Applied Data Science Project

An Open Dataset for Human Activity Analysis using Smart Devices

Adi Cohen, Amit Eliav , Rina Veler and Daniel Levi

Dr. Omri Alush and Ran Eisenberg

https://github.com/amiteliav/Python-Project-Human-Activity-Analysis-from-smart-devices

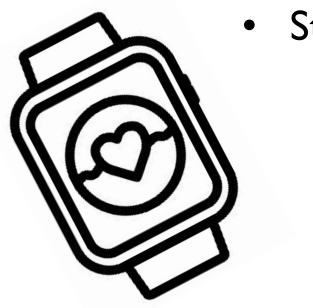
10.9.2023

Introduction

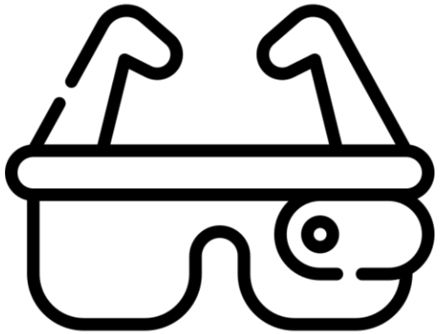
- **Objective:** Classify User Activities Based on Smart Device Data
- **Data Sources:** Information from three smart devices:
  1) Smart Glasses
  2) Smartwatch
  3) Smartphone
- **Labeling Data:** CSV file with activity labels for time frames
- **Dataset:** https://www.kaggle.com/datasets/sasanj/human-activity-smart-devices?resource=download&select=smartphone.csv
- **Dataset Details:**
  1) Data Collection Span: 15 consecutive days.
  2) Varied time intervals across devices (morning to evening).

Measurements

- **Smart-watch**:
  - Accelerometer
  - Battery
  - Gravity
  - Gyroscope
  - Heart rate
  - Linear acceleration
  - Orientation
  - Pressure
  - Rotation vector
  - Step counter
  - Step detector



- **Smart-glasses**:
  - Acceleration in the X, Y, and Z axes
  - Gyroscope in the X, Y, and Z axes
  - EOG Left, Right, Horizontal, Vertical (tracking the eye movement)



- **Smartphone**
  - Accelerometer
  - Activity
  - Audio
  - Battery
  - Bluetooth
  - Gravity
  - Gyroscope
  - Light
  - Linear acceleration
  - Magnetometer
  - Orientation
  - Pressure
  - Proximity
  - Rotation vector
  - Step counter
  - Step detector
  - Wi-Fi

Data Challenged and Issues

- **Feature Integration:** all features consolidated in a single column.
- **Inconsistent Sampling Times:** variability in sampling rates, some features are event-based.
- **Lack of Uniformity:** absence of consistent sampling times across features.
- **Separate Labeling File:** data tags stored in separate file.
- **Incomplete Labeling:** some time frames lack activity labels.
- **Labeling Discrepancies:** different labels for overlapping time frames.
- **Unclear Feature Meanings:** lack of definitions for individual characteristics.
- **Limited Data Volume:** insufficient quantity of data available

Workflow Stages

1. **Label-Data Synchronization:** organizing and aligning labels with the data table.
2. **Organizing and Cleaning the Data**
3. **Visual Data Exploration**
4. **Dimensionality Reduction:** utilizing dimensionality reduction techniques to assess data separation.
5. **Model Exploration:** testing various models, selecting the optimal model, and evaluating performance.
6. **Feature Significance Analysis:** investigating the importance of individual features.

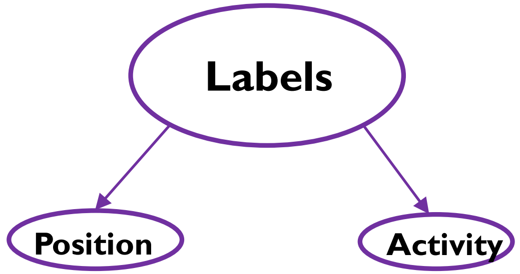
The procedure was conducted individually for each device (smartphone and smartwatch) and subsequently for the combination of both devices.

Addressing Label Challenges

- The labeling of the activities is in a separate table that is divided into time segments. In each section it is indicated what activities were done as follows:

index	activity_type	duration	from	to
0	Video games	01:43	2017-07-01 21:34	2017-07-01 23:17
1	In computer	00:03	2017-07-01 21:29	2017-07-01 21:32
2	At home	13:35	2017-07-01 21:13	2017-07-02 10:49
3	In computer	00:05	2017-07-01 21:08	2017-07-01 21:13
4	Eat	00:18	2017-07-01 20:49	2017-07-01 21:07
...	...	...	...	...
19	Train	00:17	2017-07-10 12:47	2017-07-10 13:04
20	Walk	00:02	2017-07-10 12:36	2017-07-10 12:38
21	In bus	00:08	2017-07-10 12:28	2017-07-10 12:36
0	Movie	01:39	2017-07-11 21:36	2017-07-11 23:16

Overlap in time →



Missing labels for a period → Remove data rows without labels

Addressing Label Challenges

- **Labels Overlap:**
  We saw that there is a lot label overlap. For instance, the "at home" label consistently intersects with common activities such as cooking, watching TV, eating, sleeping, and more. To address this, labels were categorized into two distinct groups: **activity** and **position**. And then we have two classification problems. In cases of remaining overlap after categorization (e.g., "Watching TV" and "Eat"), the label with the longer duration was selected.

```
# map from activity to type
dic = {'At home': 'Position','In bus': 'Position','In vehicle': 'Position','Running': 'Activity','Train': 'Position'
       ,'Walk':'Activity','Work': 'Position','Cooking':'Activity','Eat':'Activity','Football':'Activity','In computer':'Activity'
       ,'Meeting':'Activity','Movie':'Activity','Pause':'Activity','Picnic ':'Activity','Sleep':'Activity','Video games':'Activity'
       ,'Watching TV':'Activity','Shop':'Position'}
report_edit['field'] = report_edit['activity'].apply(lambda x: dic.get(x))
```

[11]

```
report_edit.head()
```

[12]

	activity	duration	from	to	field
129	Eat	00:20	2017-06-29 09:57	2017-06-29 10:17	Activity
128	In vehicle	00:21	2017-06-29 10:46	2017-06-29 11:07	Position
127	Walk	01:52	2017-06-29 11:09	2017-06-29 13:01	Activity
126	Shop	00:28	2017-06-29 12:12	2017-06-29 12:40	Position
123	In bus	00:22	2017-06-29 13:02	2017-06-29 13:24	Position

→ Rarely encounter temporal overlap following division, eliminating any remaining labels.

Addressing Label Challenges

- **Merging synonymous labels:**
  Merge some labels that representing the same activity as shopping and shop.
- **Un-common labels:**
  Delete labels that appeared only once and for brief durations e.g., “On the stop”, “Took off glasses”.

```
report_edit.groupby('activity').size()
```

[9]

activity	
At home	11
Cooking	4
Eat	18
Football	1
In bus	19
In computer	18
In vehicle	8
Meeting	5
Movie	3
On bus stop	1
Pause	4
Phone was out of the pocket (forgot)	1
Picnic	1
Running	2
Shop	3
Shopping& wearing	1
Sleep	1
Took off glasses	1
Train	9
Video games	6
Walk	29
Walking&party	2
Watching TV	1
Work	2

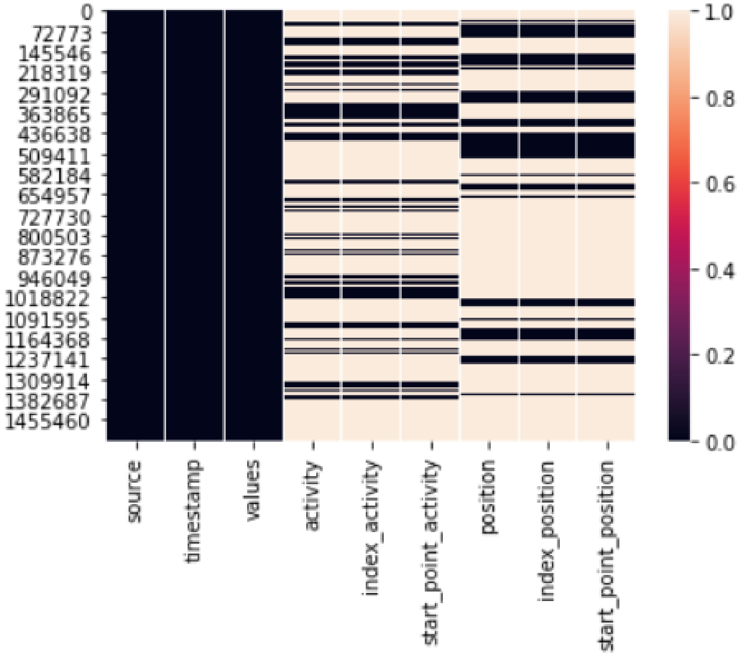
dtype: int64

shop

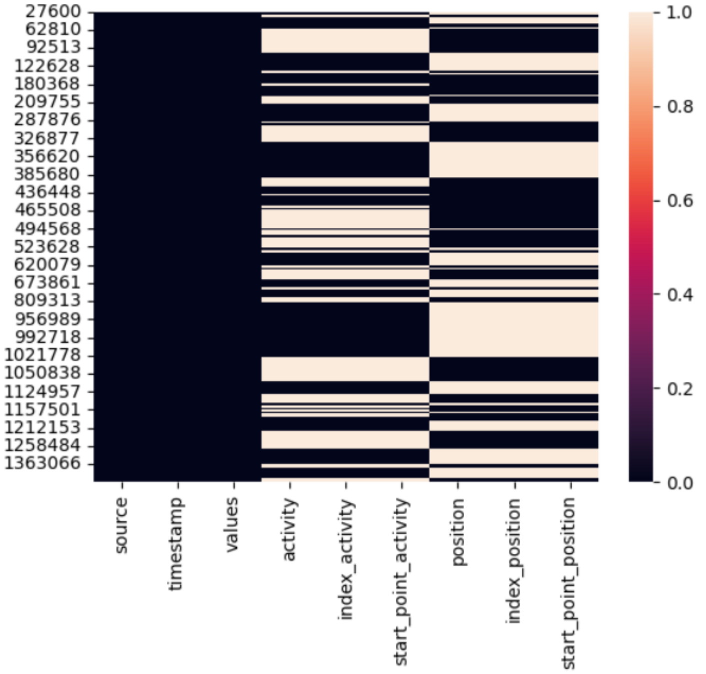
walk

Addressing Label Challenges

- **Synchronization:**
  Labels and device data were initially stored in separate tables. To assign labels to each data row, we determined its corresponding duration.
  Rows lacking labels were subsequently removed from the dataset.



