D4.5: Long-term behaviours analysis prototype

Dissemination level: Public

Document type: Demonstrator

Version: 1.0.0

Date: June 3rd, 2019

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement #769553. This result only reflects the author's view and the EU is not responsible for any use that may be made of the information it contains.

Document Details

Project Number	769553
Project title	Council of Coaches
Title of deliverable	Long-term behaviours analysis prototype
Due date of deliverable	May 31, 2019 (M21)
Work package	WP4
Author(s)	Oresti Banos (CMC), Kostas Konsolakis (CMC)
Reviewer(s)	Álvaro Fides (UPV)
Approved by	Coordinator
Dissemination level	Public
Document type	Demonstrator
Total number of pages	29

Partners

- University of Twente – Centre for Monitoring and Coaching (CMC)
- Roessingh Research and Development (RRD)
- Danish Board of Technology Foundation (DBT)
- Sorbonne University (SU)
- University of Dundee (UDun)
- Universitat Politècnica de València, Grupo SABIEN (UPV)
- Innovation Sprint (iSPRINT)

Abstract

This deliverable (D4.5) describes the software implementation of the methods developed in D4.4 for the inference of long-term behaviours based on short-term behaviours, the latter already presented in D4.2. This deliverable is further accompanied with a number of different scenarios for identifying long-term behaviours.

Table of Contents

List of figures

Symbols, abbreviations and acronyms

AR	Autoregression
ARIMA	Autoregressive Integrated Moving Average
CMC	Centre for Monitoring and Coaching
COUCH	Council of Coaches
D	Deliverable
EC	European Commission
HBAF	Holistic Behaviour Analysis Framework
HCI	Human-Computer Interaction
M	Month
MS	Milestone
RRD	Roessingh Research and Development
UPMC	Université Pierre et Marie Curie, Paris 6
UPV	Universitat Politècnica de València
UT	University of Twente

1 Introduction

After presenting the methods used for measuring and modelling short-term and long-term behaviours in D4.2 and D4.4 respectively, this deliverable deepens more into the software components needed for the characterisation and identification of long-term behaviour. In particular, D4.5 aims to implement and demonstrate some of the long-term behaviour characterisation and identification methods proposed in D4.4.

Initially, the technical setup including the hardware components and the system's architecture is presented in Section 3. Then, the developed models for long-term physical and social behaviours are implemented in Section 4. Finally, Section 5 demonstrates various scenarios for identifying long-term behaviours.

2 Objectives

The main objective of this deliverable (D4.5) is to describe the software components developed for the inference of long-term behaviours based on short-term behaviour information (D4.4).

3 Technical Setup

3.1 Hardware

The hardware setup is based on the same approach that we presented in D4.3 (please refer to Section 3.1 of that deliverable for a full description). Specifically, on-body and off-body sensor data such as acceleration, phone call logs or ambient sound are collected and stored into the Holistic Behaviour Analysis Framework (HBAF) in order to detect physical, social, emotional and cognitive short-term behaviours. Based on these behaviours, a number of features are extracted for identifying physical, social, emotional and cognitive long-term behaviours over different periods of time (e.g., day, week, month).

3.2 Software

One of the key points of the HBAF is to analyse multimodal sensor data and extract useful information to describe the user's behaviour. In that spirit, both short-term and long-term behaviours are detected and constantly pushed through a secured connection to the shared knowledge base for further analysis. Thus, the provided information can be used in order to feed the dialogues among coaches and track behaviour changes over time. The complete operation flow of the HBAF is presented in the following picture (see Figure 1).

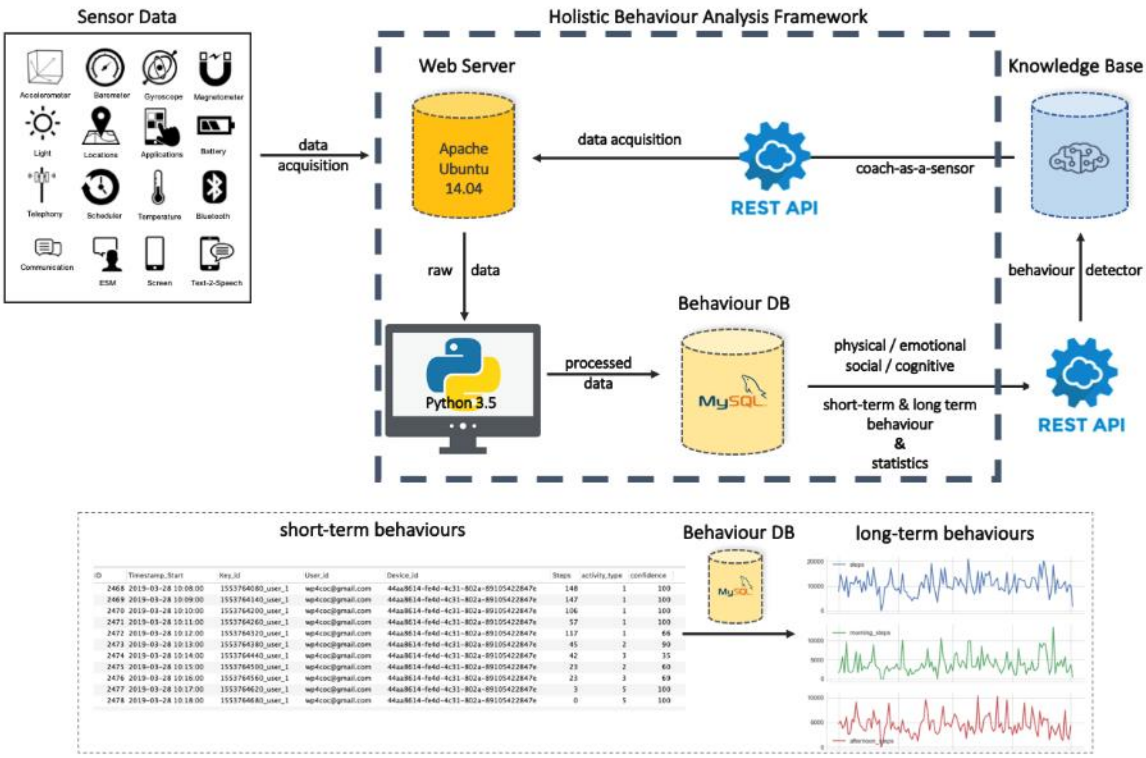

Figure 1: Operation flow of the HBAF (from raw sensor data to long-term behaviours).

As it was first introduced in D4.3, the operation flow, which is extended here, unwraps as follows. Smartphone raw data such as accelerometer, GPS, Bluetooth and ambient noise are recorded and stored into the Apache Web server on a Linux system (Ubuntu 14.04.5 LTS Trusty Tahr, 2018). Then, scripts are developed in Python 3.5 (Python, 2015) in order to process the raw data and detect short-term and long-term behaviours. At first, short-term behaviour models detect physical, social, emotional and cognitive behaviours, which are stored into a MySQL 5.7 database (MySQL, 2018). Additionally, the detected short-term behaviours are further analysed in order to extract relevant statistics regarding user's behaviour (e.g., the total or average number of steps per hour/day, the most performed activities, etc.). These statistics are used in order to extract a number of different features over a longer period of time, which

in turn can help outlining the behaviour of the user for the long run. Then, the shared knowledge base gets permanent access to this database via a secured rest API connection (Slim 3.9.0 released, 2017), based on JSON web token authentication. The HBAF communicates with the shared knowledge base through the cURL web2web communication protocol. The data are distributed through JSON format according to RFC 8259 (Datatracker, 2017) and ECMA-404 (ecma-international, 2017). It is worth mentioning that the shared knowledge base component can get access to the HBAF and post data extracted from the dialogs between user and coach (coach as a sensor) for further analysis.

