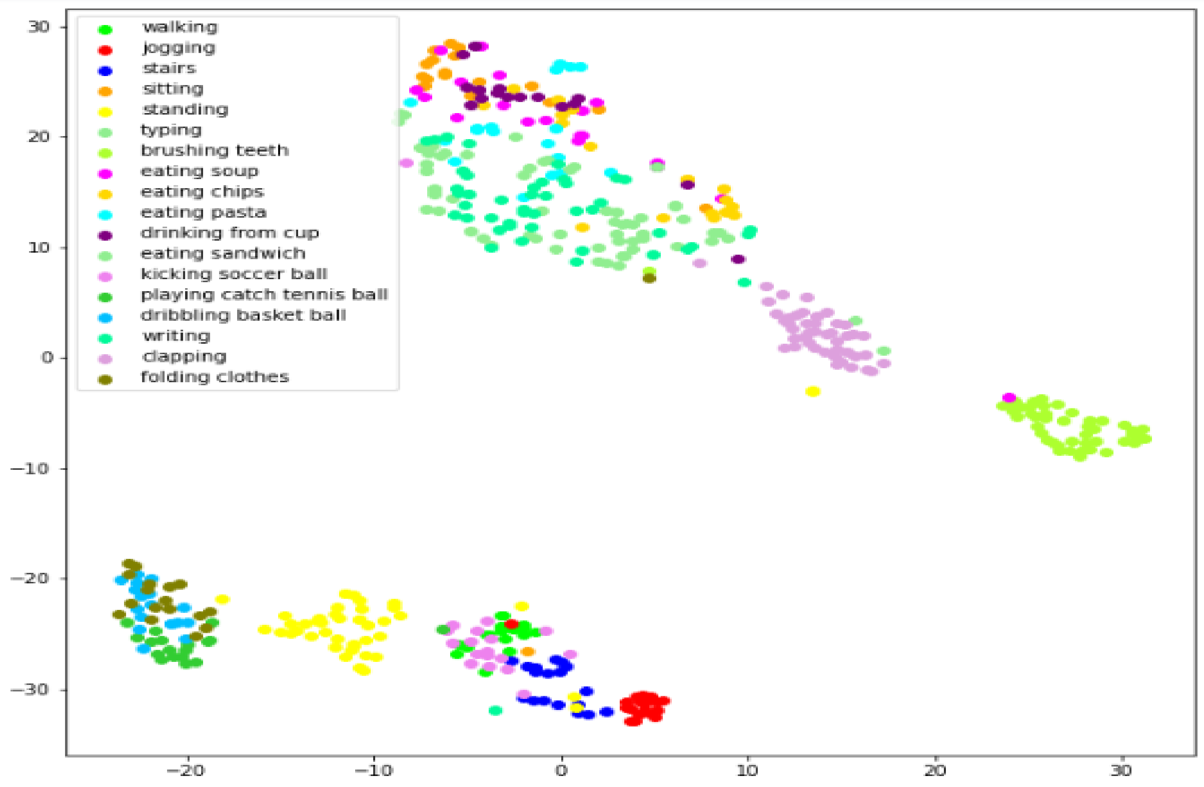
which defines the relation, and this line will always be followed by a blank line:

The remainder of the header, starting with line 3, is described by Table 5. The header goes from line 3 to line 95, with one attribute per line, and thus we get 93 attributes/features. Every line in the header from line 3 to line 95 will begin with "@attribute". The first column in Table 5 specifies the corresponding line number in the ARFF file. The second column specifies the name of the attribute, which follows the "@attribute" text that starts each line. The attribute name is always included in double quotes. The third column specifies the attribute type, or, in the case of a categorical feature, a list of all possible feature values. This information follows the attribute name on the line. Some of the attributes/features have many variants (e.g., several have a value per axis so come in multiples of three), so in some cases the table entries are compressed into a single row (but with multiple line numbers in the first column). For example, feature X0 appears on line 4, X1 on line 4, and X9 on line 13. Similarly, XAVG is on line 34, YAVG on line 35, and ZAVG on line 36.

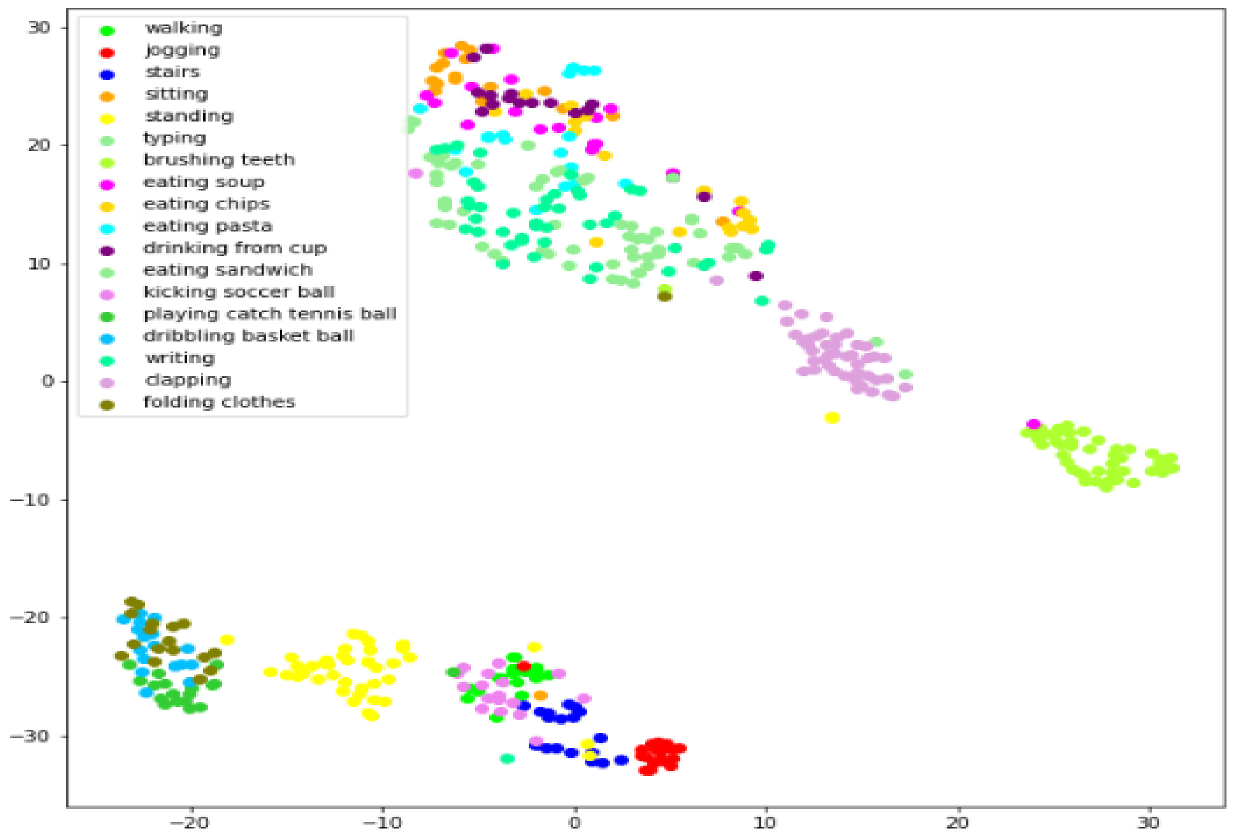
The 49 features in Table 5 that are denoted with an asterisk (*) and highlighted in bold are not used in any of our published research. Descriptions of all of the features are provided following Table 5. Recall that each of the features are based on the sensor readings of one sensor over a 10 second window.