Remove unlabeled data rows



Example from smartphone data

```
Num of activity before dropping unlabels rows: 1528218
Num of activity after dropping unlabels rows : 755553
```

Organize Smart Devices Table

- The feature data is presented in rows, and we aim to arrange it into a table format.

	source	timestamp	values	activity
0	pressure	2017-06-29 09:57:00.070	['963.3365']	Eat
1	step_detector	2017-06-29 09:57:00.280	['1.0']	Eat
2	audio	2017-06-29 09:57:00.448	['8513.0', '7546.236', '32767.0', '45.0']	Eat
3	step_detector	2017-06-29 09:57:01.000	['1.0']	Eat
4	gravity	2017-06-29 09:57:01.441	['2.3287392', '-8.675122', '-3.9356887']	Eat
5	audio	2017-06-29 09:57:01.447	['8273.0', '754	
6	step_detector	2017-06-29 09:57:01.521		
7	linear_acceleration	2017-06-29 09:57:01.600	['3.6848893',	
8	rotationVector	2017-06-29 09:57:01.607	['-0.6266626', '0.5223	
9	step_detector	2017-06-29 09:57:02.275		

index	timestamp	pressure	ndex_activity	activity	step_detector	audio_mic_1	audio_mic_2	audio_mic_3	audio_mic_4	...	gyroscope_Z	magn
1	2017-06-29 09:57:30	963.210350	0.0	Eat	52.0	11927.200000	7563.650340	32767.0	45.0	...	0.018634	
2	2017-06-29 09:58:00	963.377710	0.0	Eat	28.0	9694.366667	7582.004297	32767.0	45.0	...	-0.001961	
3	2017-06-29 09:58:30	963.352867	0.0	Eat	25.0	7590.000000	7579.942257	32767.0	45.0	...	-0.176842	
4	2017-06-29 09:59:00	963.356162	0.0	Eat	52.0	11810.733333	7589.977663	32767.0	45.0	...	-0.343290	
5	2017-06-29 09:59:30	963.335040	0.0	Eat	20.0	7881.866667	7600.249840	32767.0	45.0	...	-0.174356	
6	2017-06-29 10:00:00	963.346502	0.0	Eat	45.0	12377.733333	7606.813603	32767.0	45.0	...	-0.788126	
7	2017-06-29 10:00:30	963.392575	0.0	Eat	51.0	10225.166667	7622.846237	32767.0	45.0	...	0.312913	
8	2017-06-29 10:01:00	963.333287	0.0	Eat	46.0	9992.400000	7629.055077	32767.0	45.0	...	0.262846	
9	2017-06-29 10:01:30	963.332382	0.0	Eat	4.0	6361.866667	7633.940360	32767.0	45.0	...	0.008958	

Organize Smart Devices Table

- To convert the feature data from rows to a structured table format, we followed these steps:
  1) We isolated specific data sources from the initial table.
  2) The table was organized chronologically based on time.
  3) For each data source we checked what its value represents and treated accordingly (examples in the following slides).
  4) Rows were resampled at a uniform rate of 30 seconds.
  5) Rows lacking labels were excluded from the dataset.

- Finally, we merged the processed tables into a unified dataset by the columns: timestamp, activity, and index_activity.

Organize Smart Devices Table

- For sources like orientation, accelerometer, gyroscope, and magnetometer, where each measurement includes three values corresponding to different axes, we split these values into separate columns for each axis

	source	timestamp	values	index_activity	activity
12	orientation	2017-06-29 09:57:03.249	[['85.41814085450436', '81.67451417042038', ...	0.0	Eat
28	orientation	2017-06-29 09:57:08.255	[['80.29948097728092', '81.19698832478811', ...	0.0	Eat
47	orientation	2017-06-29 09:57:13.258	[['-106.77375182292474', '-12.406880679203606...	0.0	Eat
70	orientation	2017-06-29 09:57:18.270	[['-90.11583568261241', '22.676325374638004',...	0.0	Eat
95	orientation	2017-06-29 09:57:23.297	[['140.52100617228405', '54.62359435881448', ...	0.0	Eat
121	orientation	2017-06-29 09:57:28.315	[['71.11310877004554', '68.27504905668935', ...	0.0	Eat
146	orientation	2017-06-29 09:57:33.335	[['-88.805887362074', '33.346895980969855', ...	0.0	Eat
170	orientation	2017-06-29 09:57:38.341	[['157.72113004966866', '73.05821687207094', ...	0.0	Eat
195	orientation	2017-06-29 09:57:43.343	[['32.711087331511706', '59.89132652976804', ...	0.0	Eat
220	orientation	2017-06-29 09:57:48.349	[['130.14925463459298', '67.79566539960284', ...	0.0	Eat

	orientation_X	orientation_Y	orientation_Z	index_activity	activity
	30.077025	49.339932	31.488352	0.0	Eat
	22.952714	60.453024	-5.277635	0.0	Eat
	43.617109	81.183346	8.174922	0.0	Eat
	4.632983	75.299680	48.593347	0.0	Eat
	-32.125872	65.046507	-46.384025	0.0	Eat
2017-06-29 09:59:30	135.779456	77.457162	-25.646591	0.0	Eat
2017-06-29 10:00:00	56.245559	62.390736	-29.571418	0.0	Eat
2017-06-29 10:00:30	-48.483381	76.154782	84.164729	0.0	Eat
2017-06-29 10:01:00	7.742334	67.175833	18.564167	0.0	Eat
2017-06-29 10:01:30	1.444898	82.312208	29.383669	0.0	Eat

orientation we split to x y z axis.

Organize Smart Devices table

- For sources like **gravity and linear acceleration**, which provided three values across different axes, we generated a new feature by calculating the square root of the sum of the squares of these values ($\sqrt{x^2 + y^2 + z^2}$, (following [1])

	source	timestamp	values	activity	index_activity	sta
4	gravity	2017-06-29 09:57:01.441	[['2.3287392', '-8.675122', '-3.9356887']]	Eat	0.0	
23	gravity	2017-06-29 09:57:06.464	[['0.28148645', '-9.659683', '1.6678368']]	Eat	0.0	
40	gravity	2017-06-29 09:57:11.492	[['3.1542559', '-8.940354', '-2.5082197']]	Eat	0.0	
62	gravity	2017-06-29 09:57:16.512	[['2.2692857', '-9.402923', '-1.6142402']]	Eat	0.0	
87	gravity	2017-06-29 09:57:21.539	[['3.140906', '-8.014466', '-4.6982408']]	Eat	0.0	
112	gravity	2017-06-29 09:57:26.559	[['2.1893961', '-9.27696', '-2.3054175']]	Eat	0.0	
137	gravity	2017-06-29 09:57:31.586	[['-0.8873541', '-9.568674', '1.9553765']]	Eat	0.0	
162	gravity	2017-06-29 09:57:36.613	[['1.5669698', '-8.502972', '-4.62758']]	Eat	0.0	
188	gravity	2017-06-29 09:57:41.632	[['-0.65013564', '-9.700961', '-1.2802526']]	Eat	0.0	
213	gravity	2017-06-29 09:57:46.660	[['0.9532338', '-9.525594', '-2.1271582']]	Eat	0.0	

$$Gravity = \sqrt{G_x^2 + G_y^2 + G_z^2}$$

	index_activity	activity	gravity
timestamp			
2017-06-29 09:57:00	0.0	Eat	9.531679
2017-06-29 09:57:30	0.0	Eat	9.428560
2017-06-29 09:58:00	0.0	Eat	9.495720
2017-06-29 09:58:30	0.0	Eat	9.548030
2017-06-29 09:59:00	0.0	Eat	9.502039
2017-06-29 09:59:30	0.0	Eat	9.609119
2017-06-29 10:00:00	0.0	Eat	9.368099
2017-06-29 10:00:30	0.0	Eat	9.477471
2017-06-29 10:01:00	0.0	Eat	9.688910
2017-06-29 10:01:30	0.0	Eat	9.775084

[1] Faye S, Bronzi W, Tahirou I, Engel T. Characterizing user mobility using mobile sensing systems. *International Journal of Distributed Sensor Networks*. 2017;13(8). doi:10.1177/1550147717726310

Organize Smart Devices table

- For the audio source, the 'value' column contained four numbers, each representing a different microphone. We transformed each microphone's data into separate columns

	source	timestamp	values	activity	index_activity
2	audio	2017-06-29 09:57:00.448	[['8513.0', '7546.236', '32767.0', '45.0']]	Eat	0.0
5	audio	2017-06-29 09:57:01.447	[['8273.0', '7546.3276', '32767.0', '45.0']]	Eat	0.0
10	audio	2017-06-29 09:57:02.448	[['8690.0', '7546.472', '32767.0', '45.0']]	Eat	
13	audio	2017-06-29 09:57:03.449	[['13602.0', '7547.238', '32767.0', '45.0']]	Eat	
18	audio	2017-06-29 09:57:04.449	[['25580.0', '7549.518', '32767.0', '45.0']]	Eat	
21	audio	2017-06-29 09:57:05.446	[['7398.0', '7549.4985', '32767.0', '45.0']]	Eat	

timestamp	audio_mic_1	audio_mic_2	audio_mic_3	audio_mic_4	index_activity	activity
2017-06-29 09:57:00	10406.766667	7549.982757	32767.0	45.0	0.0	Eat
2017-06-29 09:57:30	11927.200000	7563.650340	32767.0	45.0	0.0	Eat
2017-06-29 09:58:00	9694.366667	7582.004297	32767.0	45.0	0.0	Eat
2017-06-29 09:58:30	7590.000000	7579.942257	32767.0	45.0	0.0	Eat
2017-06-29 09:59:00	11810.733333	7589.977663	32767.0	45.0	0.0	Eat
2017-06-29 09:59:30	7881.866667	7600.249840	32767.0	45.0	0.0	Eat
2017-06-29 10:00:00	12377.733333	7606.813603	32767.0	45.0	0.0	Eat

We split the audio data for separate microphones.

Organize Smart Devices table

- For sources representing numerical data, such as pressure, step detector, step counter, light, and battery, we converted the 'value' from a string to a float.
- Measurements with unclear information or insufficient row count, such as rotation_vector, activity, and proximity, were excluded.
- For Wi-Fi and Bluetooth sources, we transformed the data to represent the number of devices, as specific device identities were not relevant to our analysis.

wifi	2017-07-12 18:20:11.892	[5500, 7207, 3564, 3544]	Meeting	141.0

4

Organize Smart Devices table

- **Sampling Rate:**
  Not all measurements were uniformly sampled , some were event-based. To ensure consistent sampling, we employed the 'resample' function with $T_s = 30_{sec}$, which corresponds to the lowest sample rate among the measurements. This resulted in uniform samples at T_s intervals.
  For each measurement, we selected an appropriate aggregation function, such as mean(), sum(), mode() or len(). For instance, we summed over steps.

	step_counter	index_activity	activity
timestamp			
2017-06-29 09:57:00	25	0.0	Eat
2017-06-29 09:57:30	45	0.0	Eat
2017-06-29 09:58:00	32	→ sum over steps	t
2017-06-29 09:58:30	8	0.0	Eat
2017-06-29 09:59:00	45	0.0	Eat
...	...	...	...
2017-07-12 18:01:30	42	141.0	Meeting
2017-07-12 18:02:30	24	141.0	Meeting
2017-07-12 18:03:00	16	141.0	Meeting
2017-07-12 18:04:30	→ Uniform T_s	141.0	Meeting
2017-07-12 18:05:00	25	141.0	Meeting

Handling Missing Data

- We initially checked the extent of missing data for each characteristic. Those with over 80 percent NaN values were excluded from further analysis ('light' and 'step_counter’).

```
step_detector percentage of nan: 0.6693894220951603
step_counter percentage of nan: 0.8260159281192567
light percentage of nan: 0.8572595466612212
battery percentage of nan: 0.7081886869511946
bluetooth percentage of nan: 0.5458443945272616
```

- **Step_detecor Data:**
We fill the NaN’s with zero, given the absence of step detection during those periods.