In the following figures, we illustrate some exemplary data frames resulting from the characterization (feature extraction) performed on short-term behaviours. For instance, we present the features used to identify long-term physical and social behaviours. For the short-term physical behaviours, including the detected activities and the total number of steps, we compute a number of features per day (Figure 2), per week (Figure 3) and per month (Figure 4). It is worth mentioning that for counting steps, we used data from a Fitbit device. In Figure 5 we compute the features based on the total duration of the performed activities, while in Figure 6 and Figure 7 we do so by clustering first the performed activities into vigorous (walking, cycling and tilting) or sedentary (commuting and being still). Similarly, we compute the features for the social behaviour based on the total duration of being socially engaged (Figure 8) or socially isolated (Figure 9). It is also worth mentioning that for detecting the total duration of the performed activities and the duration of being socially engaged, we used a smartphone device to collect raw smartphone data and detect the short-term behaviours (further explanation of the dataset is presented in Section 5).

steps_count_day - DataFrame

Index	steps	start_day_steps	end_day_steps	day_steps	morning_steps	afternoon_steps	evening_steps	daytime_steps	work_steps	indoors_steps	outdoors_steps	diff1_steps	diff2_steps	ratio_steps
2019-01-01 00:00:00	7871	2019-01-01 00:00:00	2019-01-02 00:00:00	Tuesday	1590	4755	1297	7219	5844	3673	4198	nan	6781	nan
2019-01-02 00:00:00	14652	2019-01-02 00:00:00	2019-01-03 00:00:00	Wednesday	2726	5277	6318	14354	5999	9962	4690	6781	3776	5278.5
2019-01-03 00:00:00	10876	2019-01-03 00:00:00	2019-01-04 00:00:00	Thursday	5846	3804	1859	10824	8279	5500	5376	3776	2786	3281
2019-01-04 00:00:00	8090	2019-01-04 00:00:00	2019-01-05 00:00:00	Friday	1595	4306	1589	7920	5041	6176	1914	2786	5792	4289
2019-01-05 00:00:00	13882	2019-01-05 00:00:00	2019-01-06 00:00:00	Saturday	9603	6185	504	13882	nan	2822	11060	5792	8285	7038.5
2019-01-06 00:00:00	5597	2019-01-06 00:00:00	2019-01-07 00:00:00	Sunday	2069	2613	1349	5597	nan	2556	3041	8285	5759	7022
2019-01-07 00:00:00	11356	2019-01-07 00:00:00	2019-01-08 00:00:00	Monday	1654	2740	6059	10895	3285	3522	7834	5759	151	2955
2019-01-08 00:00:00	11205	2019-01-08 00:00:00	2019-01-09 00:00:00	Tuesday	1872	5679	2176	10938	6195	7861	3344	151	421	286
2019-01-09 00:00:00	10784	2019-01-09 00:00:00	2019-01-10 00:00:00	Wednesday	3012	5135	2067	10435	7052	5278	5506	421	3176	1798.5
2019-01-10 00:00:00	13960	2019-01-10 00:00:00	2019-01-11 00:00:00	Thursday	7240	9181	1603	13754	12151	8660	5300	3176	2869	3022.5
2019-01-11 00:00:00	11091	2019-01-11 00:00:00	2019-01-12 00:00:00	Friday	1662	6261	2541	10821	6928	7504	3587	2869	1443	2156
2019-01-12 00:00:00	12534	2019-01-12 00:00:00	2019-01-13 00:00:00	Saturday	6554	4446	2250	12534	nan	10062	2472	1443	5064	3253.5
2019-01-13 00:00:00	7470	2019-01-13 00:00:00	2019-01-14 00:00:00	Sunday	2202	4065	1698	7406	nan	5203	2267	5064	4383	4723.5

Figure 2: Feature extraction for detecting long-term physical behaviour based on steps per day.

steps_count_week - DataFrame

Index	steps	start_day_steps	end_day_steps	week_steps	morning_steps	afternoon_steps	evening_steps	daytime_steps	work_steps	weekdays_steps	weekends_steps	indoors_steps	outdoors_steps	diff1_steps	diff2_steps	ratio_steps
2019-01-06 00:00:00	60968	2018-12-31 00:00:00	2019-01-06 00:00:00	1	23429	26940	12916	59796	25163	41489	19479	30689	30279	nan	17432	nan
2019-01-13 00:00:00	78400	2019-01-07 00:00:00	2019-01-13 00:00:00	2	24196	37507	18394	76783	35611	58396	20004	48090	30310	17432	7678	12555
2019-01-20 00:00:00	86078	2019-01-14 00:00:00	2019-01-20 00:00:00	3	29965	34973	29036	85407	32246	56854	29224	39645	46433	7678	35593	21635.5
2019-01-27 00:00:00	50485	2019-01-21 00:00:00	2019-01-27 00:00:00	4	13392	29176	16558	50077	16864	29331	21154	27451	23034	35593	25296	30444.5
2019-02-03 00:00:00	75781	2019-01-28 00:00:00	2019-02-03 00:00:00	5	27457	28595	20743	74321	28252	53958	21823	40321	35460	25296	33958	29627
2019-02-10 00:00:00	41823	2019-02-04 00:00:00	2019-02-10 00:00:00	6	13620	19642	6612	40463	21285	31183	10640	16969	24854	33958	29026	31492
2019-02-17 00:00:00	70849	2019-02-11 00:00:00	2019-02-17 00:00:00	7	26572	36628	13141	69939	34962	50381	20468	24372	46477	29026	6855	17940.5
2019-02-24 00:00:00	77704	2019-02-18 00:00:00	2019-02-24 00:00:00	8	30053	32953	18082	76659	28243	49960	27744	41203	36501	6855	2962	4908.5
2019-03-03 00:00:00	80666	2019-02-25 00:00:00	2019-03-03 00:00:00	9	26795	36202	21309	78747	29526	54242	26424	38899	41767	2962	4173	3567.5

Figure 3: Feature extraction for detecting long-term physical behaviour based on steps per week.

steps_count_month - DataFrame

Index	steps	start_day_steps	end_day_steps	month_steps	morning_steps	afternoon_steps	evening_steps	daytime_steps	work_steps	weekdays_steps	weekends_steps	indoors_steps	outdoors_steps	diff1_steps	diff2_steps	ratio_steps
2019-01-31 00:00:00	320673	2019-01-01 00:00:00	2019-01-31 00:00:00	1	103353	144760	90701	315919	133501	230812	89861	170023	150650	nan	53706	nan
2019-02-28 00:00:00	266967	2019-02-01 00:00:00	2019-02-28 00:00:00	2	97032	119612	58847	261260	113198	186292	80675	121243	145724	53706	83022	68364
2019-03-31 00:00:00	349989	2019-03-01 00:00:00	2019-03-31 00:00:00	3	138634	152910	90021	345088	139157	223310	126679	156775	193214	83022	2573	42797.5
2019-04-30 00:00:00	347416	2019-04-01 00:00:00	2019-04-30 00:00:00	4	128723	126130	55441	314133	170478	267296	80120	198943	148473	2573	335319	168946
2019-05-31 00:00:00	12097	2019-05-01 00:00:00	2019-05-31 00:00:00	5	4169	4410	1690	9512	5996	12097	0	4837	7260	335319	nan	nan

Figure 4: Feature extraction for detecting long-term physical behaviour based on steps per month.

activity_day - DataFrame

Index	total_duration	Vehicle	Cycling	Walking	Still	Tilting	start_day_activity	end_day_activity	day_activity	morning_activity	afternoon_activity	evening_activity	daytime_activity	work_activity	indoors_activity	outdoors_activity	diff1_activity	diff2_activity	ratio_activity
2019-04-11 00:00:00	270	0	1	76	149	44	2019-04-11 00:00:00	2019-04-12 00:00:00	Thursday	25	126	119	270	133	132	138	nan	82	nan
2019-04-12 00:00:00	352	0	17	71	207	57	2019-04-12 00:00:00	2019-04-13 00:00:00	Friday	101	46	177	351	145	172	180	82	117	99.5
2019-04-13 00:00:00	235	0	0	24	165	46	2019-04-13 00:00:00	2019-04-14 00:00:00	Saturday	69	57	94	231	nan	116	119	117	333	225
2019-04-14 00:00:00	568	194	0	32	243	99	2019-04-14 00:00:00	2019-04-15 00:00:00	Sunday	114	238	197	567	nan	289	279	333	62	197.5
2019-04-15 00:00:00	506	3	2	131	264	106	2019-04-15 00:00:00	2019-04-16 00:00:00	Monday	139	169	174	503	282	244	262	62	34	48
2019-04-16 00:00:00	540	41	0	83	320	96	2019-04-16 00:00:00	2019-04-17 00:00:00	Tuesday	125	192	196	530	292	274	266	34	21	27.5
2019-04-17 00:00:00	561	115	0	126	216	104	2019-04-17 00:00:00	2019-04-18 00:00:00	Wednesday	134	231	167	548	311	261	300	21	35	28
2019-04-18 00:00:00	526	80	1	124	220	101	2019-04-18 00:00:00	2019-04-19 00:00:00	Thursday	129	193	170	522	278	261	265	35	69	52
2019-04-19 00:00:00	595	180	6	84	250	75	2019-04-19 00:00:00	2019-04-20 00:00:00	Friday	126	199	248	593	285	293	302	69	29	49
2019-04-20 00:00:00	566	280	0	22	173	91	2019-04-20 00:00:00	2019-04-21 00:00:00	Saturday	131	240	168	564	nan	285	281	29	239	134
2019-04-21 00:00:00	327	13	28	13	222	51	2019-04-21 00:00:00	2019-04-22 00:00:00	Sunday	46	83	188	325	nan	157	170	239	120	179.5

Figure 5: Feature extraction for detecting long-term physical behaviour based on the total duration of the performed activities per day.