LAYOUT OF ARFFHEADER FILE

Line #	Attribute Name	Attribute Type or Values
3	ACTIVITY	{A,B,C,D,E,F,G,H,I,J,K,L,M,O,P,Q,R,S}
4-13	X{0-9}	numeric
14-23	Y{0-9}	numeric
24-33	Z{0-9}	numeric
34-36	{X,Y,Z}AVG	numeric
37-39	{X,Y,Z}PEAK	numeric
40-42	{X,Y,Z}ABSOLDEV	numeric
43-45	{X,Y,Z}STANDDEV	numeric
46-48	**{X,Y,Z}VAR***	numeric
49-61	**XMFCC{0-12}***	numeric
62-77	**YMFCC{0-12}***	numeric
75-87	**ZMFCC{0-12}***	numeric
88-90	**{XY, XZ, YZ}COS***	numeric
91-93	**{XY, XZ, YZ}COR***	numeric
94	RESULTANT	numeric
95	**class***	{16XX}



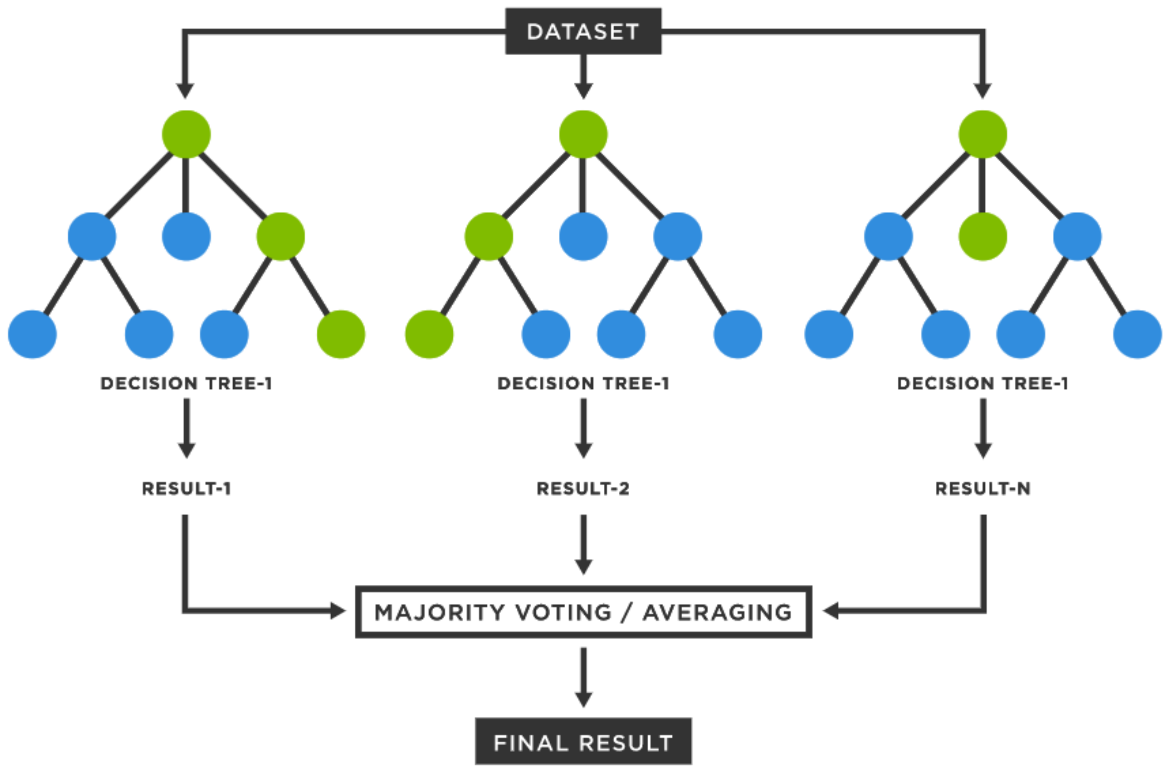
(fig: cluster diagram of user 20)

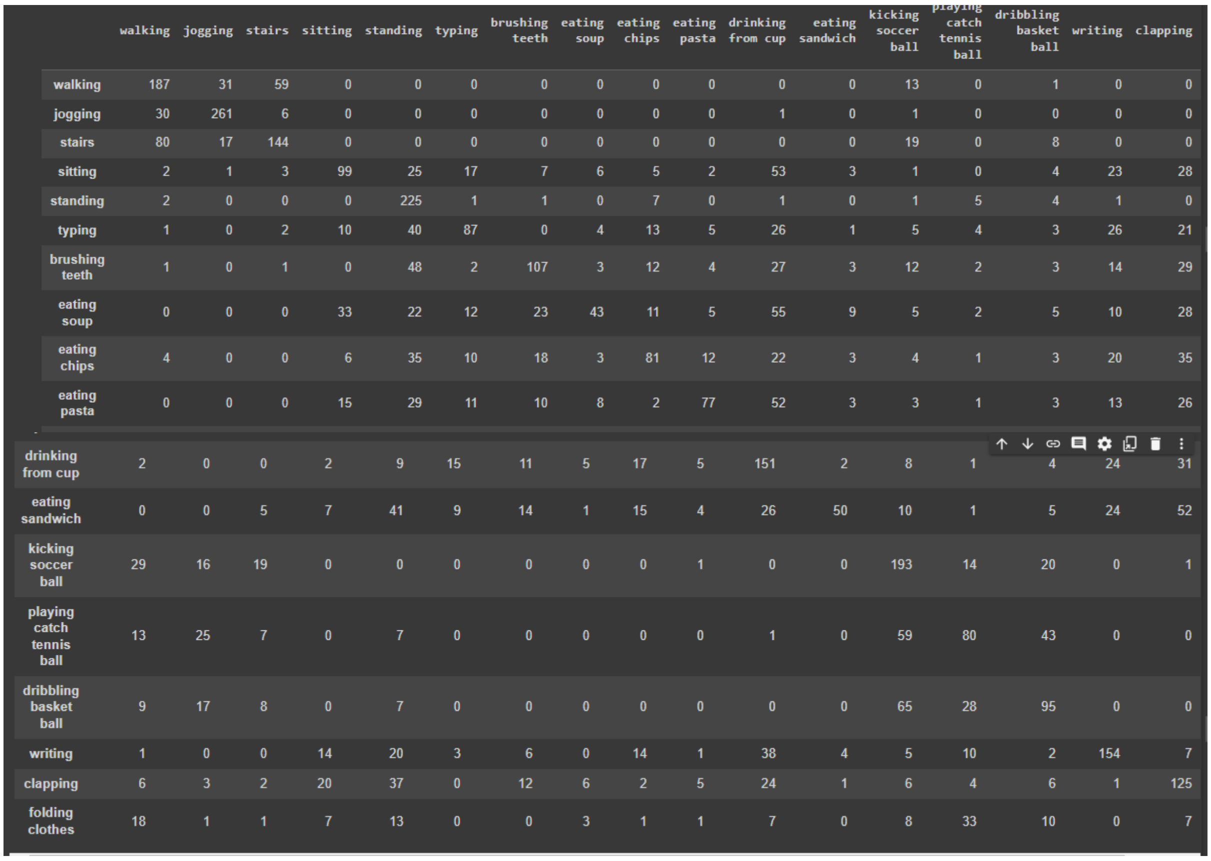


(fig: cluster diagram of user 35)