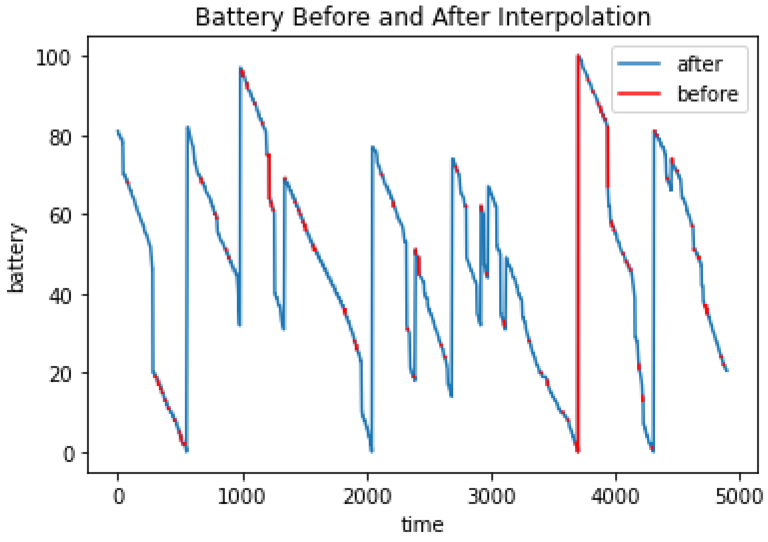
	step_detector	step_counter
0	39.0	25.0
1	52.0	45.0
2	28.0	32.0
3	25.0	8.0
4	52.0	45.0
...	...	...
4892	0.0	0.0
4893	0.0	0.0
4894	0.0	0.0
4895	0.0	0.0
4896	0.0	0.0

Handling Missing Data

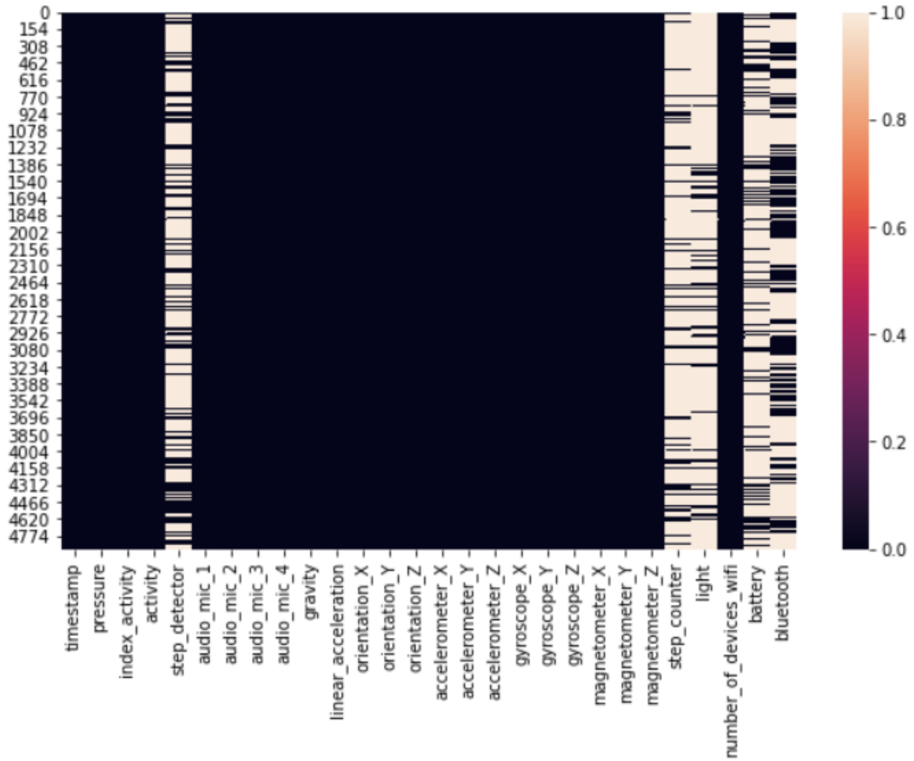
- **Bluetooth Data:**
  NaN values were addressed by evaluating the time difference between the NaN row and the previous row. If the time difference was less than a minute, we filled the NaN with the value from the previous row. For larger time intervals, we filled the NaN's with the average Bluetooth value across the dataset. The time window approach ensured accuracy in cases where Bluetooth stability indicated minimal movement to detect additional devices.

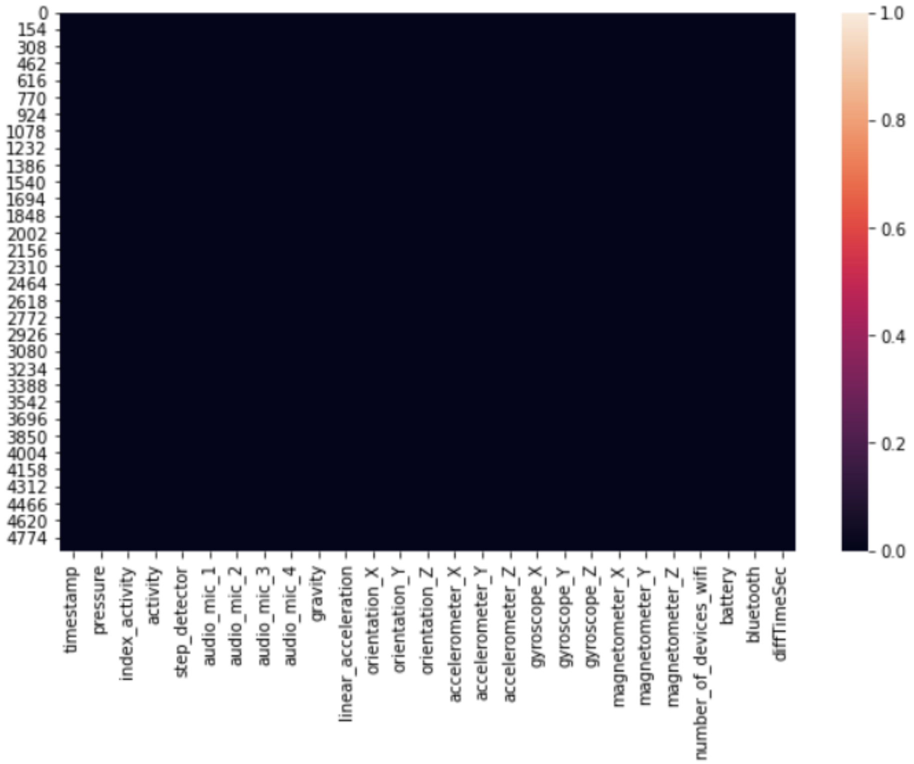
- **Battery Data:**
  NaN's in battery and light data were resolved using interpolation techniques.

- Any remaining rare NaN values were removed from the dataset.