vigorous_day - DataFrame

Index	vigorous_duration	Cycling	Walking	Tilting	start_day_vigorous	end_day_vigorous	day_vigorous	morning_vigorous	afternoon_vigorous	evening_vigorous	daytime_vigorous	work_vigorous	indoors_vigorous	outdoors_vigorous	diff1_vigorous	diff2_vigorous	ratio_vigorous
2019-04-11 00:00:00	121	1	76	44	2019-04-11 00:00:00	2019-04-12 00:00:00	Thursday	8	53	60	121	60	51	70	nan	24	nan
2019-04-12 00:00:00	145	17	71	57	2019-04-12 00:00:00	2019-04-13 00:00:00	Friday	44	14	81	145	58	59	86	24	75	49.5
2019-04-13 00:00:00	70	0	24	46	2019-04-13 00:00:00	2019-04-14 00:00:00	Saturday	14	17	35	70	nan	29	41	75	61	68
2019-04-14 00:00:00	131	0	32	99	2019-04-14 00:00:00	2019-04-15 00:00:00	Sunday	24	49	54	131	nan	67	64	61	108	84.5
2019-04-15 00:00:00	239	2	131	106	2019-04-15 00:00:00	2019-04-16 00:00:00	Monday	88	81	69	239	166	122	117	108	60	84
2019-04-16 00:00:00	179	0	83	96	2019-04-16 00:00:00	2019-04-17 00:00:00	Tuesday	66	73	35	179	136	90	89	60	51	55.5
2019-04-17 00:00:00	230	0	126	104	2019-04-17 00:00:00	2019-04-18 00:00:00	Wednesday	73	116	36	226	148	121	109	51	4	27.5
2019-04-18 00:00:00	226	1	124	101	2019-04-18 00:00:00	2019-04-19 00:00:00	Thursday	96	74	54	226	157	116	110	4	61	32.5
2019-04-19 00:00:00	165	6	84	75	2019-04-19 00:00:00	2019-04-20 00:00:00	Friday	46	56	60	165	86	80	85	61	52	56.5
2019-04-20 00:00:00	113	0	22	91	2019-04-20 00:00:00	2019-04-21 00:00:00	Saturday	18	47	39	113	nan	50	63	52	21	36.5
2019-04-21 00:00:00	92	28	13	51	2019-04-21 00:00:00	2019-04-22 00:00:00	Sunday	5	28	57	92	nan	48	44	21	119	70

Figure 6: Feature extraction for detecting long-term physical behaviour based on the total duration of the performed vigorous activities per day.

sedentary_day - DataFrame

Index	sedentary_duration	Vehicle	Still	start_day_sedentary	end_day_sedentary	day_sedentary	morning_sedentary	afternoon_sedentary	evening_sedentary	daytime_sedentary	work_sedentary	indoors_sedentary	outdoors_sedentary	diff1_sedentary	diff2_sedentary	ratio_sedentary
2019-04-11 00:00:00	149	0	149	2019-04-11 00:00:00	2019-04-12 00:00:00	Thursday	17	73	59	149	73	77	72	nan	58	nan
2019-04-12 00:00:00	207	0	207	2019-04-12 00:00:00	2019-04-13 00:00:00	Friday	57	32	96	206	87	97	110	58	42	50
2019-04-13 00:00:00	165	0	165	2019-04-13 00:00:00	2019-04-14 00:00:00	Saturday	55	40	59	161	nan	91	74	42	272	157
2019-04-14 00:00:00	437	194	243	2019-04-14 00:00:00	2019-04-15 00:00:00	Sunday	90	189	143	436	nan	223	214	272	170	221
2019-04-15 00:00:00	267	3	264	2019-04-15 00:00:00	2019-04-16 00:00:00	Monday	51	88	105	264	116	127	140	170	94	132
2019-04-16 00:00:00	361	41	320	2019-04-16 00:00:00	2019-04-17 00:00:00	Tuesday	59	119	161	351	156	181	180	94	30	62
2019-04-17 00:00:00	331	115	216	2019-04-17 00:00:00	2019-04-18 00:00:00	Wednesday	61	115	131	322	163	167	164	30	31	30.5
2019-04-18 00:00:00	300	80	220	2019-04-18 00:00:00	2019-04-19 00:00:00	Thursday	33	119	116	296	121	148	152	31	130	80.5
2019-04-19 00:00:00	430	180	250	2019-04-19 00:00:00	2019-04-20 00:00:00	Friday	80	143	188	428	199	217	213	130	23	76.5
2019-04-20 00:00:00	453	280	173	2019-04-20 00:00:00	2019-04-21 00:00:00	Saturday	113	193	129	451	nan	233	220	23	218	120.5
2019-04-21 00:00:00	235	13	222	2019-04-21 00:00:00	2019-04-22 00:00:00	Sunday	41	55	131	233	nan	120	115	218	1	109.5

Figure 7: Feature extraction for detecting long-term physical behaviour based on the total duration of the performed sedentary activities per day.

social_interaction_day - DataFrame

Index	Interaction	start_day_interaction	end_day_interaction	day_interaction	morning_interaction	afternoon_interaction	evening_interaction	daytime_interaction	work_interaction	indoors_interaction	outdoors_interaction	diff1_interaction	diff2_interaction	ratio_interaction
2019-04-11 00:00:00	114	2019-04-11 00:00:00	2019-04-12 00:00:00	Thursday	0	59	55	114	47	55	59	nan	50	nan
2019-04-12 00:00:00	164	2019-04-12 00:00:00	2019-04-13 00:00:00	Friday	30	10	99	147	40	77	87	50	72	61
2019-04-13 00:00:00	92	2019-04-13 00:00:00	2019-04-14 00:00:00	Saturday	34	30	28	92	nan	48	44	72	236	154
2019-04-14 00:00:00	328	2019-04-14 00:00:00	2019-04-15 00:00:00	Sunday	21	213	79	312	nan	170	158	236	213	224.5
2019-04-15 00:00:00	115	2019-04-15 00:00:00	2019-04-16 00:00:00	Monday	46	27	40	115	73	61	54	213	62	137.5
2019-04-16 00:00:00	177	2019-04-16 00:00:00	2019-04-17 00:00:00	Tuesday	42	104	31	177	131	91	86	62	115	88.5
2019-04-17 00:00:00	292	2019-04-17 00:00:00	2019-04-18 00:00:00	Wednesday	35	160	92	291	194	147	145	115	116	115.5
2019-04-18 00:00:00	176	2019-04-18 00:00:00	2019-04-19 00:00:00	Thursday	17	75	84	176	85	89	87	116	115	115.5
2019-04-19 00:00:00	291	2019-04-19 00:00:00	2019-04-20 00:00:00	Friday	56	108	127	291	139	147	144	115	121	118
2019-04-20 00:00:00	412	2019-04-20 00:00:00	2019-04-21 00:00:00	Saturday	120	222	67	412	nan	192	220	121	302	211.5
2019-04-21 00:00:00	110	2019-04-21 00:00:00	2019-04-22 00:00:00	Sunday	20	33	57	110	nan	54	56	302	35	168.5

Figure 8: Feature extraction for detecting long-term social behaviour based on the total duration of being socially engaged per day.

social_isolation_day - DataFrame

Index	isolation	start_day_isolation	end_day_isolation	day_isolation	morning_isolation	afternoon_isolation	evening_isolation	daytime_isolation	work_isolation	indoors_isolation	outdoors_isolation	diff1_isolation	diff2_isolation	ratio_isolation
2019-04-11 00:00:00	648	2019-04-11 00:00:00	2019-04-12 00:00:00	Thursday	43	242	365	648	236	312	336	nan	628	nan
2019-04-12 00:00:00	1276	2019-04-12 00:00:00	2019-04-13 00:00:00	Friday	211	291	322	874	441	636	640	628	72	350
2019-04-13 00:00:00	1348	2019-04-13 00:00:00	2019-04-14 00:00:00	Saturday	207	271	393	929	nan	692	656	72	236	154
2019-04-14 00:00:00	1112	2019-04-14 00:00:00	2019-04-15 00:00:00	Sunday	220	88	342	709	nan	552	560	236	213	224.5
2019-04-15 00:00:00	1325	2019-04-15 00:00:00	2019-04-16 00:00:00	Monday	195	274	381	906	408	665	660	213	62	137.5
2019-04-16 00:00:00	1263	2019-04-16 00:00:00	2019-04-17 00:00:00	Tuesday	199	197	390	844	350	617	646	62	115	88.5
2019-04-17 00:00:00	1148	2019-04-17 00:00:00	2019-04-18 00:00:00	Wednesday	206	141	329	730	287	566	582	115	116	115.5
2019-04-18 00:00:00	1264	2019-04-18 00:00:00	2019-04-19 00:00:00	Thursday	224	226	337	845	396	655	609	116	115	115.5
2019-04-19 00:00:00	1149	2019-04-19 00:00:00	2019-04-20 00:00:00	Friday	185	193	294	730	342	578	571	115	121	118
2019-04-20 00:00:00	1028	2019-04-20 00:00:00	2019-04-21 00:00:00	Saturday	121	79	354	609	nan	529	499	121	302	211.5
2019-04-21 00:00:00	1330	2019-04-21 00:00:00	2019-04-22 00:00:00	Sunday	221	268	364	911	nan	656	674	302	35	168.5

Figure 9: Feature extraction for detecting long-term social behaviour based on the total duration of being socially isolated per day.