Random Forest :

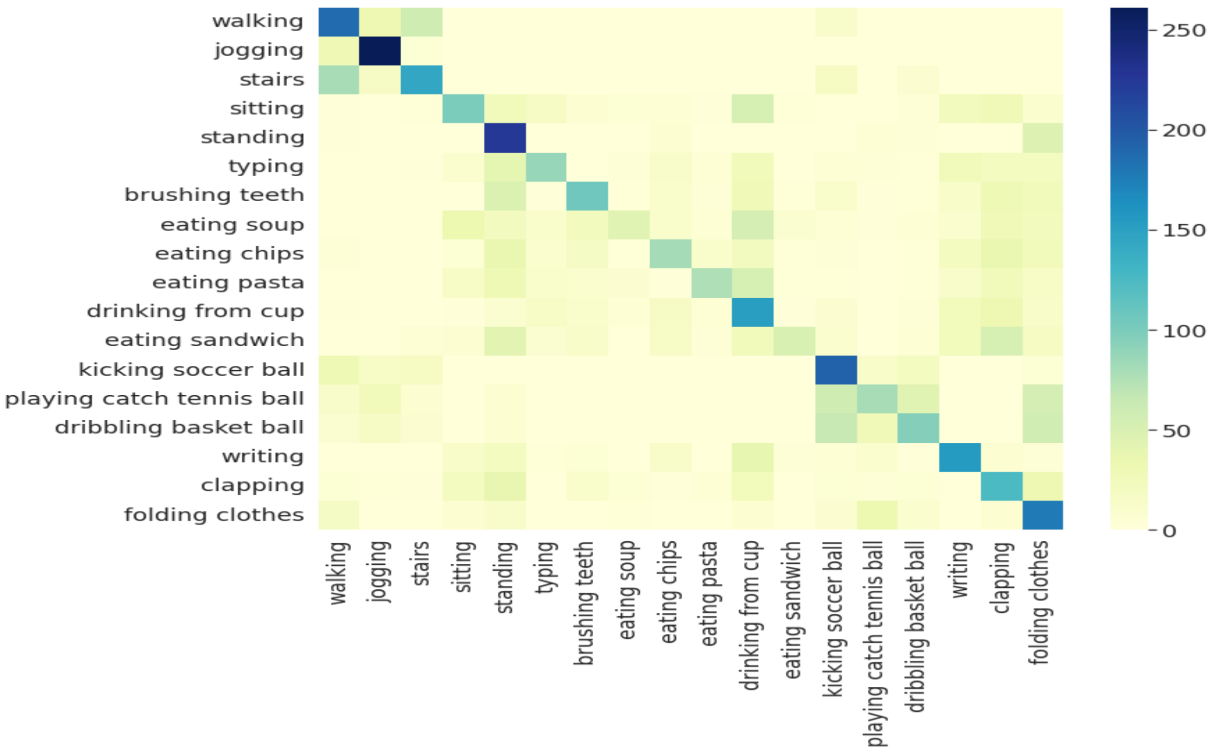
A random forest is a supervised machine learning algorithm that is constructed from decision tree algorithms. This algorithm is applied in various industries such as banking and e-commerce to predict behavior and outcomes.



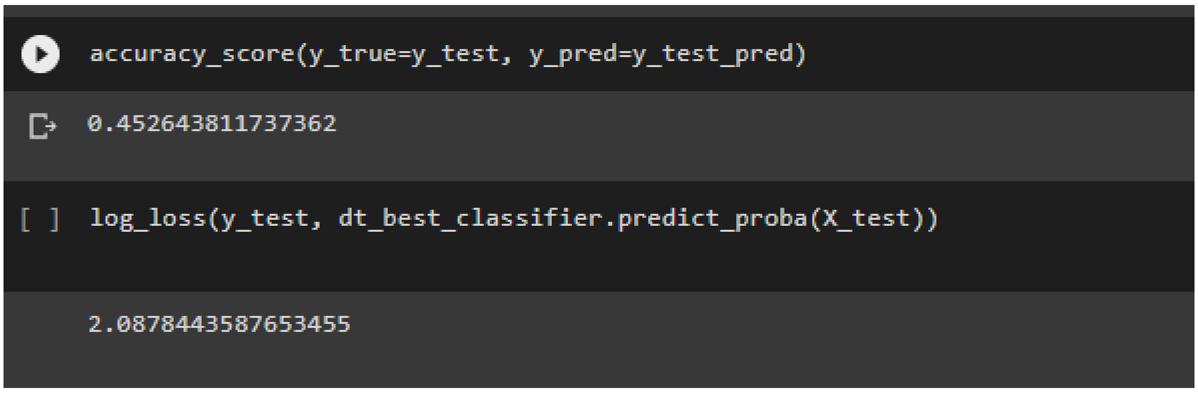


	walking	jogging	stairs	sitting	standing	typing	brushing teeth	eating soup	eating chips	eating pasta	drinking from cup	eating sandwich	kicking soccer ball	playing catch tennis ball	dribbling basket ball	writing	clapping
walking	187	31	59	0	0	0	0	0	0	0	0	0	13	0	1	0	0
jogging	30	261	6	0	0	0	0	0	0	0	1	0	1	0	0	0	0
stairs	80	17	144	0	0	0	0	0	0	0	0	0	19	0	8	0	0
sitting	2	1	3	99	25	17	7	6	5	2	53	3	1	0	4	23	28
standing	2	0	0	0	225	1	1	0	7	0	1	0	1	5	4	1	0
typing	1	0	2	10	40	87	0	4	13	5	26	1	5	4	3	26	21
brushing teeth	1	0	1	0	48	2	107	3	12	4	27	3	12	2	3	14	29
eating soup	0	0	0	33	22	12	23	43	11	5	55	9	5	2	5	10	28
eating chips	4	0	0	6	35	10	18	3	81	12	22	3	4	1	3	20	35
eating pasta	0	0	0	15	29	11	10	8	2	77	52	3	3	1	3	13	26
drinking from cup	2	0	0	2	9	15	11	5	17	5	151	2	8	1	4	24	31
eating sandwich	0	0	5	7	41	9	14	1	15	4	26	50	10	1	5	24	52
kicking soccer ball	29	16	19	0	0	0	0	0	0	1	0	0	193	14	20	0	1
playing catch tennis ball	13	25	7	0	7	0	0	0	0	0	1	0	59	80	43	0	0
dribbling basket ball	9	17	8	0	7	0	0	0	0	0	0	0	65	28	95	0	0
writing	1	0	0	14	20	3	6	0	14	1	38	4	5	10	2	154	7
clapping	6	3	2	20	37	0	12	6	2	5	24	1	6	4	6	1	125
folding clothes	18	1	1	7	13	0	0	3	1	1	7	0	8	33	10	0	7

(fig: Confusion Matrix)



(fig:HeatMap)



```
accuracy_score(y_true=y_test, y_pred=y_test_pred)

0.452643811737362

[ ] log_loss(y_test, dt_best_classifier.predict_proba(X_test))

2.0878443587653455
```

Accuracy for Random Forest – 0.45

K nearest neighbor:

K Nearest Neighbor Algorithm. K nearest neighbor algorithm is very simple. It works based on minimum distance from the query instance to the training samples to determine the K-nearest neighbors. The data for KNN algorithm consist of several multivariate attributes name that will be used to classify.

k' in KNN is a parameter that refers to the number of nearest neighbors to include in the majority of the voting process.