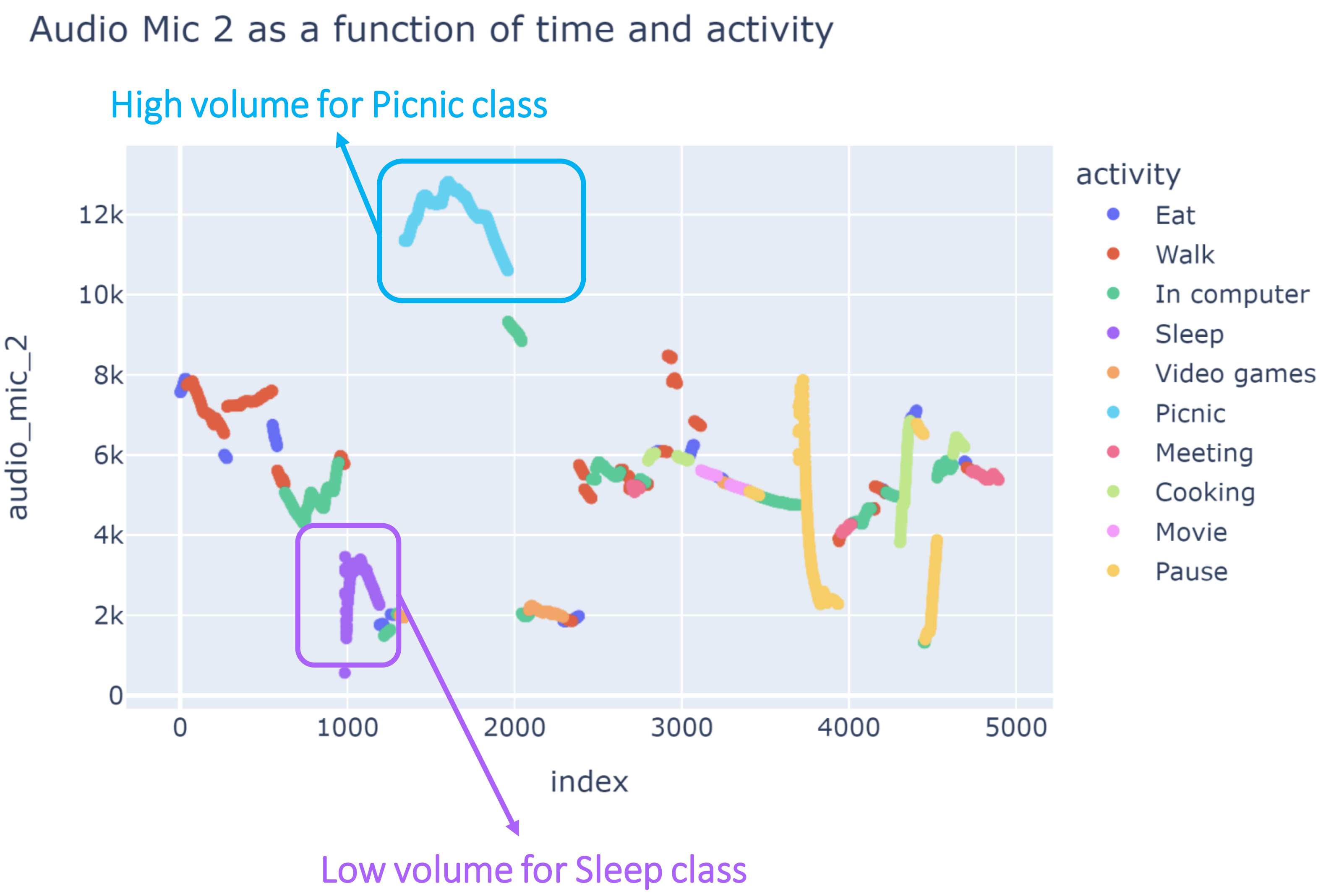


Handling Missing Data

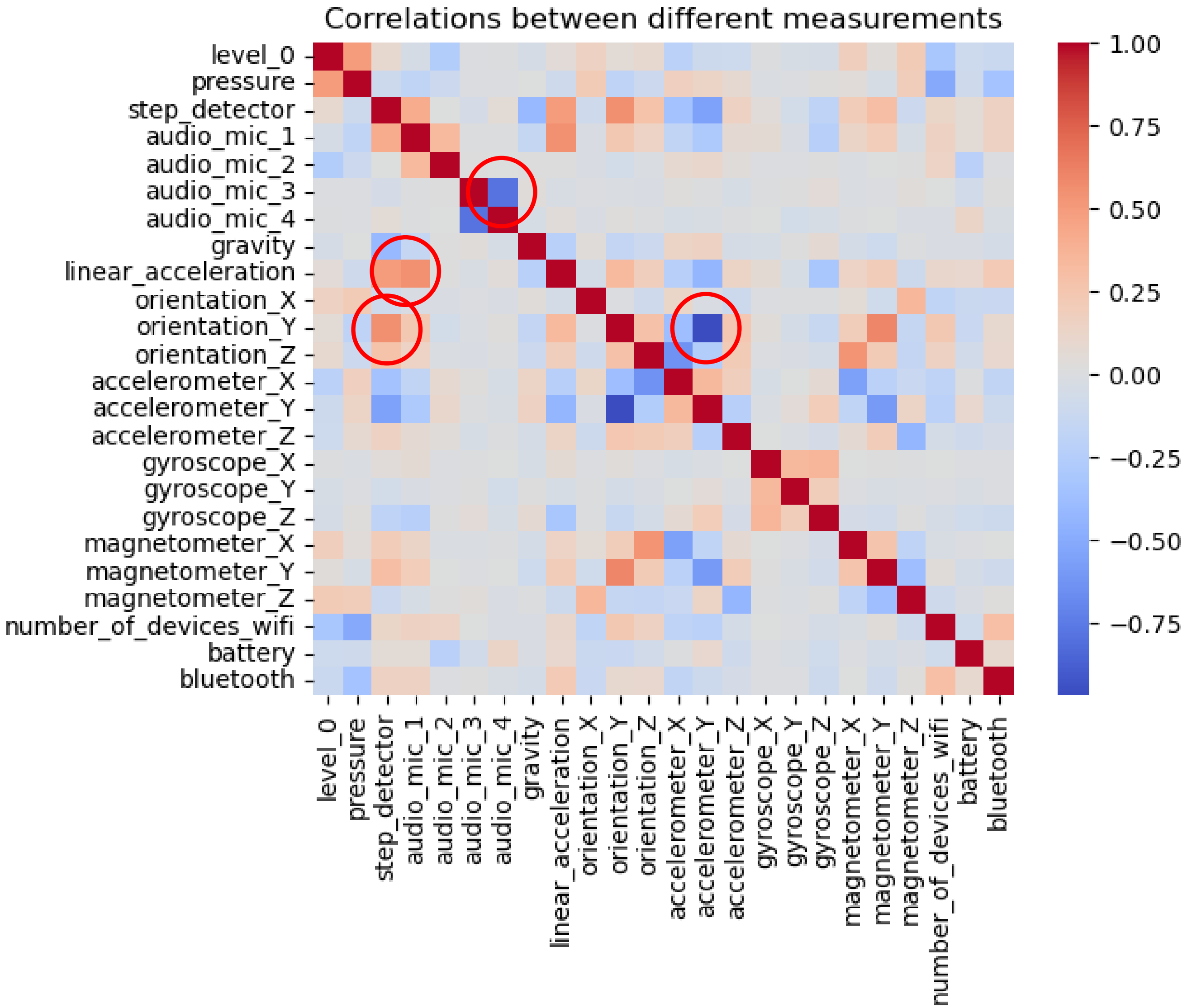




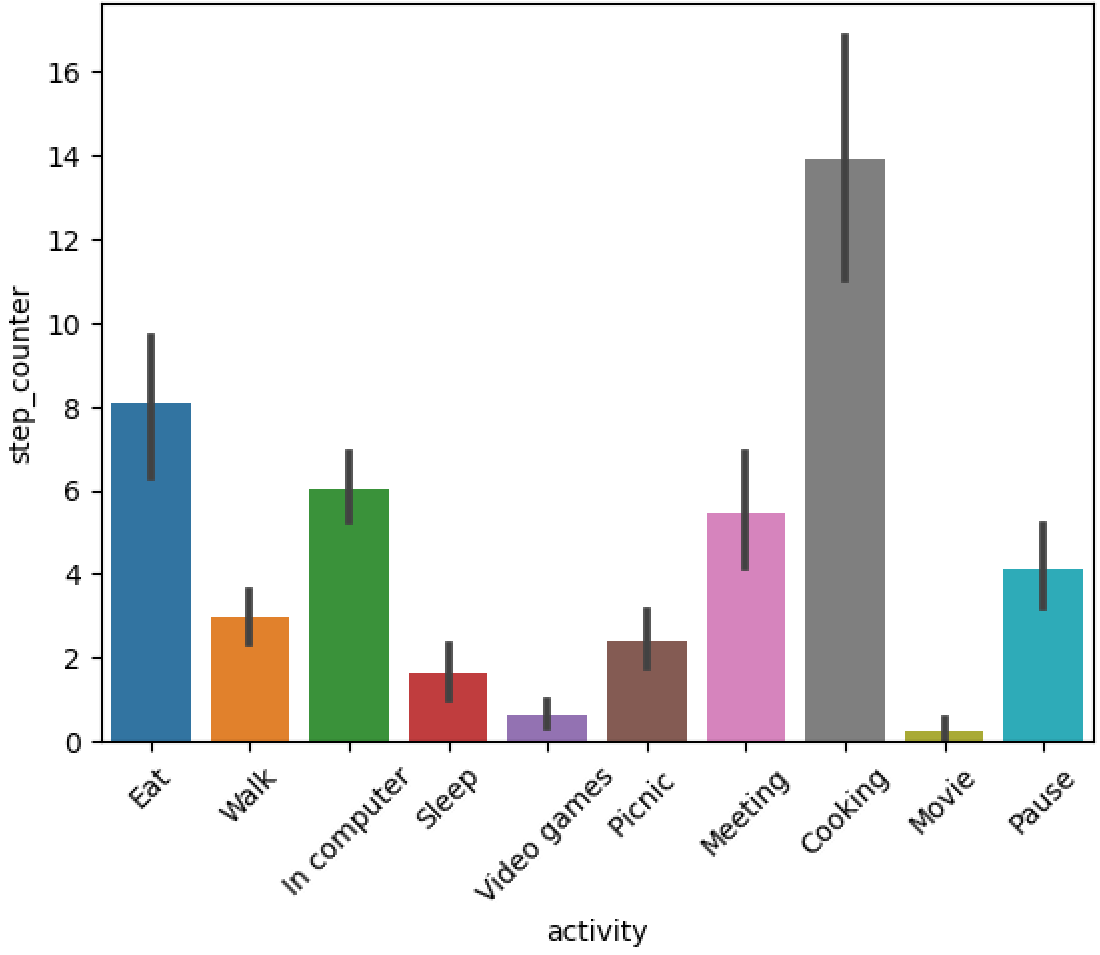
Visualization



Visualization – Correlation Heatmap



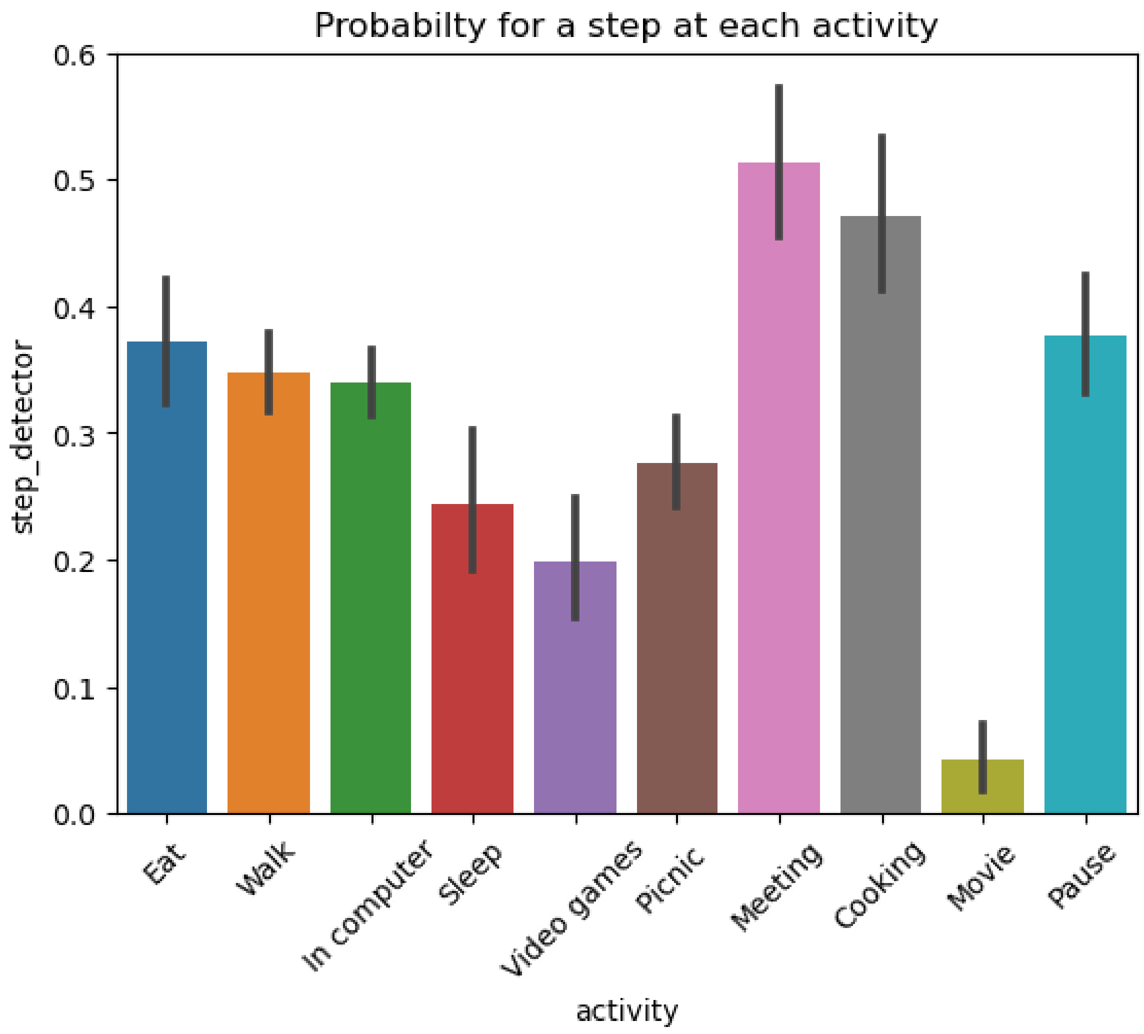
Visualization



Mean value of step detector for each activity.

‘Walk’ activity surprised us!

Visualization

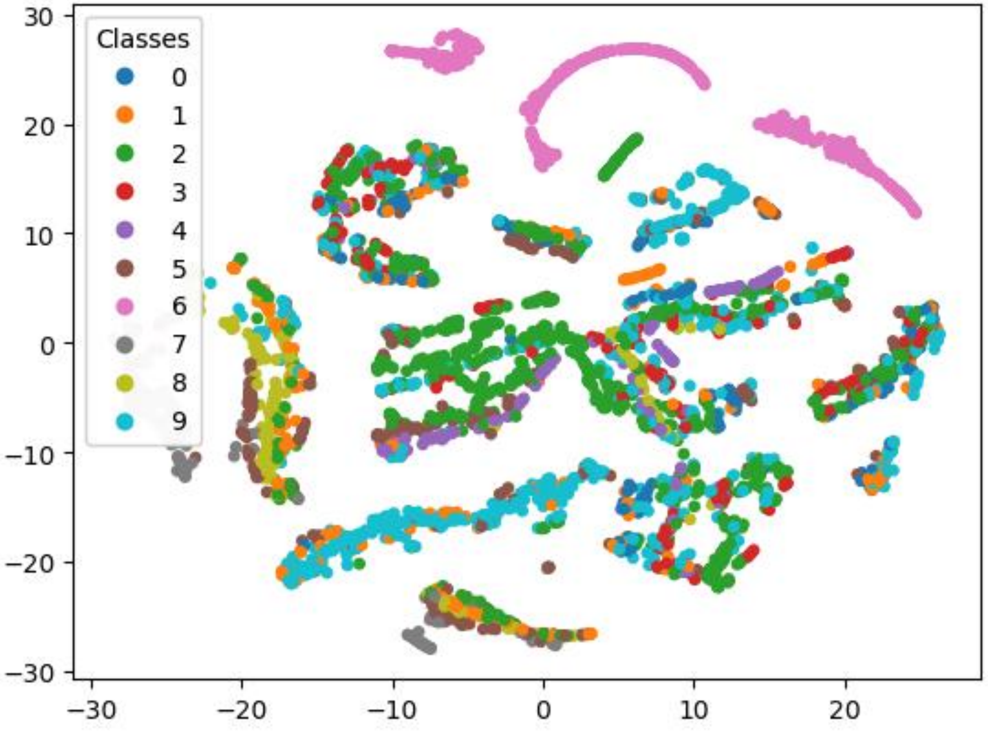


‘Walk’ activity surprised us!