4 Scripts for Long-term Behaviour Identification

The following script describes in detail the logic developed for extracting features that can be used to characterise the long-term behaviours. Hence, different features are extracted from the physical, social, emotional and cognitive short-term behaviours. Even though in this deliverable we are focusing only on physical and social long-term behaviours, the same approach will be used for identifying emotional and cognitive long-term behaviours. A detailed list is presented in D4.4 for all behaviours (see D4.4, Tables 3-7 in that deliverable).

```
# ============================================================================
#  Long Term Behaviours: Features Extraction Model
# ============================================================================

import pandas as pd
import numpy as np
from pandas import datetime
import os.path
import matplotlib.pyplot as plt

# ============================================================================
# Features Extraction: Physical Behaviour (Steps)
# ============================================================================
# Physical Activity DataFrame created for D4.3 and contains the short-term physical behaviour (steps and performed activities)
# Location DataFrame contains the indoors/outdoors location
steps_count_day, steps_count_week, steps_count_month = features_df(Physical_Activity[['steps']], Location, key='steps')

# ============================================================================
# Features Extraction: Physical Behaviour (total_duration of all activities)
# ============================================================================
# Physical Activity DataFrame created for D4.3 and contains the short-term physical behaviour (steps and performed activities)
Activity_features = Physical_Activity['activity_type'].copy()
Activity_features['total_duration'] = Activity_features['Vehicle'] + Activity_features['Cycling'] + Activity_features['Walking'] + Activity_features['Still'] + Activity_features['Tilting']

activity_day, activity_week, activity_month = features_df(Activity_features[['total_duration', 'Vehicle', 'Cycling', 'Walking', 'Still', 'Tilting']], Location, key='activity')

# ============================================================================
# Features Extraction: Physical Behaviour (total_duration of SEDENTARY activities: being still & vehicle)
# ============================================================================
Activity_features['sedentary_duration'] = Activity_features['Vehicle'] + Activity_features['Still']
sedentary_day, sedentary_week, sedentary_month = features_df(Activity_features[['sedentary_duration', 'Vehicle', 'Still']], Location, key='sedentary')

# ============================================================================
# Features Extraction: Physical Behaviour (total_duration of VIGOROUS activities: cycling, walking & tilting)
# ============================================================================
Activity_features['vigorous_duration'] = Activity_features['Cycling'] + Activity_features['Walking'] + Activity_features['Tilting']
vigorous_day, vigorous_week, vigorous_month = features_df(Activity_features[['vigorous_duration', 'Cycling', 'Walking', 'Tilting']], Location, key='vigorous')

# ============================================================================
# Features Extraction: Social Behaviour (Social Interaction)
# ============================================================================
# Social Activity DataFrame created for D4.3 and contains the short-term social behaviour
```

```
Social_features = Social_Activity[['Detected_Social']].copy()
Social_features['interaction'] = 0
Social_features['isolation'] = 0
Social_features.loc[(Social_features['Detected_Social'] == 1), ['interaction'] ] = 1
Social_features.loc[(Social_features['Detected_Social'] == 0), ['isolation'] ] = 1
interaction_day, interaction_week, interaction_month = features_df(Social_features[['interaction']], Location, key='interaction')

# ==========================================================================
# Features Extraction: Social Behaviour (Social Isolation)
# ==========================================================================
isolation_day, isolation_week, isolation_month = features_df(Social_features[['isolation']], Location, key='isolation')
```