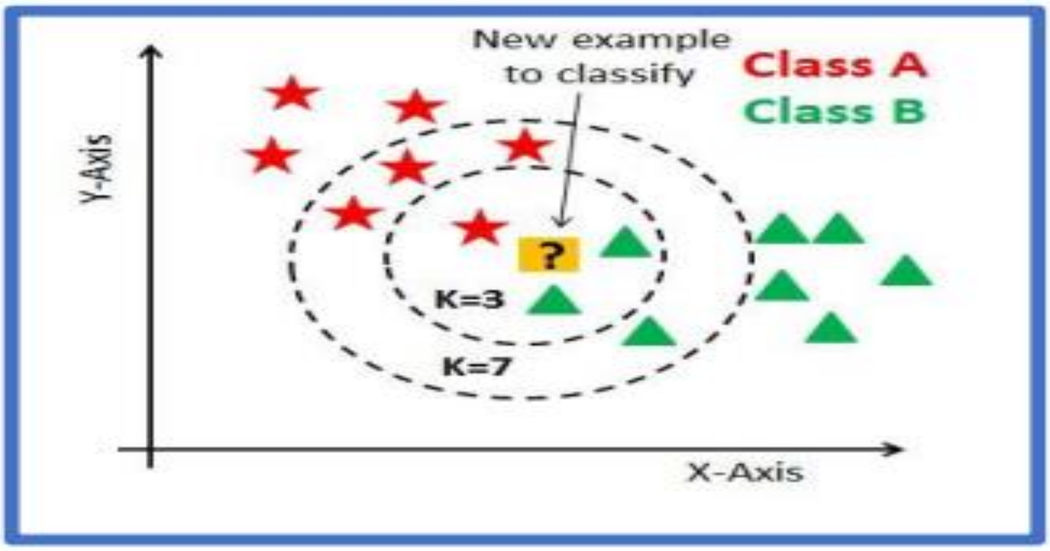
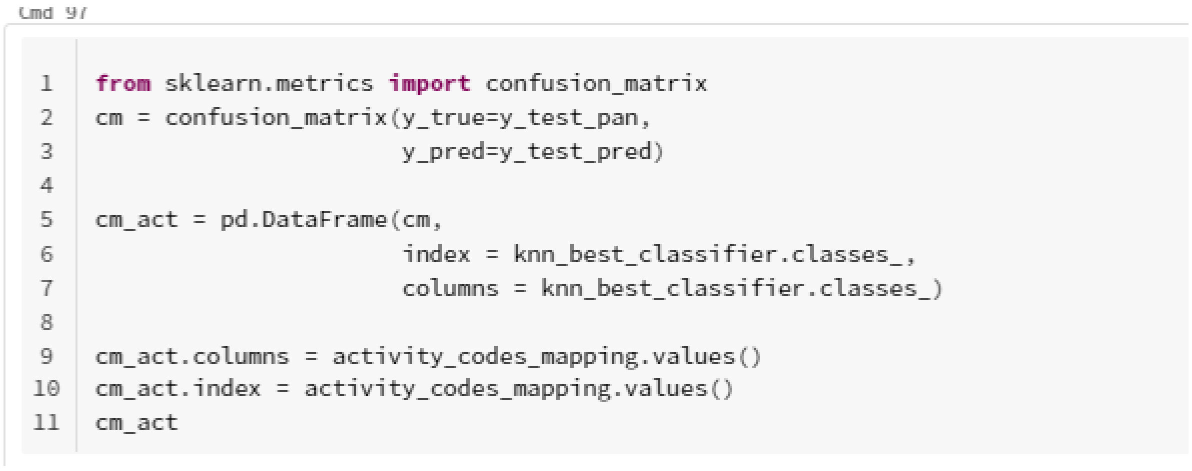
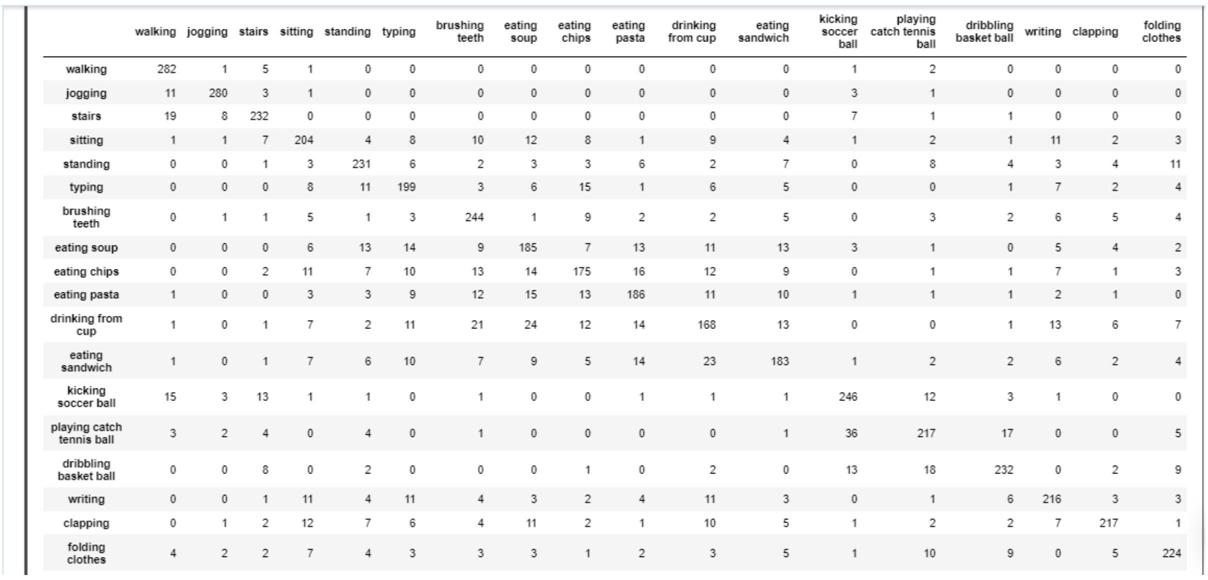


Figure 14 Example of KNN.