It is not expected that the step detector probability is less than 50% in the activity 'walk', and beyond 50% in 'meeting'.

Unsupervised Methods - TSNE, K-means & UMAP

TSNE



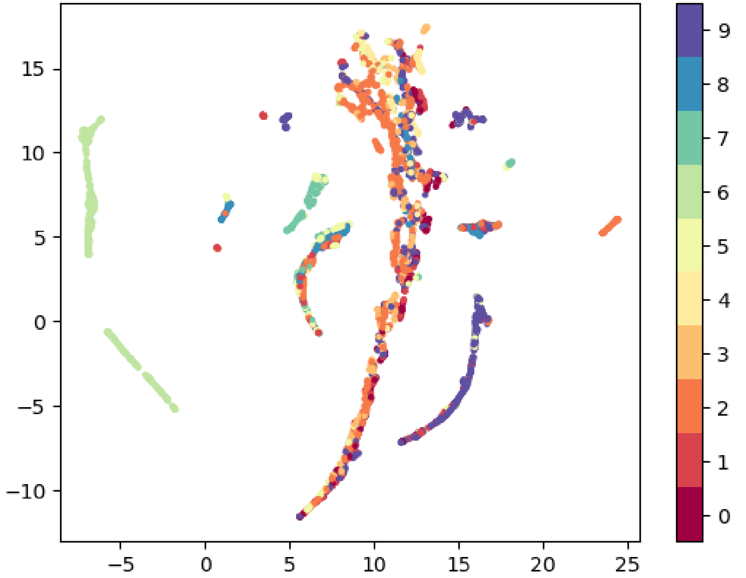
{'Cooking': 0, 'Eat': 1, 'In computer': 2, 'Meeting': 3, 'Movie': 4, 'Pause': 5, **'Picnic ': 6**, 'Sleep': 7, 'Video games': 8, 'Walk': 9}

K-means

1) Accuracy = 45.5%
2) normalized mutual information score: 0.333
3) homogeneity score: 0.305

* classification results obtained after running a code to find best match between clustering and ground-truth labels

UMAP



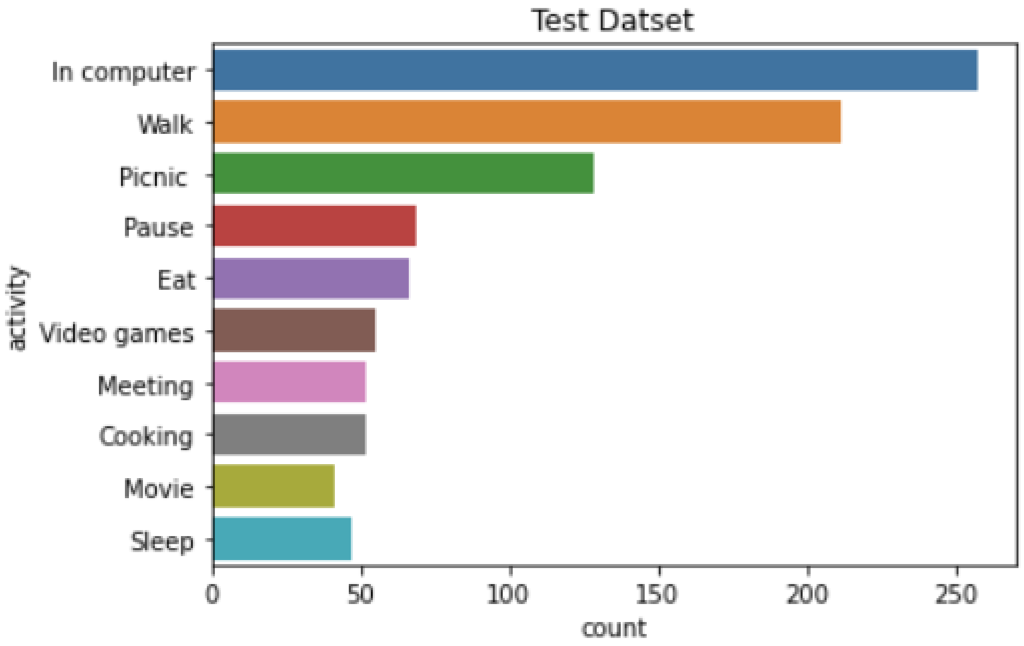
Basic unsupervised methods did not result in good enough data clustering

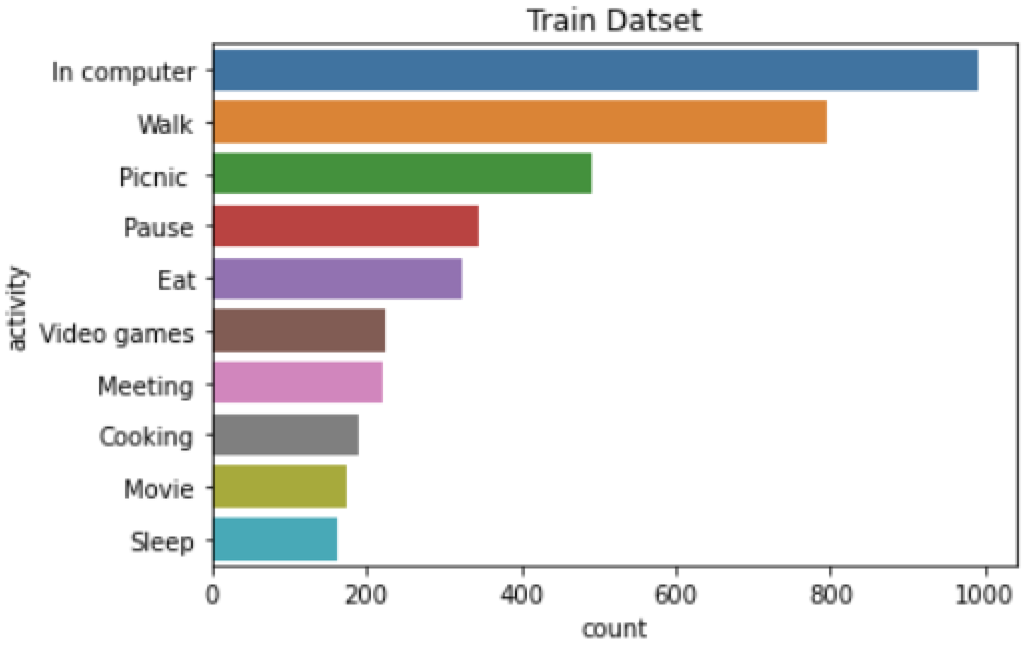
Model Evaluation Process

- Models Under Consideration:
  1) Logistic Regression
  2) Support Vector Machine (SVM)
  3) Decision Tree
  4) AdaBoost
  5) Random Forest
  6) k-Nearest Neighbors (KNN)

- To identify the most suitable model, we conducted the following steps:
  1) Divided the dataset into training and testing sets.
  2) Analyzed the distribution of labels across different categories.
  3) Applied standard scaling to the dataset.
  4) Evaluated the performance of various models with diverse parameter configurations, utilizing the grid_search and cross_val_score functions.

Label Distribution





Performance Comparison

```
Mean score: $0.695$ for algorithm: $lr$
Mean score: $0.917$ for algorithm: $svm$
Mean score: $0.254$ for algorithm: $adaboost$
Mean score: $0.487$ for algorithm: $tree$
Mean score: $0.983$ for algorithm: $forest$
Mean score: $0.742$ for algorithm: $knn$
The best algorithm is: $forest$. The accurate on all training data is: $0.9997441509741644$
```

- The model obtained as the most successful is the Random Forest, with the following parameters:

criterion	entropy
n_estimators	10
max_depth	None
class_weight	None

Performance Comparison

- We assessed the test performance of the chosen model:

Confusion Matrix

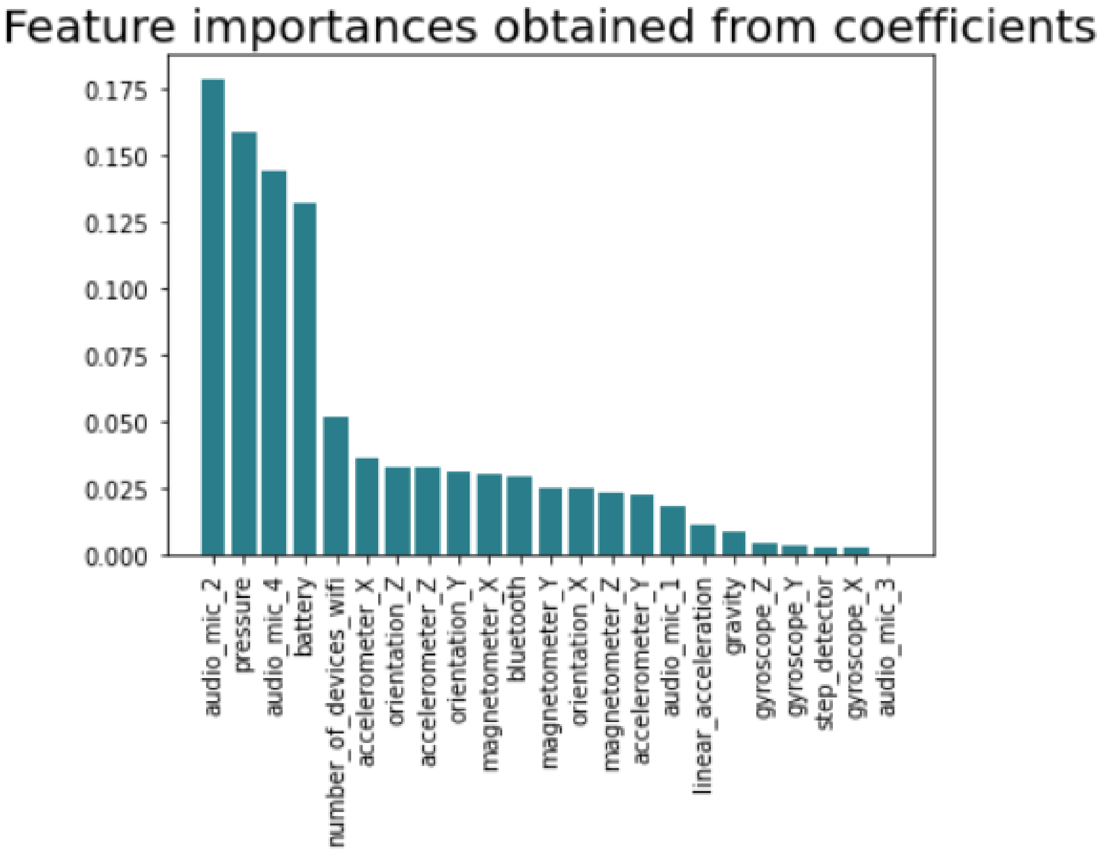
```
[[ 48   3   1   0   0   0   0   0   0   0]
 [  0  66   0   0   0   0   0   0   0   0]
 [  0   0 254   0   0   0   0   0   1   2]
 [  0   0   0  51   0   0   0   0   0   1]
 [  0   0   0   0  41   0   0   0   0   0]
 [  0   0   0   0   0  69   0   0   0   0]
 [  0   0   0   0   0   0 128   0   0   0]
 [  0   0   0   0   0   0   0  47   0   0]
 [  0   1   0   1   0   0   0   0  53   0]
 [  0   0   4   0   0   0   0   0   0 207]]
```

Classification Report

	precision	recall	f1-score	support
Cooking	1.00	0.92	0.96	52
Eat	0.94	1.00	0.97	66
In computer	0.98	0.99	0.98	257
Meeting	0.98	0.98	0.98	52
Movie	1.00	1.00	1.00	41
Pause	1.00	1.00	1.00	69
Picnic	1.00	1.00	1.00	128
Sleep	1.00	1.00	1.00	47
Video games	0.98	0.96	0.97	55
Walk	0.99	0.98	0.98	211
accuracy			0.99	978
macro avg	0.99	0.98	0.99	978
weighted avg	0.99	0.99	0.99	978

Analyzing Feature Importance in the Chosen Model

- As the 'Random Forest' emerged as the best-performing model, we conducted a thorough analysis to identify its key contributing features.