The following script contains the functions that were used in the above code in order to extract the features for identifying physical, social, emotional and cognitive long-term behaviours. These features are namely: *morning, afternoon, evening, daytime, work, weekdays, weekends, indoors, outdoors and ratio*. The ratio feature is defined as the sum of the differences of a value with the values from the previous and following period (window segment), divided by two (more information can be found in code lines 61-84).

```
1.  # ==========================================================================
2.  # Functions for Features Extraction
3.  # ==========================================================================
4.  
5.  # number of values during morning (8am-12pm)
6.  def feat_morning(df):
7.      df_morning = df.between_time('08:00', '12:00')
8.      df_morning = df_morning.resample('1D').sum()
9.      df_morning.columns = ['morning']
10.     return df_morning
11. 
12. # number of values during afternoon (12pm -5pm)
13. def feat_afternoon(df):
14.     df_afternoon = df.between_time('12:00', '17:00')
15.     df_afternoon = df_afternoon.resample('1D').sum() #calculate the sum of values per day
16.     df_afternoon.columns = ['afternoon']
17.     return df_afternoon
18. 
19. # number of values during evening (5pm -12am)
20. def feat_evening(df):
21.     df_evening = df.between_time('17:00', '00:00')
22.     df_evening = df_evening.resample('1D').sum() #calculate the sum of values per day
23.     df_evening.columns = ['evening']
24.     return df_evening
25. 
26. # number of values during daytime (7am -12am)
27. def feat_daytime(df):
28.     df_daytime = df.between_time('07:00', '00:00')
29.     df_daytime = df_daytime.resample('1D').sum() #calculate the sum of values per day
30.     df_daytime.columns = ['daytime']
31.     return df_daytime
32. 
33. # number of values during worktime (8am -4pm) and during weekdays / generalized approach
34. def feat_work(df):
35.     df_work = df.between_time('08:00', '16:00')
36.     df_work = df_work.resample('1D').sum() #calculate the sum of values per day
37.     df_work.columns = ['work']
38.     df_work['day']= df_work.index.strftime('%A')
39.     df_work.loc[(df_work['day'] == 'Saturday') | (df_work['day'] == 'Sunday'), ['work'] ] = np.nan
40.     return df_work['work']
41. 
```

```
# number of values indoors / location condition 0 for indoors & 1 for outdoors
def feat_indoors(df,location):
    df.columns = ['value']
    df_indoors = df.copy()
    df_indoors.loc[location['location']==1, ['value'] ] = 0
    df_indoors = df_indoors.resample('1D').sum() #calculate the sum of values per day
    df_indoors.columns = ['indoors']
    return df_indoors

# number of values outdoors / location condition 0 for indoors & 1 for outdoors
def feat_outdoors(df,location):
    df.columns = ['value']
    df_outdoors = df.copy()
    df_outdoors.loc[location['location']==0, ['value'] ] = 0
    df_outdoors = df_outdoors.resample('1D').sum() #calculate the sum of values per day
    df_outdoors.columns = ['outdoors']
    return df_outdoors

# difference of values from one period compared to the previous one
def feat_diff1(df,freq):
    if freq ==1:
        df_diff1 = df.resample('1D').sum() #calculate the sum of values per day
    if freq == 7:
        df_diff1 = df.resample('W').sum() #calculate the sum of values per week
    if freq == 30:
        df_diff1 = df.resample('M').sum() #calculate the sum of values per month
    df_diff1.columns = ['diff1']
    df_diff1 = df_diff1.diff()
    df_diff1 = df_diff1.abs()
    return df_diff1

# difference of values from one period compared to the next one
def feat_diff2(df,freq):
    if freq ==1:
        df_diff2 = df.resample('1D').sum() #calculate the sum of values per day
    if freq == 7:
        df_diff2 = df.resample('W').sum()
    if freq == 30:
        df_diff2 = df.resample('M').sum()
    df_diff2.columns = ['diff2']
    df_diff2 = df_diff2.diff(periods=-1)
    df_diff2 = df_diff2.abs()
    return df_diff2

# number of values during weekdays
def feat_weekdays(df,freq):
    df_weekdays = df.resample('1D').sum() #calculate the sum of values per day
    df_weekdays.columns = ['weekdays']
    df_weekdays['day']= df_weekdays.index.strftime('%A')
    df_weekdays.loc[(df_weekdays['day'] == 'Saturday') | (df_weekdays['day'] == 'Sunday'), ['weekdays'] ] = np.nan
    if freq == 7:
        df_weekdays = df_weekdays.resample('W').sum()
    if freq == 30:
        df_weekdays = df_weekdays.resample('M').sum()
    return df_weekdays['weekdays']

# number of values during weekends
def feat_weekends(df, freq):
    df_weekends = df.resample('1D').sum() #calculate the sum of values per day
    df_weekends.columns = ['weekends']
    df_weekends['day']= df_weekends.index.strftime('%A')
    df_weekends.loc[(df_weekends['day'] != 'Saturday') & (df_weekends['day'] != 'Sunday'), ['weekends'] ] = np.nan
    if freq == 7:
        df_weekends = df_weekends.resample('W').sum()
```

```
    if freq == 30:
        df_weekends = df_weekends.resample('M').sum()
    return df_weekends['weekends']

def features_df (df, location, key):
    # Calculate features per day
    df_day = df.resample('1D').sum()
    df_day['start_day_%s'%key] = df_day.index
    df_day['end_day_%s'%key] = df_day.index + pd.Timedelta(days=1)
    df_day['day_%s'%key]= df_day['start_day_%s'%key].dt.weekday_name
    df_day['morning_%s'%key] = feat_morning(df.iloc[:,0].to_frame() )
    df_day['afternoon_%s'%key] = feat_afternoon(df.iloc[:,0].to_frame() )
    df_day['evening_%s'%key] = feat_evening(df.iloc[:,0].to_frame())
    df_day['daytime_%s'%key] = feat_daytime(df.iloc[:,0].to_frame())
    df_day['work_%s'%key] = feat_work(df.iloc[:,0].to_frame())
    df_day['indoors_%s'%key] = feat_indoors(df.iloc[:,0].to_frame(),location)
    df_day['outdoors_%s'%key] = feat_outdoors(df.iloc[:,0].to_frame(),location)
    df_day['diff1_%s'%key] = feat_diff1(df.iloc[:,0].to_frame(),freq=1)
    df_day['diff2_%s'%key] = feat_diff2(df.iloc[:,0].to_frame(),freq=1)
    df_day['ratio_%s'%key] = (df_day['diff1_%s'%key] + df_day['diff2_%s'%key])/2
    df_day['ratio_%s'%key] = df_day['ratio_%s'%key].abs()

    # Calculate features per week
    df_week = df.resample('W').sum()
    df_week['start_day_%s'%key] = df_week.index - pd.Timedelta(days=6)
    df_week['end_day_%s'%key] = df_week.index.copy()
    df_week['week_%s'%key]= df_week['start_day_%s'%key].dt.week
    df_week['morning_%s'%key] = df_day[['morning_%s'%key]].resample('W').sum()
    df_week['afternoon_%s'%key] = df_day[['afternoon_%s'%key]].resample('W').sum()
    df_week['evening_%s'%key] = df_day[['evening_%s'%key]].resample('W').sum()
    df_week['daytime_%s'%key] = df_day[['daytime_%s'%key]].resample('W').sum()
    df_week['work_%s'%key] = df_day[['work_%s'%key]].resample('W').sum()
    df_week['weekdays_%s'%key] = feat_weekdays(df_day.iloc[:,0].to_frame(name='weekdays_%s'%key) , freq=7)
    df_week['weekends_%s'%key] = feat_weekends(df_day.iloc[:,0].to_frame(name='weekends_%s'%key) , freq=7)
    df_week['indoors_%s'%key] = df_day[['indoors_%s'%key]].resample('W').sum()
    df_week['outdoors_%s'%key] = df_day[['outdoors_%s'%key]].resample('W').sum()
    df_week['diff1_%s'%key] = feat_diff1(df_day.iloc[:,0].to_frame(name='diff1_%s'%key) , freq=7)
    df_week['diff2_%s'%key] = feat_diff2(df_day.iloc[:,0].to_frame(name='diff2_%s'%key) , freq=7)
    df_week['ratio_%s'%key] = (df_week['diff1_%s'%key] + df_week['diff2_%s'%key])/2
    df_week['ratio_%s'%key] = df_week['ratio_%s'%key].abs()

    # Calculate features per month
    df_month = df.resample('M').sum()
    df_month['start_day_%s'%key] = df_month.index - pd.Timedelta(days=6)
    df_month['end_day_%s'%key] = df_month.index.copy()
    for i in (range(len(df_month))):
        if (i==0):
            df_month['start_day_%s'%key][i] = df_day['start_day_%s'%key][i]
        else:
            df_month['start_day_%s'%key][i] = df_month['end_day_%s'%key][i-1] + pd.Timedelta(days=1)
    df_month['month_%s'%key]= df_month['start_day_%s'%key].dt.month
    df_month['morning_%s'%key] = df_day[['morning_%s'%key]].resample('M').sum()
    df_month['afternoon_%s'%key] = df_day[['afternoon_%s'%key]].resample('M').sum()
    df_month['evening_%s'%key] = df_day[['evening_%s'%key]].resample('M').sum()
    df_month['daytime_%s'%key] = df_day[['daytime_%s'%key]].resample('M').sum()
    df_month['work_%s'%key] = df_day[['work_%s'%key]].resample('M').sum()
    df_month['weekdays_%s'%key] = feat_weekdays(df_day.iloc[:,0].to_frame(name='weekdays_%s'%key) ,freq=30)
    df_month['weekends_%s'%key] = feat_weekends(df_day.iloc[:,0].to_frame(name='weekends_%s'%key) ,freq=30)
    df_month['indoors_%s'%key] = df_day[['indoors_%s'%key]].resample('M').sum()
```

```
    df_month['outdoors_%s'%key] = df_day[['outdoors_%s'%key]].resample('M').sum()
    df_month['diff1_%s'%key] = feat_diff1(df_day.iloc[:,0].to_frame(name='diff1_%s'%key) , freq=30)
    df_month['diff2_%s'%key] = feat_diff2(df_day.iloc[:,0].to_frame(name='diff2_%s'%key) , freq=30)
    df_month['ratio_%s'%key] = (df_month['diff1_%s'%key] + df_month['diff2_%s'%key])/2
    df_month['ratio_%s'%key] = df_month['ratio_%s'%key].abs()

    return df_day, df_week, df_month
```

5 Demonstrator

In this section we aim to provide meaningful representations of the above figures (see Figure 2 through Figure 9) for identifying physical and social long-term behaviours. Thus, we present the second demonstrator (see link below) related to the inference of long-term behaviours.

https://www.youtube.com/watch?v=OkU0JFeDd5I

Specifically, we used a smartphone (Android device with the AWARE app installed) in order to collect data from a single subject (N=1) and detect physical and social short-term behaviours from 11 April 2019 to 8 May 2019. In parallel, we collected user's steps from a Fitbit device since 1 January 2019 (4 months in total). The idea of using the Fitbit for detecting steps, instead of the smartphone device (based on the short-term behaviour model for counting steps that we presented in D4.3 and D4.4), is because the type of information for counting steps is equivalent to what can be measured through smartphones. Furthermore, our intention is to include wearables such as Fitbit devices in the near future. It is worthwhile mentioning that the scope of this deliverable D4.5 is to show our preliminary approach for identifying long-term behaviours. Thus, emotional and cognitive long-term behaviours will be further examined and presented during T4.4.

At first, we plot the time series of the different datasets including the figures for counting steps per day (Figure 10), per week (Figure 11) and per month (Figure 12). It is noticeable that the time series contain irregularities due to noise and seasonality, and thus, they have to be converted to stationary in order to be further analysed through a forecasting model (e.g., ARIMA model). However, it is worth mentioning that the down trend that appears after April is caused by the limited amount of data (steps data were available up to 2/5/2019).

In order to evaluate the different features that we extracted and their distribution, we plot the boxplots for steps per day (Figure 13), week (Figure 14) and month (Figure 15). The '*daytime*' feature has a similar distribution with the '*steps*' feature, while most of the steps were performed during the weekdays and during the afternoons. Thus, these features have to be further analysed over different periods of time. However, we can assume that the user tends to a sedentary life during the weekends.

Trying to get further insights of the datasets, we use the swarm plots in order to identify the days of the week that are related to sedentary lifestyle. For instance, in Figure 16 we depict the number of steps grouped by each day, comparing the total number of steps with the steps performed during the morning, the afternoon and the evening. It can be seen that for almost every day of the week, the user has performed more than 10.000 steps per day (recommended number of steps per day based on D4.4), with some exceptions on Tuesdays and Wednesdays. In general, we can see in Figure 17 and Figure 18 that most of the steps have been performed during the afternoon hours, while the user did not manage to exceed the steps goal in February (month 2). On the other hand, the user achieved the maximum number of steps in April during week 16.

Furthermore, in Figure 19 and Figure 20 we present the total duration of the performed activities, compared to the sedentary and vigorous activities as an additional approach to identify long-term physical behaviour. It can be seen in these figures that the user tends to a sedentary lifestyle during weekends. Similarly, in Figure 21 and Figure 22 we present the total duration of being socially interacted and isolated per day and per week. Even though the current dataset does not suffice to identify the long-term social behaviour over time, we decided to compare the social level of a person with the activity level (see Figure 23 and Figure 24). For instance, we can see that as we move up and down the y-axis,

the sedentary activities seem to be a stronger determinant of the social level for being socially isolated. However, these assumptions need to be further validated with a time series forecasting method.