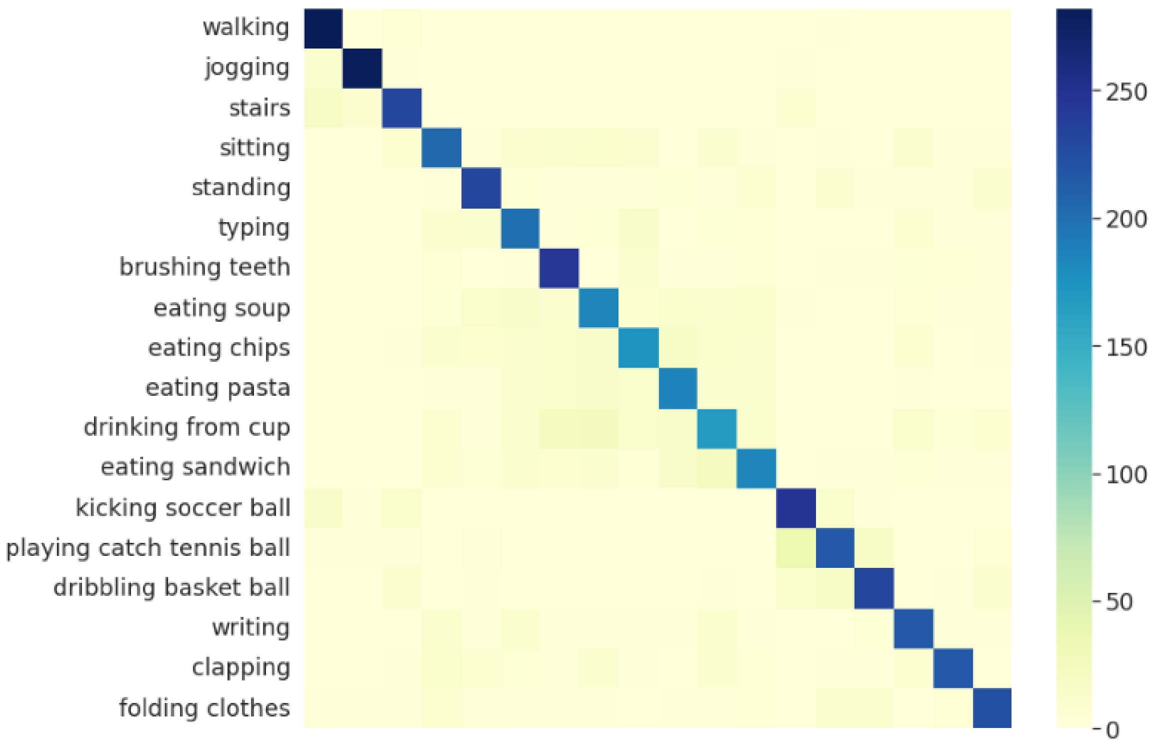
Cmd 97

```python
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_true=y_test_pan,
                      y_pred=y_test_pred)

cm_act = pd.DataFrame(cm,
                      index = knn_best_classifier.classes_,
                      columns = knn_best_classifier.classes_)

cm_act.columns = activity_codes_mapping.values()
cm_act.index = activity_codes_mapping.values()
cm_act
```

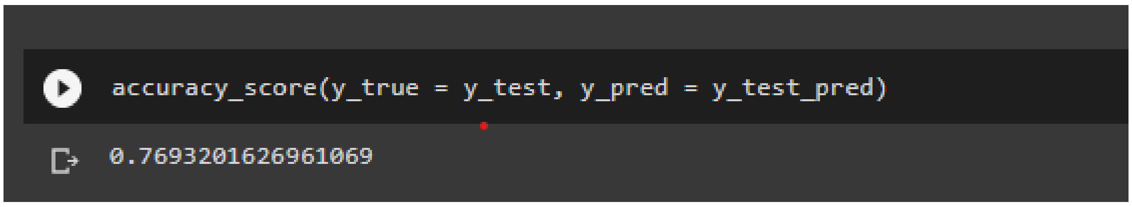


	walking	jogging	stairs	sitting	standing	typing	brushing teeth	eating soup	eating chips	eating pasta	drinking from cup	eating sandwich	kicking soccer ball	playing catch tennis ball	dribbling basket ball	writing	clapping	folding clothes
walking	282	1	5	1	0	0	0	0	0	0	0	0	1	2	0	0	0	0
jogging	11	280	3	1	0	0	0	0	0	0	0	0	3	1	0	0	0	0
stairs	19	8	232	0	0	0	0	0	0	0	0	0	7	1	1	0	0	0
sitting	1	1	7	204	4	8	10	12	8	1	9	4	1	2	1	11	2	3
standing	0	0	1	3	231	6	2	3	3	6	2	7	0	8	4	3	4	11
typing	0	0	0	8	11	199	3	6	15	1	6	5	0	0	1	7	2	4
brushing teeth	0	1	1	5	1	3	244	1	9	2	2	5	0	3	2	6	5	4
eating soup	0	0	0	6	13	14	9	185	7	13	11	13	3	1	0	5	4	2
eating chips	0	0	2	11	7	10	13	14	175	16	12	9	0	1	1	7	1	3
eating pasta	1	0	0	3	3	9	12	15	13	186	11	10	1	1	1	2	1	0
drinking from cup	1	0	1	7	2	11	21	24	12	14	168	13	0	0	1	13	6	7
eating sandwich	1	0	1	7	6	10	7	9	5	14	23	183	1	2	2	6	2	4
kicking soccer ball	15	3	13	1	1	0	1	0	0	1	1	1	246	12	3	1	0	0
playing catch tennis ball	3	2	4	0	4	0	1	0	0	0	0	1	36	217	17	0	0	5
dribbling basket ball	0	0	8	0	2	0	0	0	1	0	2	0	13	18	232	0	2	9
writing	0	0	1	11	4	11	4	3	2	4	11	3	0	1	6	216	3	3
clapping	0	1	2	12	7	6	4	11	2	1	10	5	1	2	2	7	217	1
folding clothes	4	2	2	7	4	3	3	3	1	2	3	5	1	10	9	0	5	224

(fig: Confusion Matrix)



(fig:HeatMap)

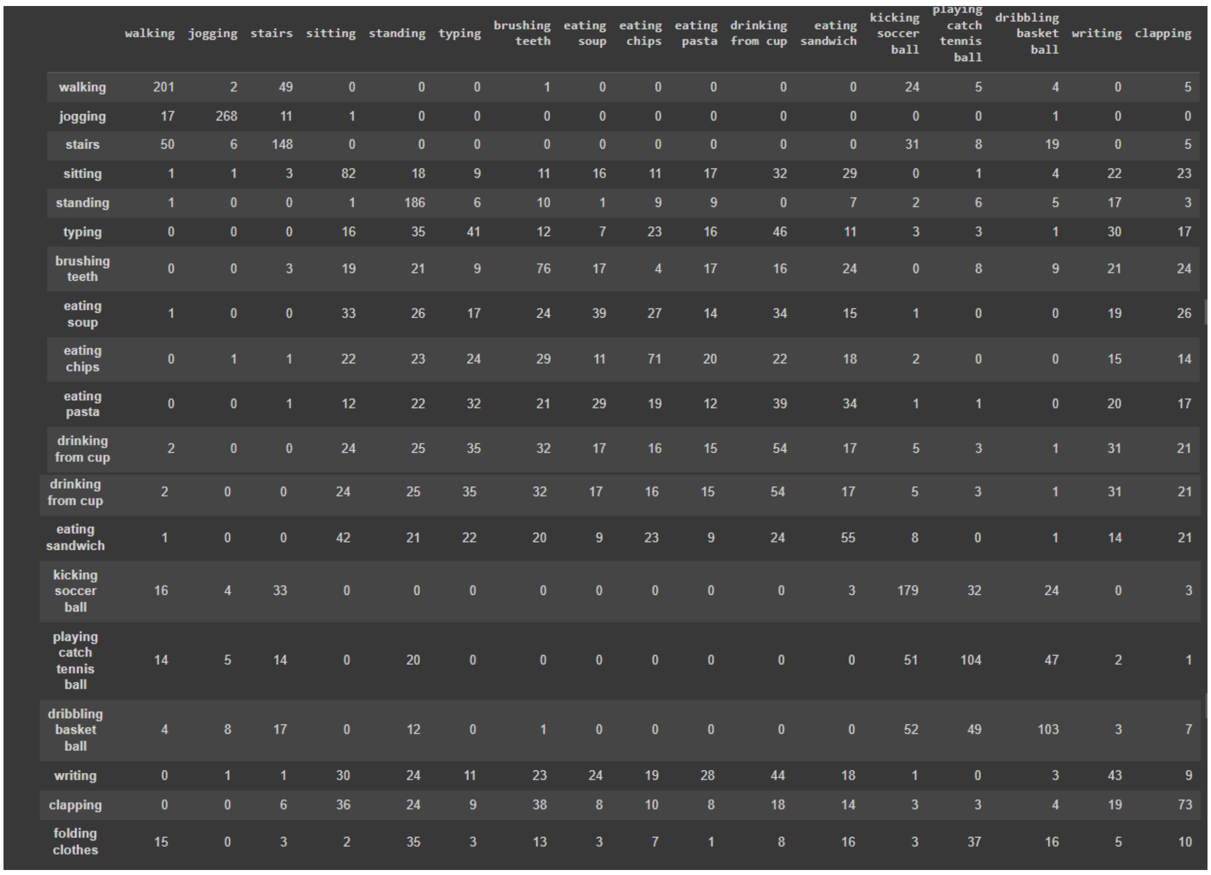


```
accuracy_score(y_true = y_test, y_pred = y_test_pred)
0.7693201626961069
```