Analyzing Feature Importance in the Chosen Model

- We selected the **<u>top five features</u>** and re-ran the model exclusively with them, the results are shown on the next slide.
- Additionally, we isolated the **<u>least significant five features</u>** and ran the model exclusively with them, the results are shown on the next slide.
- Evidently, the results align closely with our initial assumptions.

Top Five Features	Least Five Features
audio_mic_2	gyroscope_Z
pressure	gyroscope_Y
audio_mic_4	step_detector
battery	gyroscope_X
wifi	audio_mic_3

Analyzing Feature Importance in the Chosen Model

- Model Performance on the **Top Five Features**:

Confusion Matrix

```
[[ 49   2   1   0   0   0   0   0   0   0]
 [  0  64   0   0   0   0   0   0   0   2]
 [  0   0 257   0   0   0   0   0   0   0]
 [  0   0   0  51   0   0   0   0   0   1]
 [  0   0   0   0  41   0   0   0   0   0]
 [  1   0   0   0   0  68   0   0   0   0]
 [  0   0   0   0   0   0 128   0   0   0]
 [  0   0   0   0   0   0   0  47   0   0]
 [  0   0   0   0   0   0   0   0  55   0]
 [  0   0   2   0   0   0   0   0   0 209]]
```

Classification Report

	precision	recall	f1-score	support
Cooking	0.98	0.94	0.96	52
Eat	0.97	0.97	0.97	66
In computer	0.99	1.00	0.99	257
Meeting	1.00	0.98	0.99	52
Movie	1.00	1.00	1.00	41
Pause	1.00	0.99	0.99	69
Picnic	1.00	1.00	1.00	128
Sleep	1.00	1.00	1.00	47
Video games	1.00	1.00	1.00	55
Walk	0.99	0.99	0.99	211
accuracy			0.99	978
macro avg	0.99	0.99	0.99	978
weighted avg	0.99	0.99	0.99	978

Analyzing Feature Importance in the Chosen Model

- Model Performance on the **Least Five Features**:

Confusion Matrix

```
[[  4   3  25   3   1   0   4   3   1   8]
 [  5   6  18   3   6   6   4   2   1  15]
 [ 17  26 103  13   5  12  32   3   9  37]
 [  2   8  17   1   0   4   6   1   1  12]
 [  3   3   9   0   2   5   5   0   3  11]
 [  2   1  32   3   1   6   8   1   2  13]
 [  7   9  48   4   3   7  19   1   5  25]
 [  2   2  16   0   2   2   8   1   1  13]
 [  3   5  16   2   4   4   9   2   1   9]
 [ 11  27  75   7   4  14  18   3   8  44]]
```

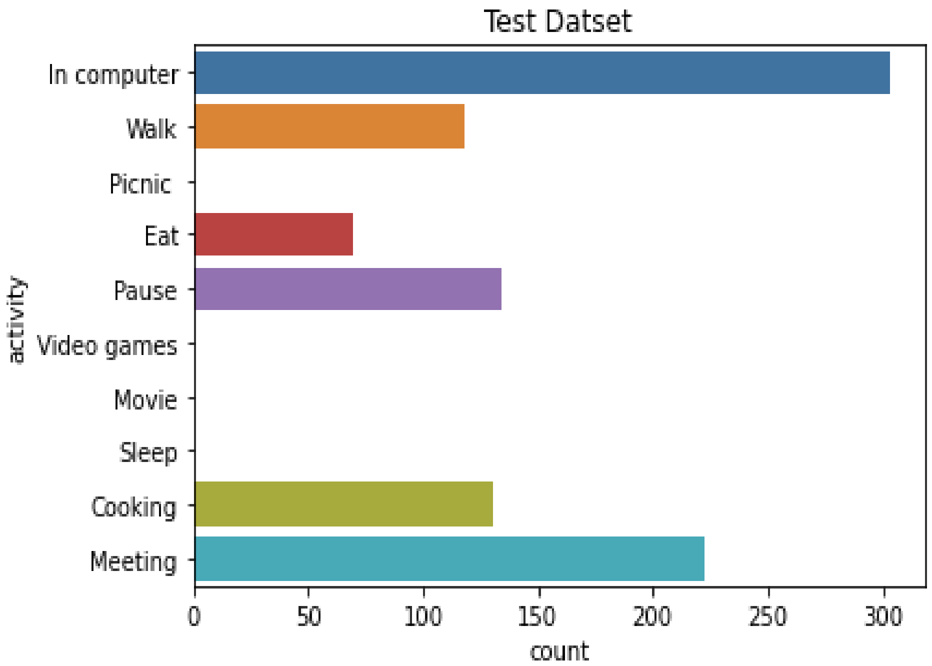
Classification Report

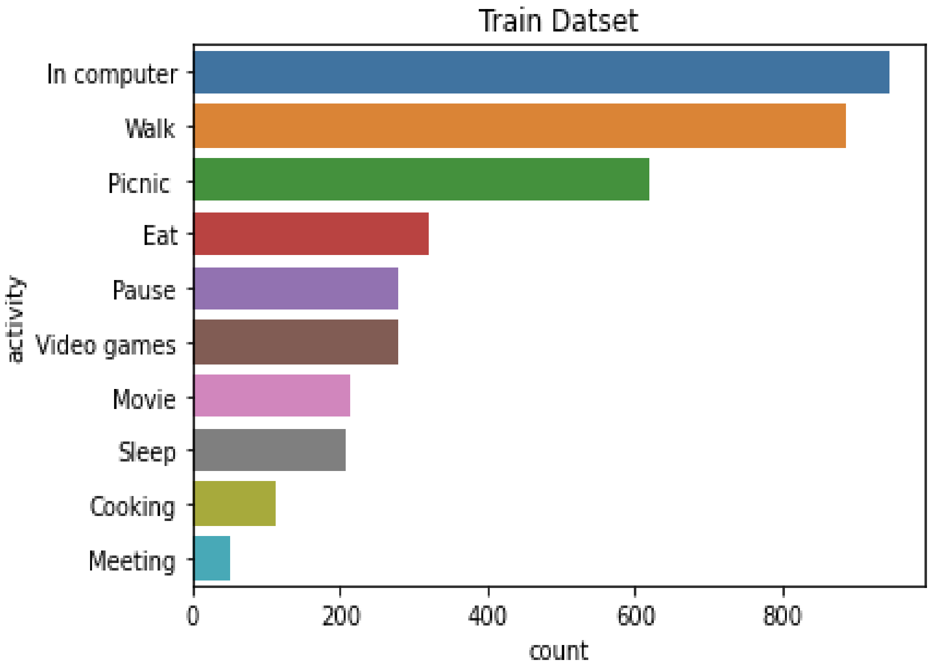
	precision	recall	f1-score	support
Cooking	0.07	0.08	0.07	52
Eat	0.07	0.09	0.08	66
In computer	0.29	0.40	0.33	257
Meeting	0.03	0.02	0.02	52
Movie	0.07	0.05	0.06	41
Pause	0.10	0.09	0.09	69
Picnic	0.17	0.15	0.16	128
Sleep	0.06	0.02	0.03	47
Video games	0.03	0.02	0.02	55
Walk	0.24	0.21	0.22	211
accuracy			0.19	978
macro avg	0.11	0.11	0.11	978
weighted avg	0.17	0.19	0.18	978

Investigating Exceptional Results: Potential Information Leakage

- Given the remarkably strong results, we began to consider the possibility of information leakage.
- During the initial dataset split into training and testing sets, we employed shuffle=True.
- However, because our dataset consists of time series data, mixing all the data together may not be the best approach, as it may lead to data leakage and unrealistic performance estimates.
- As a solution, we tried another approach of time-based splitting.
- In this method, we divide the data into a training set and a test set based on the <u>temporal order of the data</u>. Thus, the training set contains data from earlier time periods, while the test set contains data from later time periods.
- This approach simulates a real scenario where a model is trained on early-stage data and later is used to predict future events.

Label Distribution





Performance Comparison

- We followed the same procedures as previously to determine the most effective model, and once again, our results indicated that the Random Forest model, with the same set of parameters, emerged as the top-performing choice on the training set.
- We assessed the test performance of the chosen model:

Confusion Matrix

1	15	35	0	0	0	0	0	79
0	4	22	0	0	0	0	0	44
0	5	261	0	0	0	0	0	37
8	61	120	0	6	0	0	7	21
0	0	0	0	0	0	0	0	0
0	2	69	0	0	20	1	0	42
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	1	97	0	0	0	0	0	20

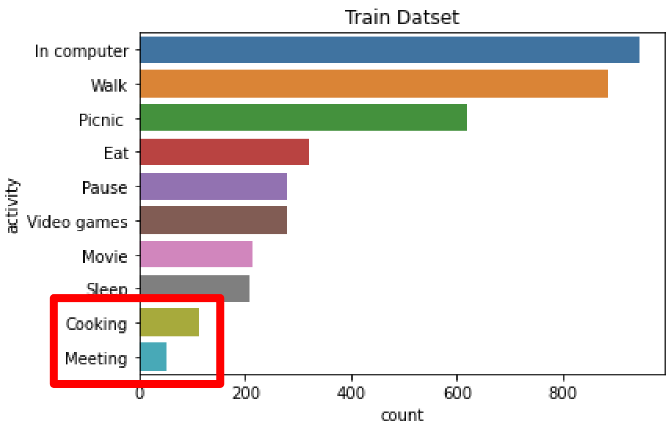
Classification Report

	precision	recall	f1-score	support
Cooking	0.11	0.01	0.01	130
Eat	0.05	0.06	0.05	70
In computer	0.43	0.86	0.58	303
Meeting	0.00	0.00	0.00	223
Movie	0.00	0.00	0.00	0
Pause	1.00	0.15	0.26	134
Sleep	0.00	0.00	0.00	0
Video games	0.00	0.00	0.00	0
Walk	0.08	0.17	0.11	118
accuracy			0.31	978
macro avg	0.19	0.14	0.11	978
weighted avg	0.30	0.31	0.23	978

- There is a significant performance drop when we implemented time-based data slicing. Which could indicate that there was indeed an information leakage.