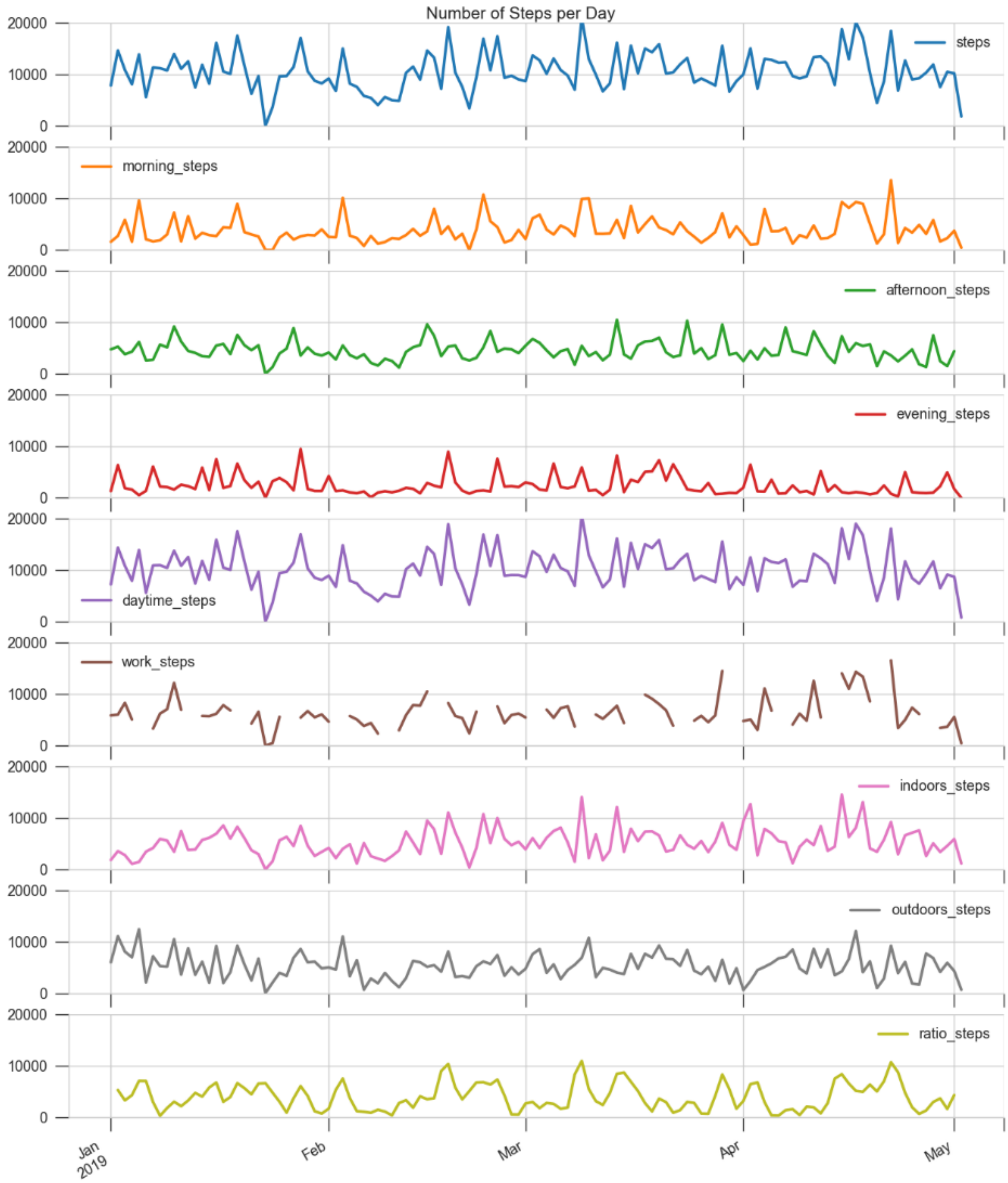

Figure 10: Daily step count over a 4-month period. The counts are marginalized over different parts of the day (morning, afternoon, evening, daytime, worktime) and location (indoors, outdoors).

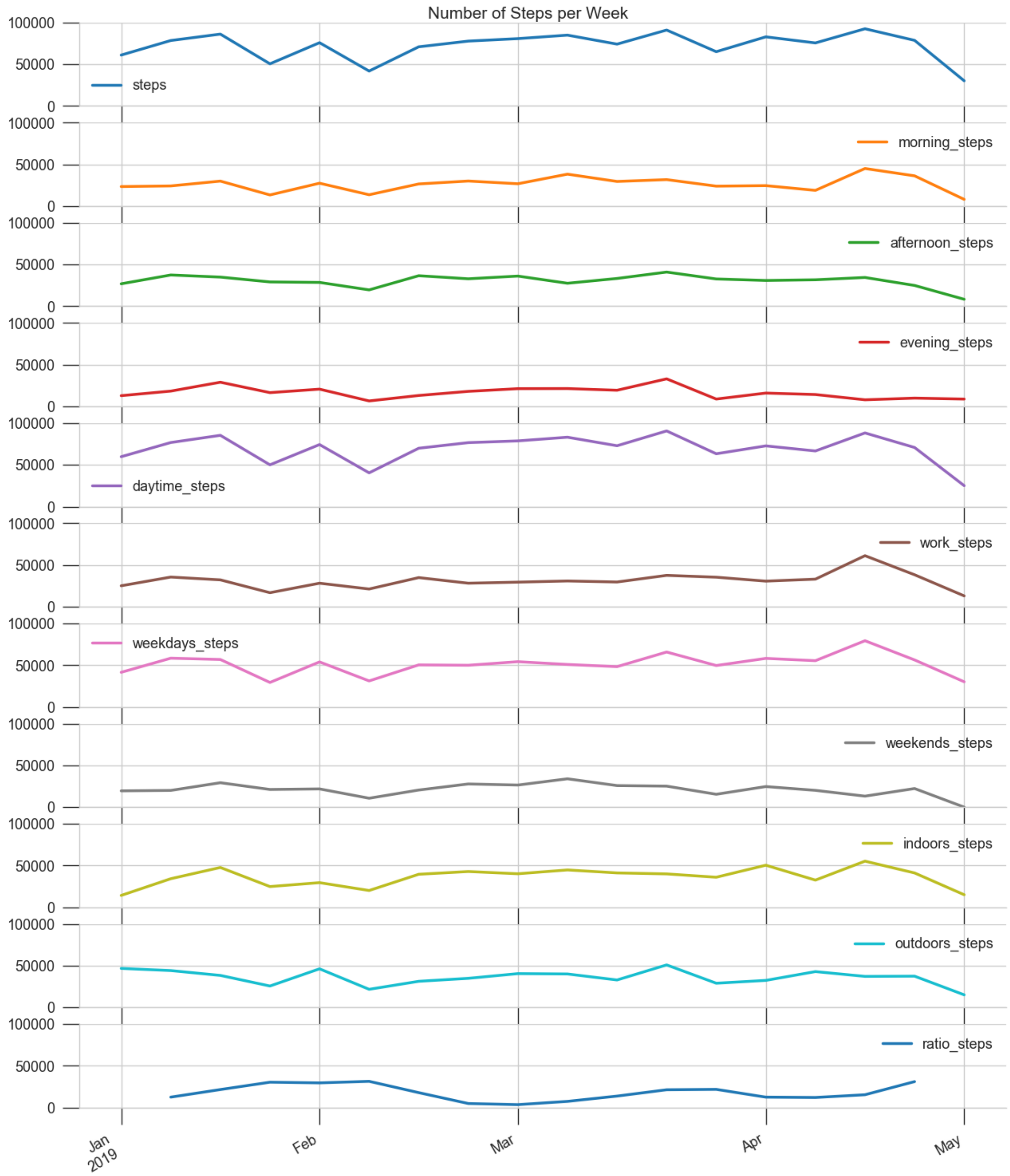

Figure 11: Weekly step count over a 4-month period. The counts are marginalized over different parts of the day (morning, afternoon, evening, daytime, worktime) and location (indoors, outdoors).