Accuracy for KNN – 0.76

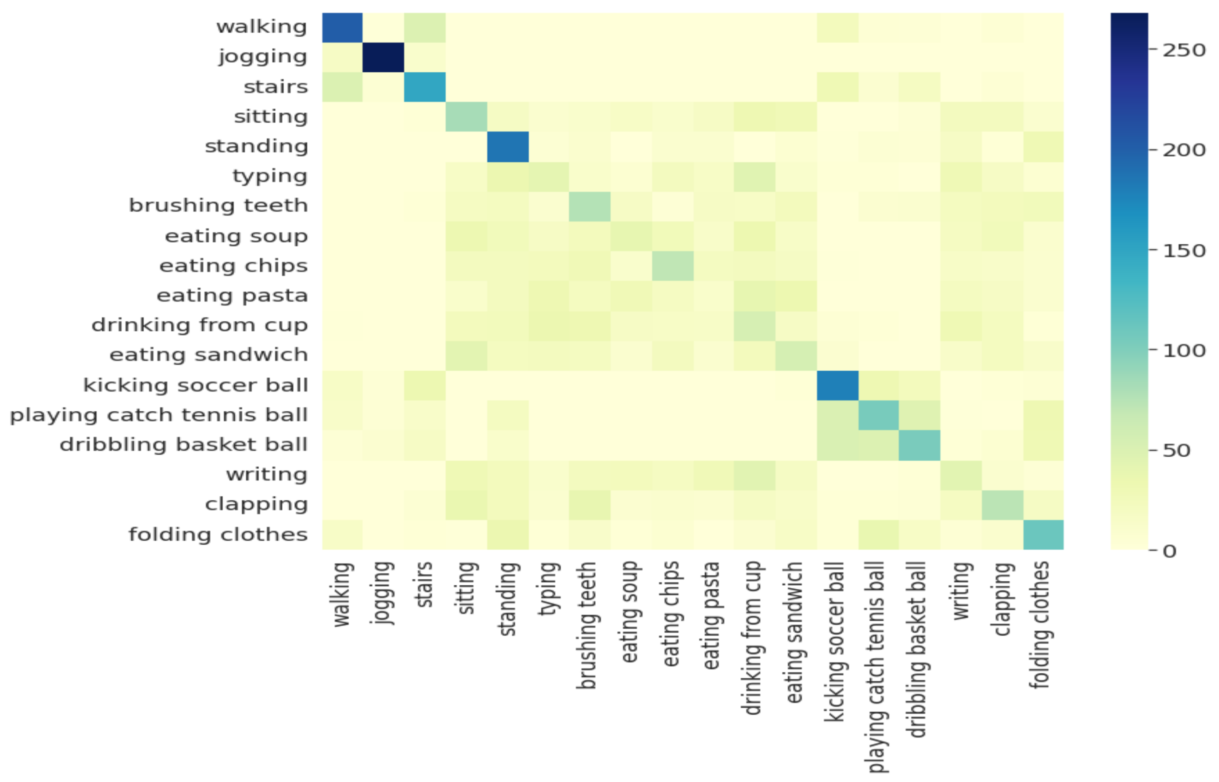
Logistic Regression :

Logistic regression is a process of modeling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes. Logistic regression is a useful analysis method for classification problems, where you are trying to determine if a new sample fits best into a category. As aspects of cyber security are classification problems, such as attack detection, logistic regression is a useful analytic technique.