Analysis of Results

- In our analysis, we believe that the primary factor contributing to the observed low performance is the imbalance in label representation within the test set, where most labels do not comprise a substantial portion of the training data.



	precision	recall	f1-score	support
Cooking	0.07	0.08	0.07	52
Eat	0.07	0.09	0.08	66
In computer	0.29	0.40	0.33	257
Meeting	0.03	0.02	0.02	52
Movie	0.07	0.05	0.06	41
Pause	0.10	0.09	0.09	69
Picnic	0.17	0.15	0.16	128
Sleep	0.06	0.02	0.03	47
Video games	0.03	0.02	0.02	55
Walk	0.24	0.21	0.22	211

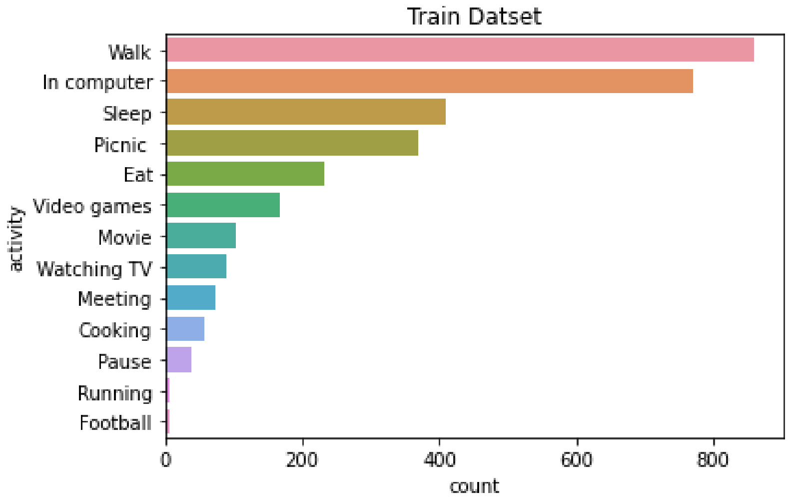
- To substantiate this theory, we conducted a performance assessment by using a test set exclusively composed of labels that constitute a significant proportion of the training dataset.

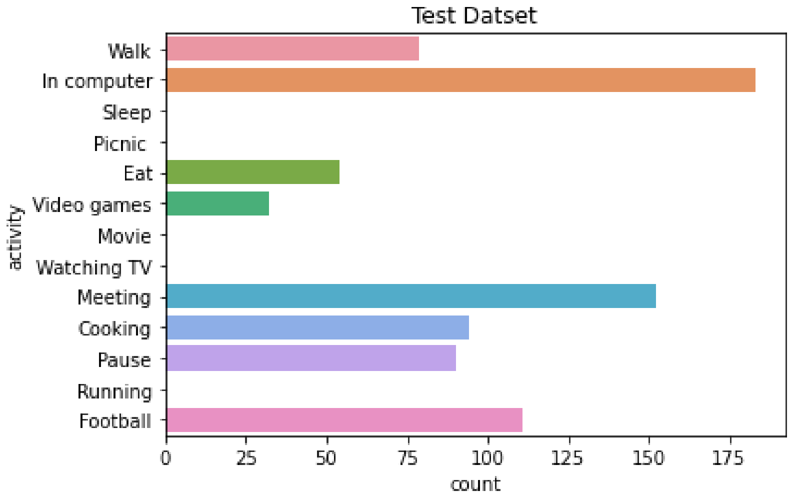
- The outcomes of this evaluation are as follows:

```
[[  0   0   0]
 [  5 261  37]
 [  1  97  20]]
```

	precision	recall	f1-score	support
Eat	0.00	0.00	0.00	0
In computer	0.73	0.86	0.79	303
Walk	0.35	0.17	0.23	118
accuracy			0.67	421
macro avg	0.36	0.34	0.34	421
weighted avg	0.62	0.67	0.63	421

Performance Comparison – SmartWatch





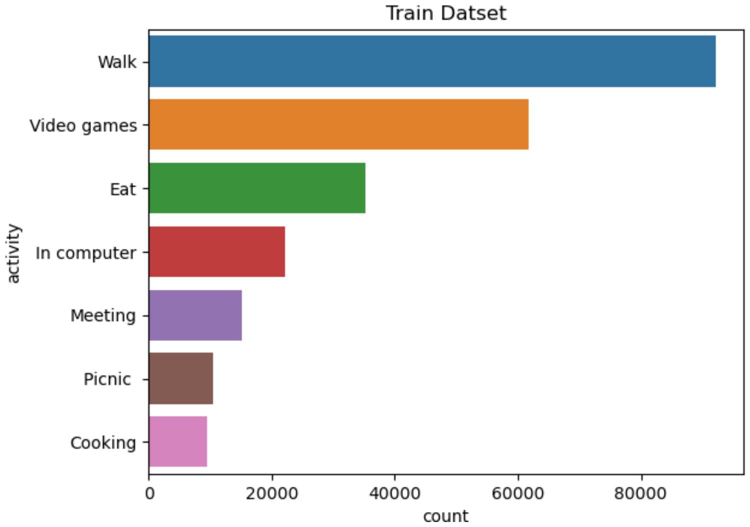
Confusion Matrix

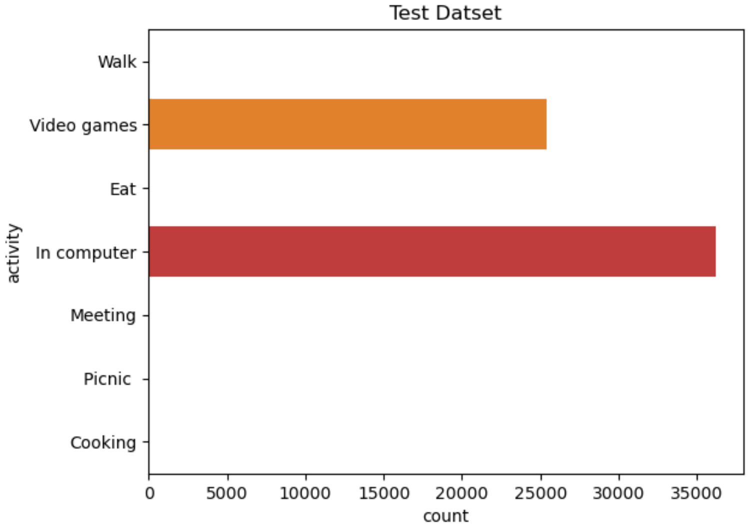
```
[[ 1 19  0 15  0  0  0 14 38  0  7]
 [ 0  1  0  1  0  0  0 14 35  1  2]
 [ 0  4 10 28  0  0  0  1  0  0 68]
 [ 0 23  0 46  0  0  0 28 64  0 22]
 [15 14  0 47  1 25  0 17 18  6  9]
 [ 0  0  0  0  0  0  0  0  0  0  0]
 [ 0 11  0  1  0  0  0  0 76  0  2]
 [ 0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0 16  0  0  0 10  0  1  5]
 [ 0 14  0 15  0  0  0 37  2  0 11]]
```

Classification Report

	precision	recall	f1-score	support
Cooking	0.06	0.01	0.02	94
Eat	0.01	0.02	0.01	54
Football	1.00	0.09	0.17	111
In computer	0.27	0.25	0.26	183
Meeting	1.00	0.01	0.01	152
Movie	0.00	0.00	0.00	0
Pause	0.00	0.00	0.00	90
Picnic	0.00	0.00	0.00	0
Sleep	0.00	0.00	0.00	0
Video games	0.12	0.03	0.05	32
Walk	0.09	0.14	0.11	79
accuracy			0.09	795
macro avg	0.23	0.05	0.06	795
weighted avg	0.42	0.09	0.10	795

Performance Comparison – Smart Glasses





Confusion Matrix

```
[[    0     0     0     0     0     0     0]
 [    0     0     0     0     0     0     0]
 [   25  4534  3553   177   126 15048 12715]
 [    0     0     0     0     0     0     0]
 [    0     0     0     0     0     0     0]
 [  204  1397  1095   124   104 20028  2420]
 [    0     0     0     0     0     0     0]]
```

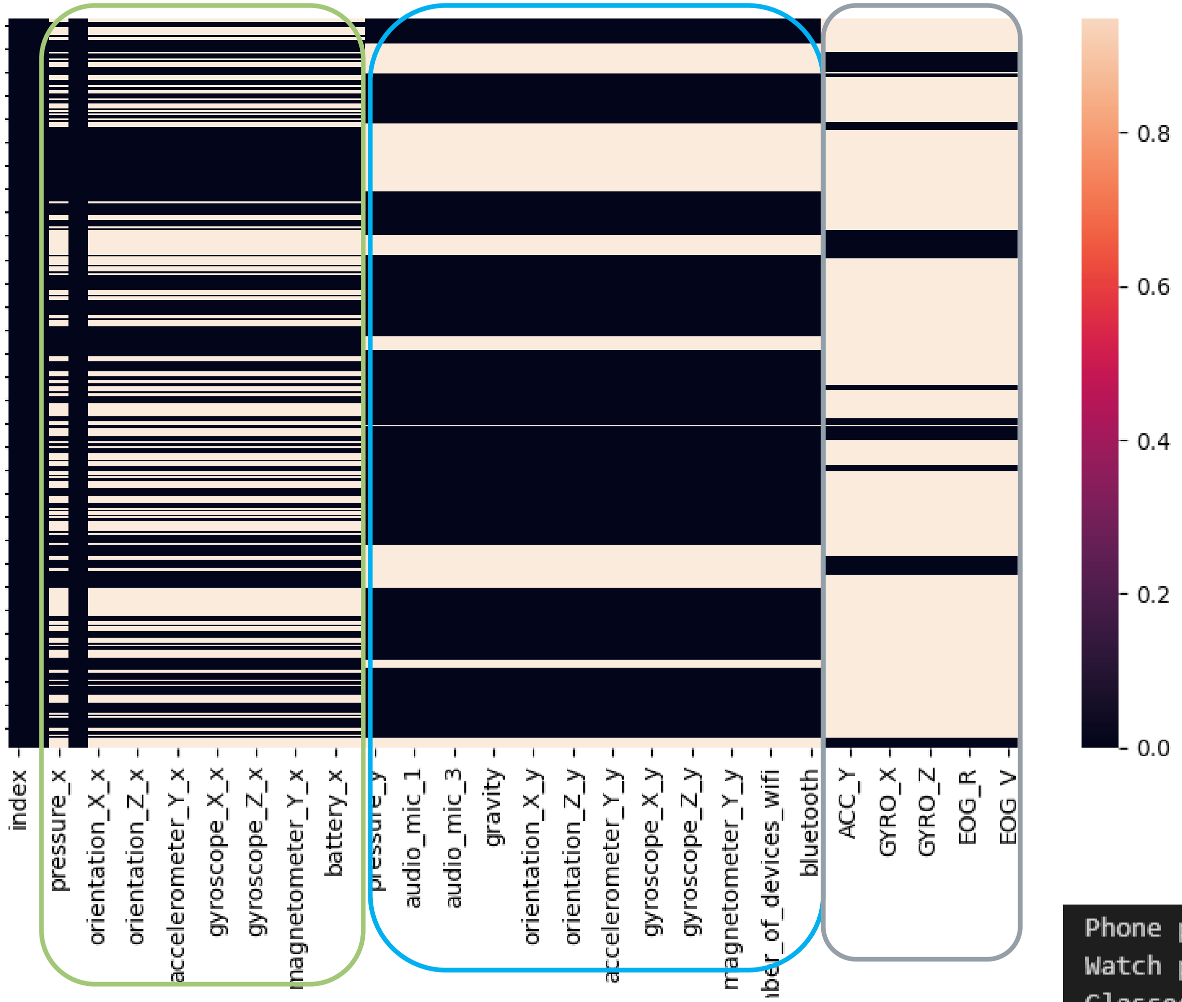
Classification Report

	precision	recall	f1-score	support
Cooking	0.00	0.00	0.00	0
Eat	0.00	0.00	0.00	0
In computer	0.76	0.10	0.17	36178
Meeting	0.00	0.00	0.00	0
Picnic	0.00	0.00	0.00	0
Video games	0.57	0.79	0.66	25372
Walk	0.00	0.00	0.00	0
accuracy			0.38	61550
macro avg	0.19	0.13	0.12	61550
weighted avg	0.68	0.38	0.38	61550

Combine Devices

Some time periods are missing data in certain devices, resulting in numerous Nan values in the ‘Samar glasses' features.