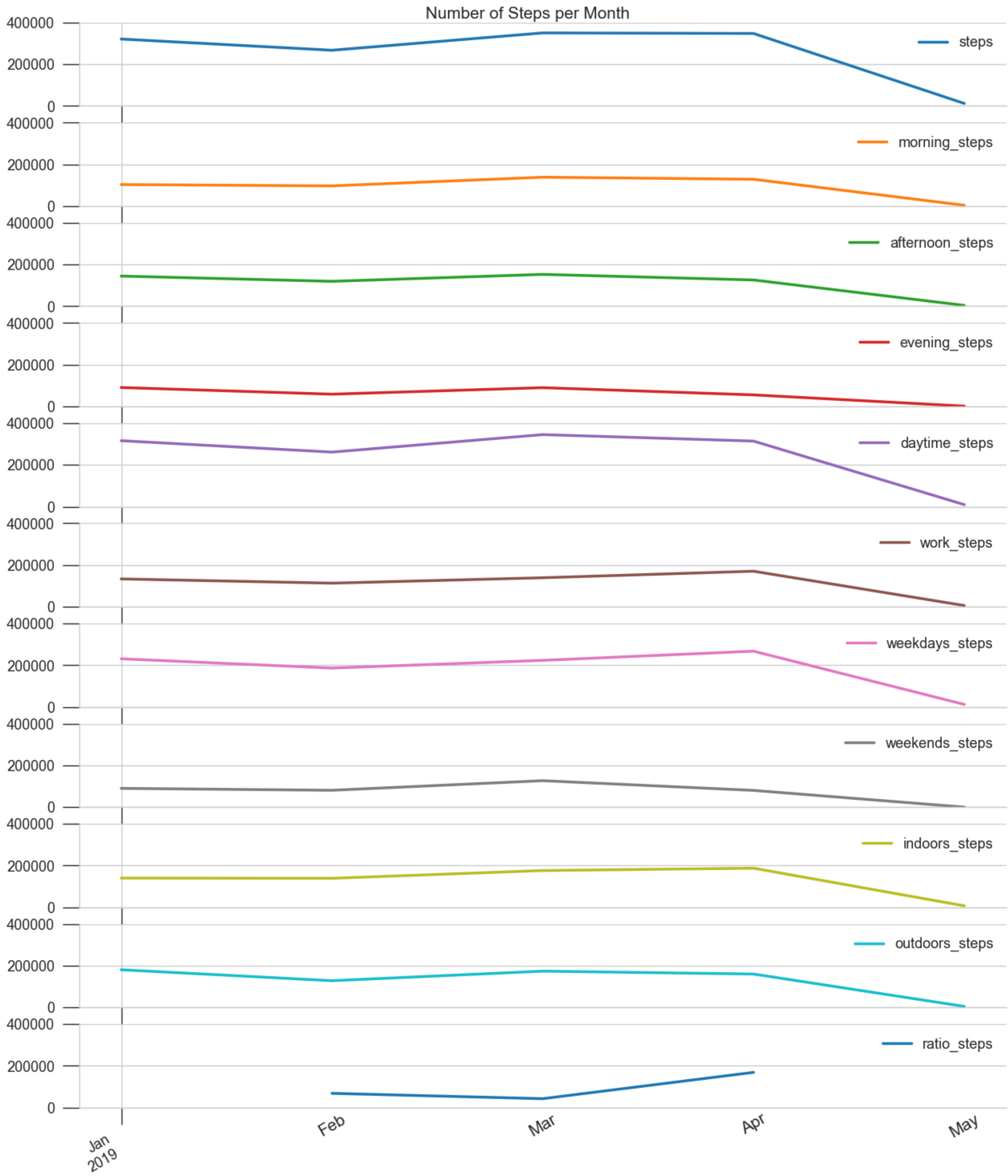

Figure 12: Monthly step count over a 4-month period. The counts are marginalized over different parts of the day (morning, afternoon, evening, daytime, worktime) and location (indoors, outdoors).

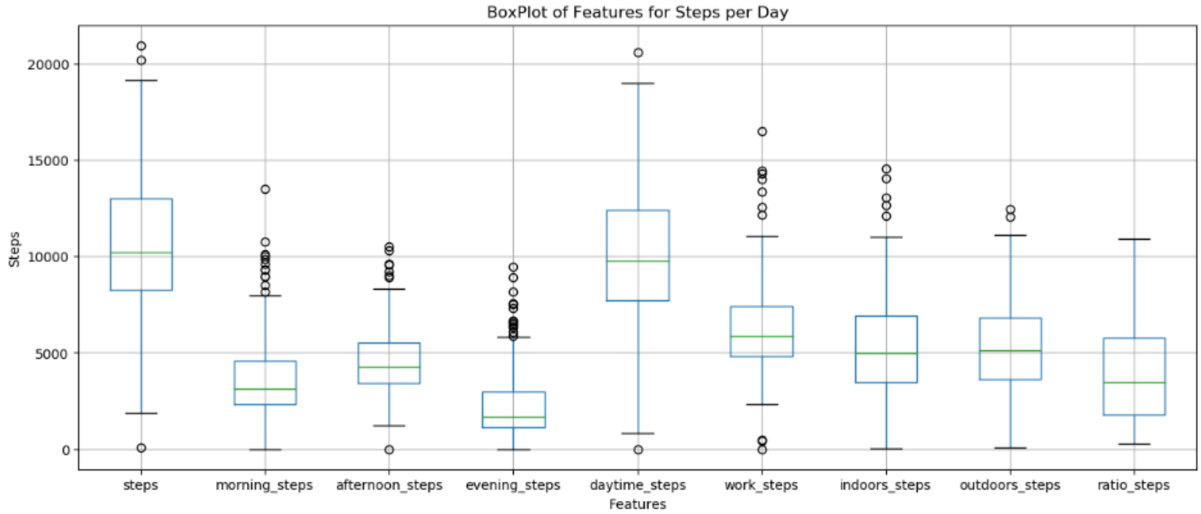

Figure 13: Distribution (box plot) of the daily step count over a 4-month period.

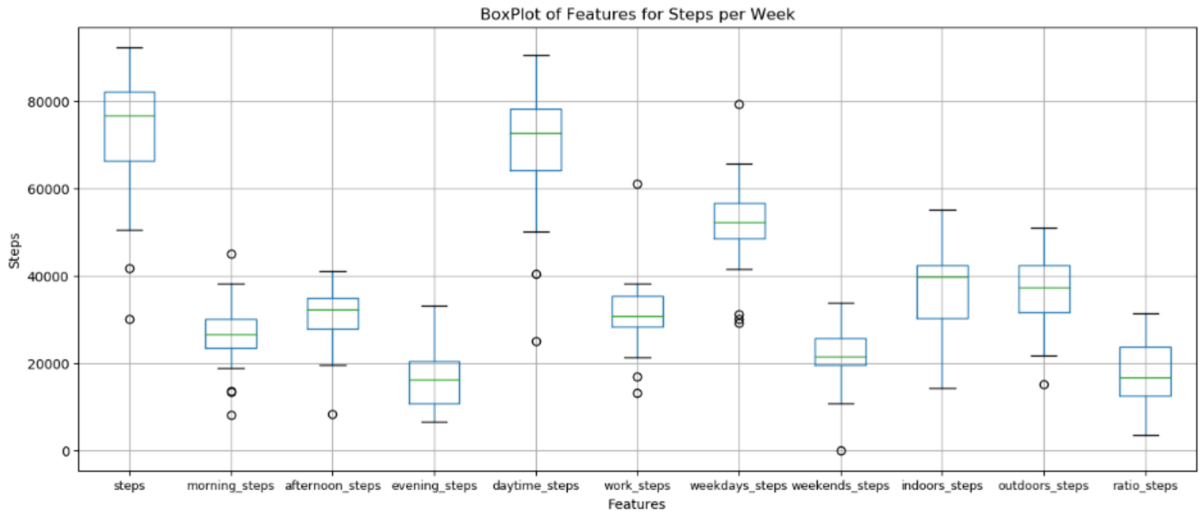

Figure 14: Distribution (box plot) of the weekly step count over a 4-month period.

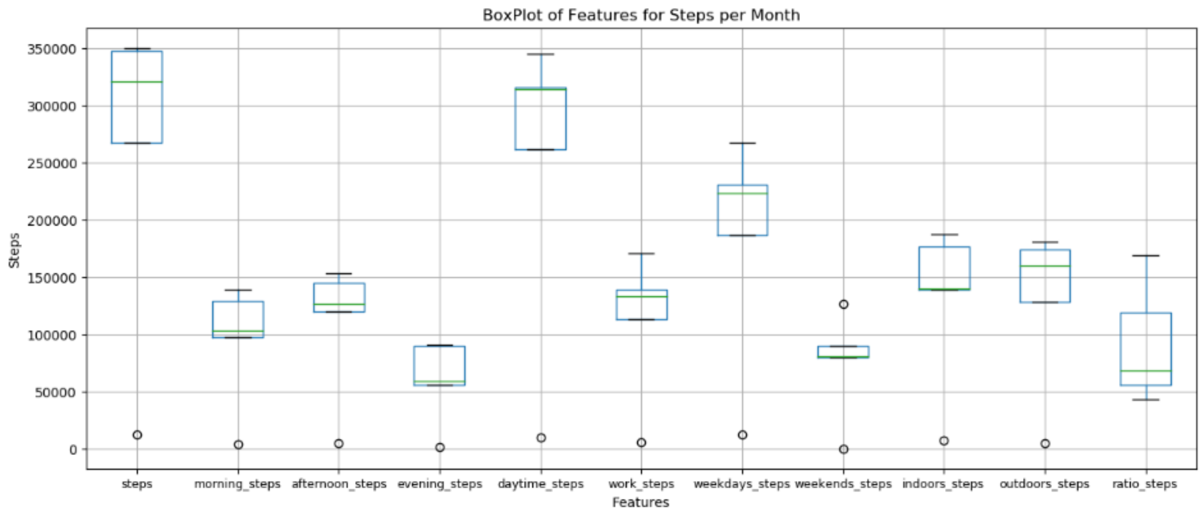

Figure 15: Distribution (box plot) of the monthly step count over a 4-month period.