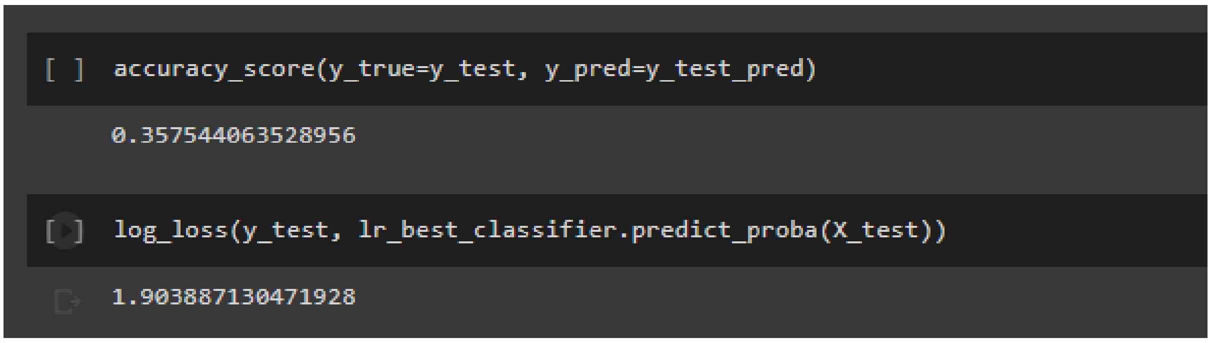


	walking	jogging	stairs	sitting	standing	typing	brushing teeth	eating soup	eating chips	eating pasta	drinking from cup	eating sandwich	kicking soccer ball	playing catch tennis ball	dribbling basket ball	writing	clapping
walking	201	2	49	0	0	0	1	0	0	0	0	0	24	5	4	0	5
jogging	17	268	11	1	0	0	0	0	0	0	0	0	0	0	1	0	0
stairs	50	6	148	0	0	0	0	0	0	0	0	0	31	8	19	0	5
sitting	1	1	3	82	18	9	11	16	11	17	32	29	0	1	4	22	23
standing	1	0	0	1	186	6	10	1	9	9	0	7	2	6	5	17	3
typing	0	0	0	16	35	41	12	7	23	16	46	11	3	3	1	30	17
brushing teeth	0	0	3	19	21	9	76	17	4	17	16	24	0	8	9	21	24
eating soup	1	0	0	33	26	17	24	39	27	14	34	15	1	0	0	19	26
eating chips	0	1	1	22	23	24	29	11	71	20	22	18	2	0	0	15	14
eating pasta	0	0	1	12	22	32	21	29	19	12	39	34	1	1	0	20	17
drinking from cup	2	0	0	24	25	35	32	17	16	15	54	17	5	3	1	31	21
drinking from cup	2	0	0	24	25	35	32	17	16	15	54	17	5	3	1	31	21
eating sandwich	1	0	0	42	21	22	20	9	23	9	24	55	8	0	1	14	21
kicking soccer ball	16	4	33	0	0	0	0	0	0	0	0	3	179	32	24	0	3
playing catch tennis ball	14	5	14	0	20	0	0	0	0	0	0	0	51	104	47	2	1
dribbling basket ball	4	8	17	0	12	0	1	0	0	0	0	0	52	49	103	3	7
writing	0	1	1	30	24	11	23	24	19	28	44	18	1	0	3	43	9
clapping	0	0	6	36	24	9	38	8	10	8	18	14	3	3	4	19	73
folding clothes	15	0	3	2	35	3	13	3	7	1	8	16	3	37	16	5	10

(fig: Confusion Matrix)



(fig:HeatMap)

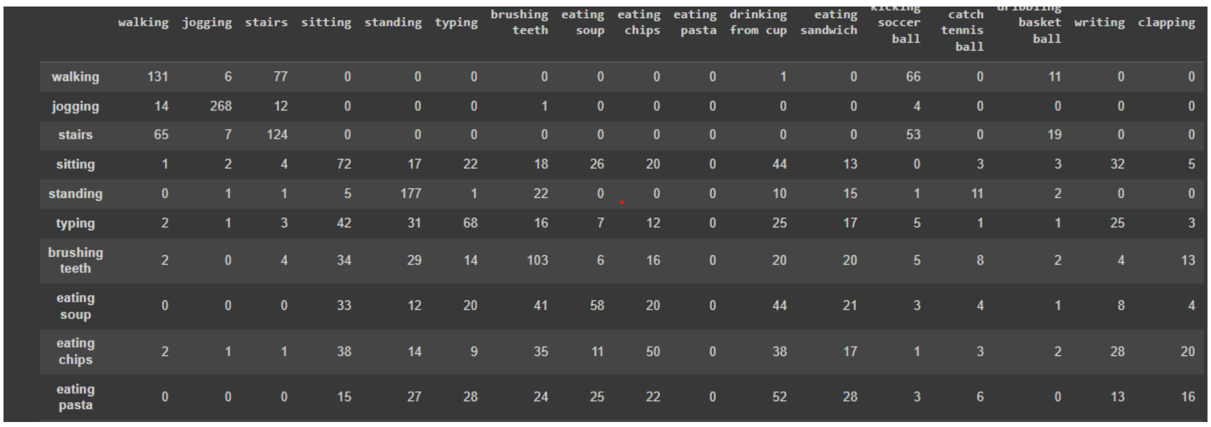
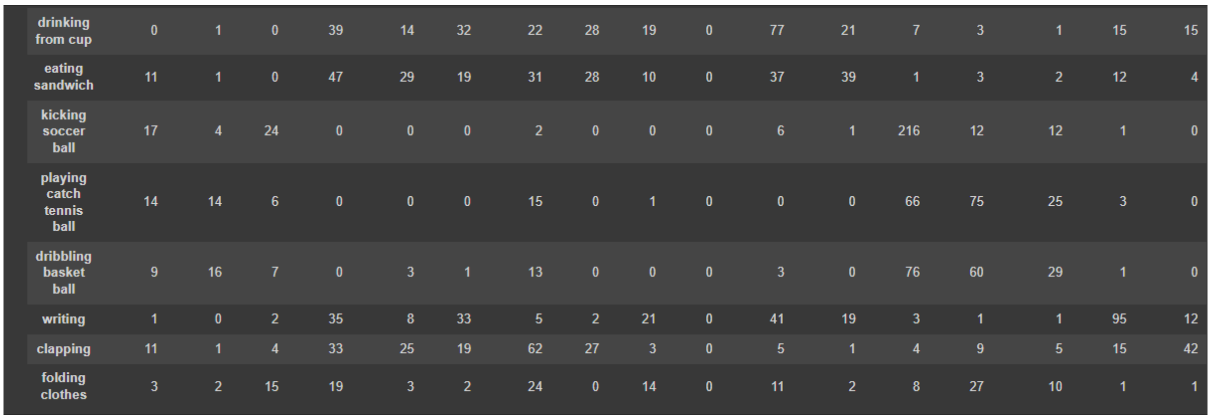


```
[ ]  accuracy_score(y_true=y_test, y_pred=y_test_pred)

     0.357544063528956

[ ]  log_loss(y_test, lr_best_classifier.predict_proba(X_test))

     1.903887130471928
```

Accuracy for Logistic Regression– 0.35

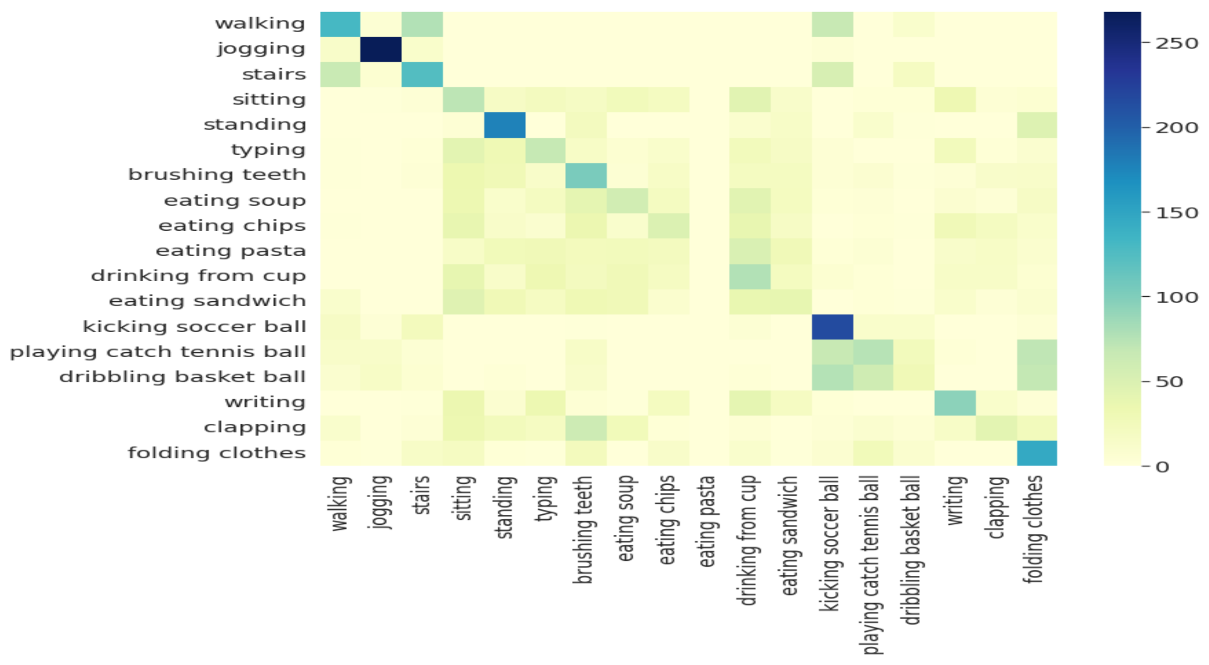
Decision Tree :