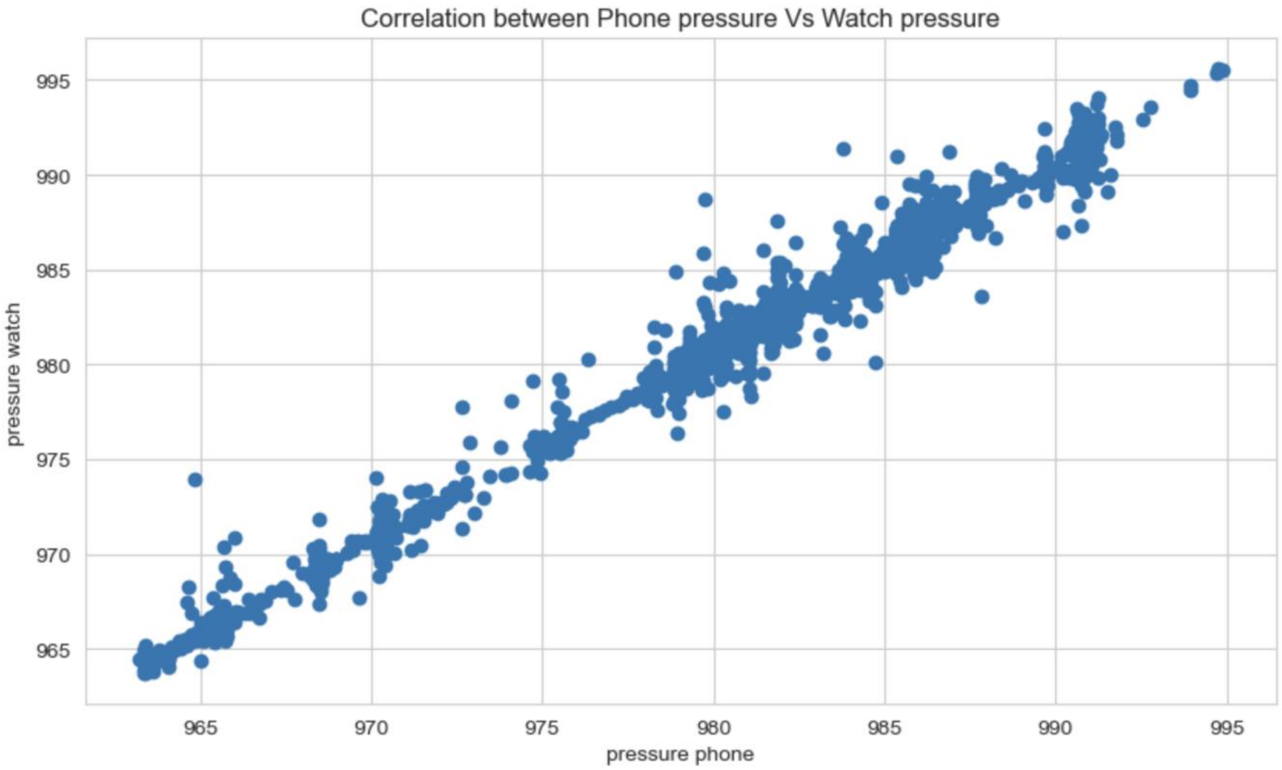
Consequently, we have chosen to merge only the 'smartwatch' and 'smartphone' features.



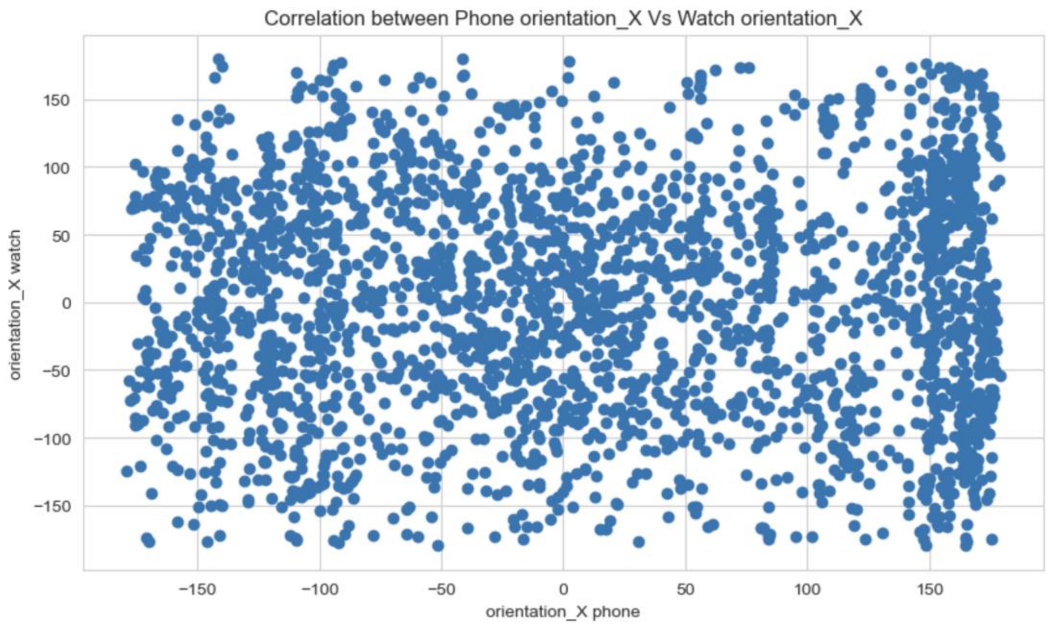

```
Phone percentage of nan: 0.3957699330493001
Watch percentage of nan: 0.2559342665855143
Glasses percentage of nan: 0.8420572124163116
```

Combine Devices

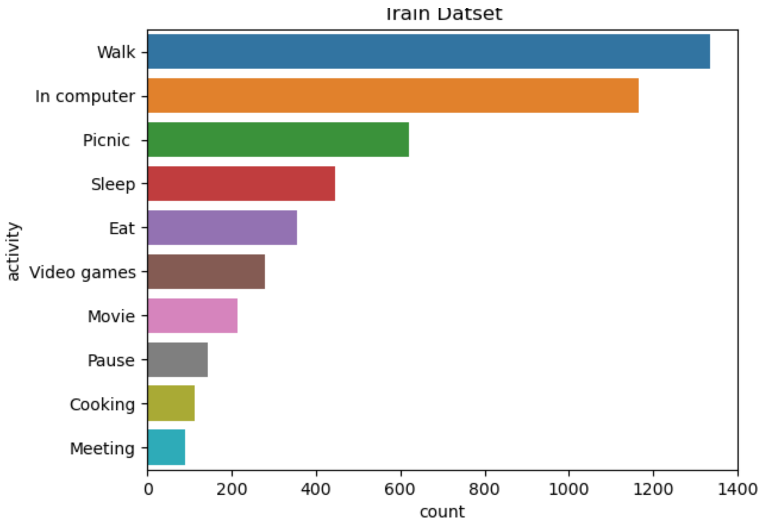
Correlation between Smartphone's data and Smartwatch's data in the 'pressure' feature

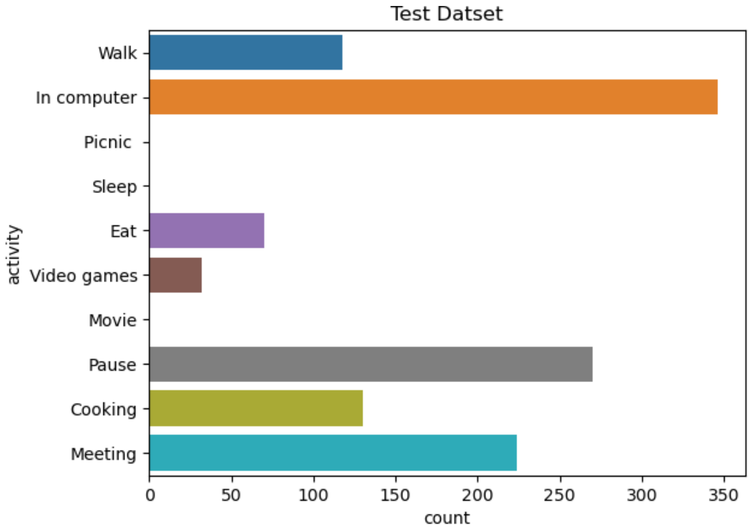


The rest of the features were not correlated



Performance Comparison – Smart Watch & Smart Phone





Confusion Matrix

```
[[  0  57   4  15   1   0   3   0  50]
 [  0  17  17   0  13   0   0   0  23]
 [  5  57 256   0   1   0   0   0  27]
 [  0  17 151   0   6   0   0  21  29]
 [  0   0   0   0   0   0   0   0   0]
 [  0  55  33  12   0  35 105   0  30]
 [  0   0   0   0   0   0   0   0   0]
 [  0   0  32   0   0   0   0   0   0]
 [  0  24  64   0  14   1   0   4  11]]
```

Classification Report

	precision	recall	f1-score	support
Cooking	0.00	0.00	0.00	130
Eat	0.07	0.24	0.11	70
In computer	0.46	0.74	0.57	346
Meeting	0.00	0.00	0.00	224
Movie	0.00	0.00	0.00	0
Pause	0.97	0.13	0.23	270
Sleep	0.00	0.00	0.00	0
Video games	0.00	0.00	0.00	32
Walk	0.06	0.09	0.08	118
accuracy			0.27	1190
macro avg	0.17	0.13	0.11	1190
weighted avg	0.37	0.27	0.23	1190

Suggestions for Improving the Model's Performance

- **Expand the Dataset**: Prioritize data collection efforts for labels that have limited representation in the current dataset. Note that data collection spans over two weeks, but there are gaps in the documentation, and not all labels are consistently observed across all devices.

- **Explore Complex Models**: Consider the adoption of more sophisticated models, such as deep learning architectures, to potentially capture intricate patterns within the data.

- **Experiment with Sampling Times**: Investigate the impact of different sampling time intervals on model performance to optimize data collection strategies.

- **Utilize Semi-Supervised Learning**: Leverage semi-supervised learning approaches to label measurements within the dataset that don't have labels, maximizing the utility of available data.

- **Incorporate Time-Aware Features**: Enhance the feature set by introducing new variables that account for temporal aspects, which can potentially enrich the model's ability to understand and predict time-dependent patterns

Summary

- **Features -** Extensive work for feature engineering and feature understanding
- **Visualizations** – some support our results and analysis
- **Models** – fit the best model for the data
- **Performance** – best model's performance overall analysis
- Further suggestions to improve our project