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

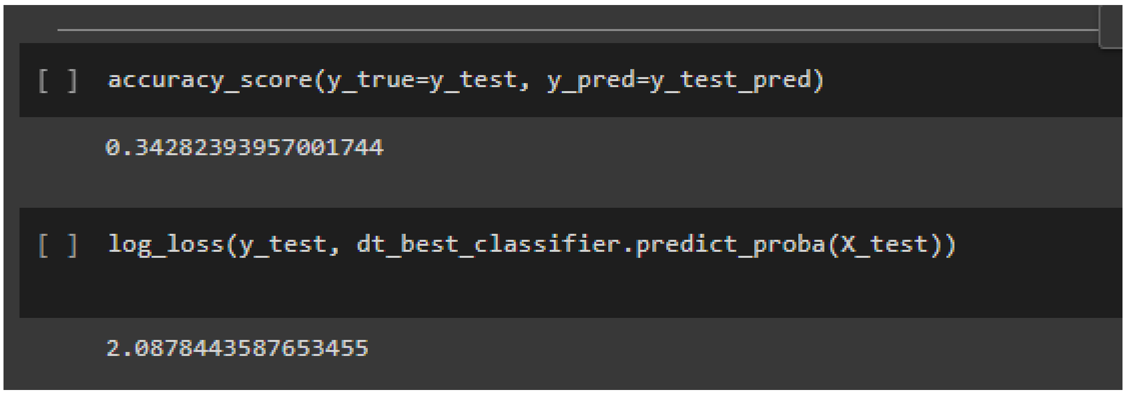



	walking	jogging	stairs	sitting	standing	typing	brushing teeth	eating soup	eating chips	eating pasta	drinking from cup	eating sandwich	kicking soccer ball	catch tennis ball	dribbling basket ball	writing	clapping
walking	131	6	77	0	0	0	0	0	0	0	1	0	66	0	11	0	0
jogging	14	268	12	0	0	0	1	0	0	0	0	0	4	0	0	0	0
stairs	65	7	124	0	0	0	0	0	0	0	0	0	53	0	19	0	0
sitting	1	2	4	72	17	22	18	26	20	0	44	13	0	3	3	32	5
standing	0	1	1	5	177	1	22	0	0	0	10	15	1	11	2	0	0
typing	2	1	3	42	31	68	16	7	12	0	25	17	5	1	1	25	3
brushing teeth	2	0	4	34	29	14	103	6	16	0	20	20	5	8	2	4	13
eating soup	0	0	0	33	12	20	41	58	20	0	44	21	3	4	1	8	4
eating chips	2	1	1	38	14	9	35	11	50	0	38	17	1	3	2	28	20
eating pasta	0	0	0	15	27	28	24	25	22	0	52	28	3	6	0	13	16
drinking from cup	0	1	0	39	14	32	22	28	19	0	77	21	7	3	1	15	15
eating sandwich	11	1	0	47	29	19	31	28	10	0	37	39	1	3	2	12	4
kicking soccer ball	17	4	24	0	0	0	2	0	0	0	6	1	216	12	12	1	0
playing catch tennis ball	14	14	6	0	0	0	15	0	1	0	0	0	66	75	25	3	0
dribbling basket ball	9	16	7	0	3	1	13	0	0	0	3	0	76	60	29	1	0
writing	1	0	2	35	8	33	5	2	21	0	41	19	3	1	1	95	12
clapping	11	1	4	33	25	19	62	27	3	0	5	1	4	9	5	15	42
folding clothes	3	2	15	19	3	2	24	0	14	0	11	2	8	27	10	1	1

(fig: Confusion Matrix)



(fig:Confusion Matrix)



```
[ ]  accuracy_score(y_true=y_test, y_pred=y_test_pred)

     0.34282393957001744

[ ]  log_loss(y_test, dt_best_classifier.predict_proba(X_test))

     2.0878443587653455
```

Accuracy for Decision Tree : – 0.34

```
Decision Tree: 0.33

Random Forest: 0.44

Logistic Regression: 0.38

KNN: 0.77
```

Insights & Conclusions:

1) The K Nearest Neighbor classifier has prooved to provide substantial higher prediction accuracy than the rest of the models (overall mean accuracy ~0.77 on test set) in this case.

2) The analysis demonstrated typical differentiation of detection accuracy accross the various physical activities:

The highest detection accuracy is for Jogging: ~0.95 walking, stairs, kicking soccer ball, dribbling basket ball also have high accuracies: ~0.8 On the other hand, the eating related activities have lower detection accuracies: ~0.6 However, this lower accuracy is still surprisingly significant ! (10 times the neutral guess accuracy, 1/18), considering the fact that the smartphone was placed inside the participant's pants leg pocket !

3) The accuracy profiles are consistent with the visual clustering patterns seen in the three dimention features visualization plots & t-SNE, in particular, the jogging points clusters where exceptionally noticable and distinct.

4) Possible explanations of the above observations:

The Jogging etc. where highly predictable because they where the most intense and strongly related to leg movement (the phone was inside the leg pocket)

The suprisingly significant accuracy rates of identifying eatings activites by the phone might be explained by the posture sitting changes characterizing such activities:

The fact that the accelerometer measures also the gravity vector, could supply additional implied information pertaining the orientation of the device and, therefore, as to the posture mode (standing/sitting) of the participant, which could contribute to the accuracy detection ability.

References

He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-
12-
10). "Deep Residual Learning for Image Recognition".
2. Srivastava, Rupesh Kumar; Greff, Klaus; Schmidhuber, Jürgen (2015-05-02). "Highway Networks".
3. Huang, Gao; Liu, Zhuang; Weinberger, Kilian Q.; van der Maaten,
Laurens (2016-08-24). "Densely Connected Convolutional
Networks".
4. S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici,B. Varadarajan, and S. Vijayanarasimhan. YouTube-8M: A large-scale
video classification benchmark. arXiv preprint, arXiv:1609.08675,
2016.
5. Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation
with pseudo-3d residual networks. In Proceedings
of the International Conference on Computer Vision (ICCV),
2017.
6. L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectorypooled deep-convolutional descriptors. In Proceedings of the IEEE
Conference on Computer Vision and Pattern Recognition (CVPR),
pages 4305–4314, 2015.
7. BishoySefen, Sebastian Baumbach et. al. / Human Activity
Recognition Using Sensor Data of Smart phones and Smart watches/
ICAART 2016[1]
8. un X, Chen C, Manjunath BS , ―Probabilistic motion parameter
models for human activity recognition,‖ In: Proceedings of 16th
international conference on pattern recognition, pp 443–450

9. Kwon W, Lee TW, ― Phoneme recognition using ICA-based feature
extraction and transformation,‖ Signal Process 84(6):1005– 1019, 2004
10. Lee SI, Batzoglou S, ―Application of independent compo-nent
analysis to microarrays,‖ Genome Biol 4(11):R76.1–21, 2003