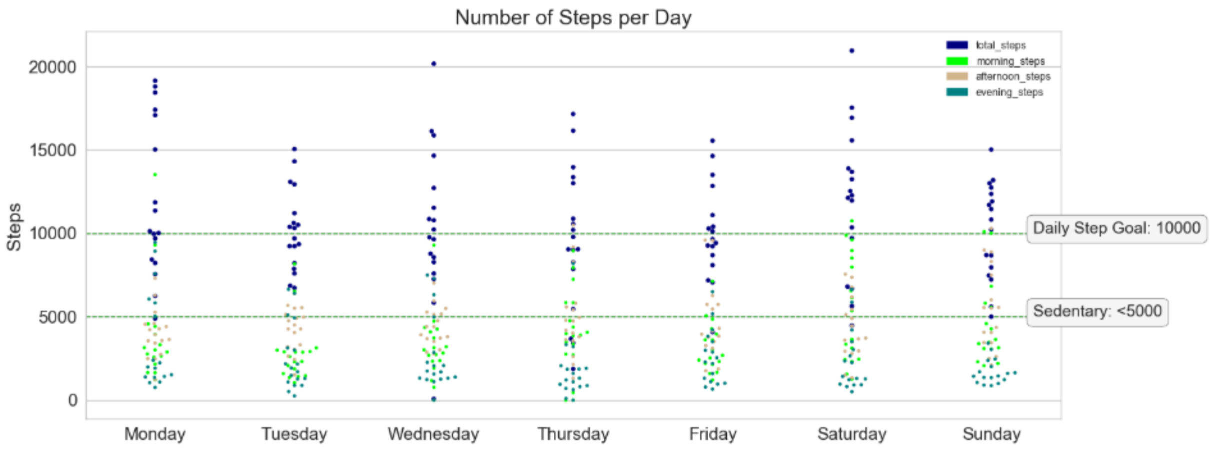

Figure 16: SwarmPlot of the steps per day comparing the total number of steps (navy colour), with steps performed during the morning (lime colour), during the afternoon (tan colour) and during the evening (teal colour).

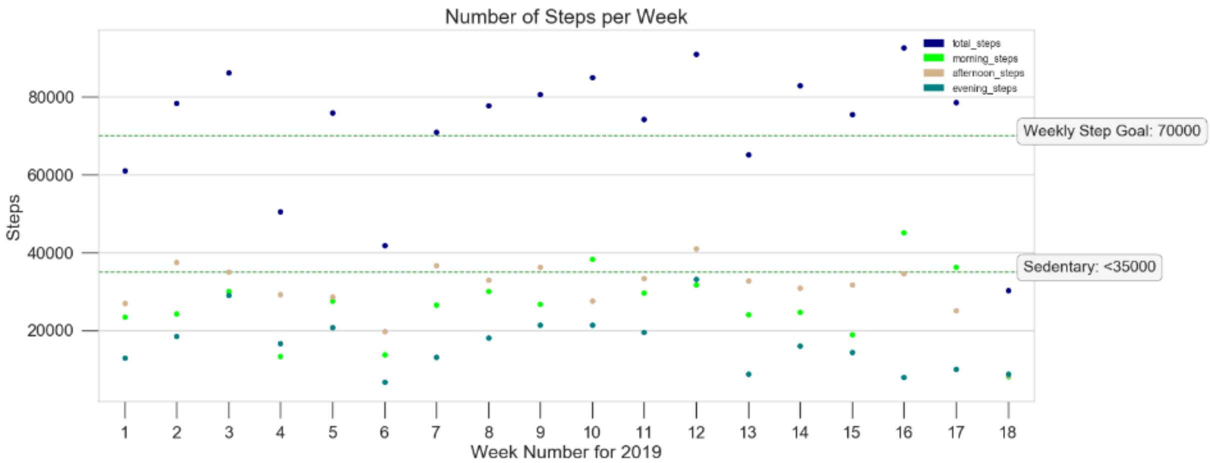

Figure 17: SwarmPlot of the steps per week comparing the total number of steps (navy colour), with steps performed during the morning (lime colour), during the afternoon (tan colour) and during the evening (teal colour).

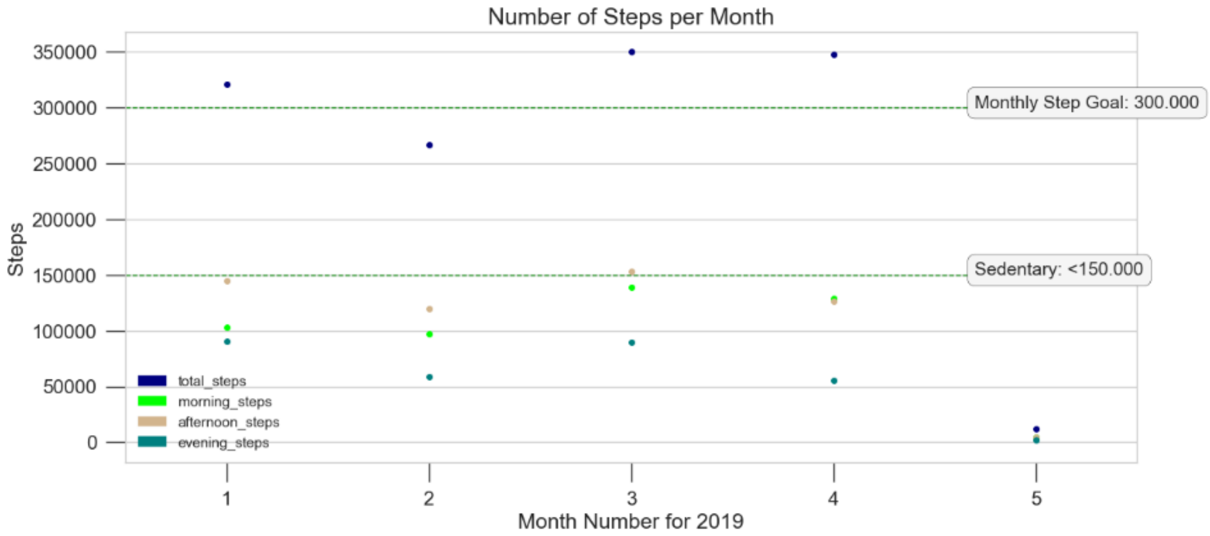

Figure 18: SwarmPlot of the steps per month comparing the total number of steps (navy colour), with steps performed during the morning (lime colour), during the afternoon (tan colour) and during the evening (teal colour).

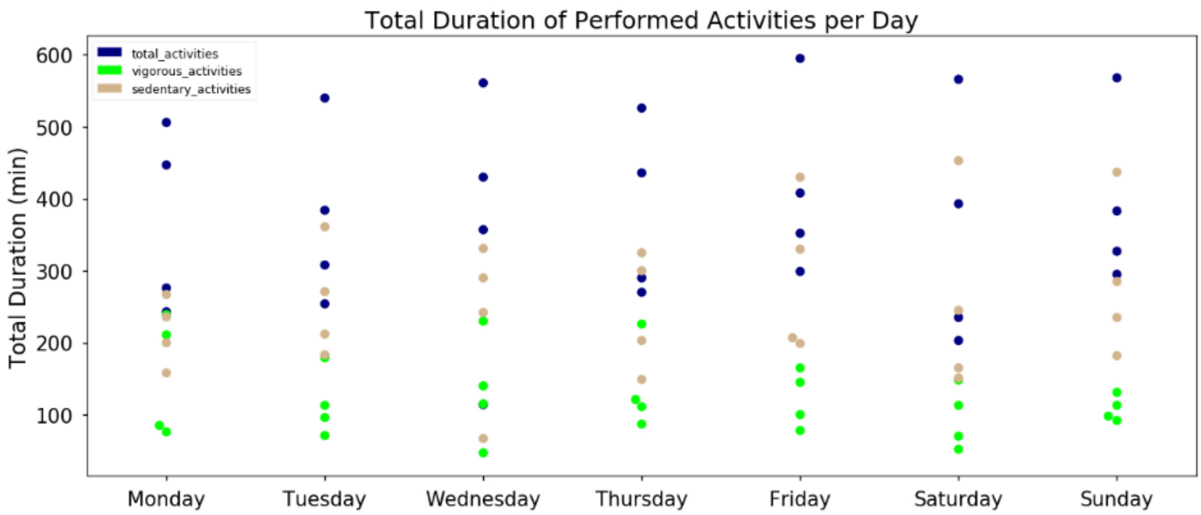

Figure 19: SwarmPlot of the total duration of performed activities per day, comparing all the performed activities (navy colour), with vigorous activities (lime colour) and sedentary activities (tan colour).

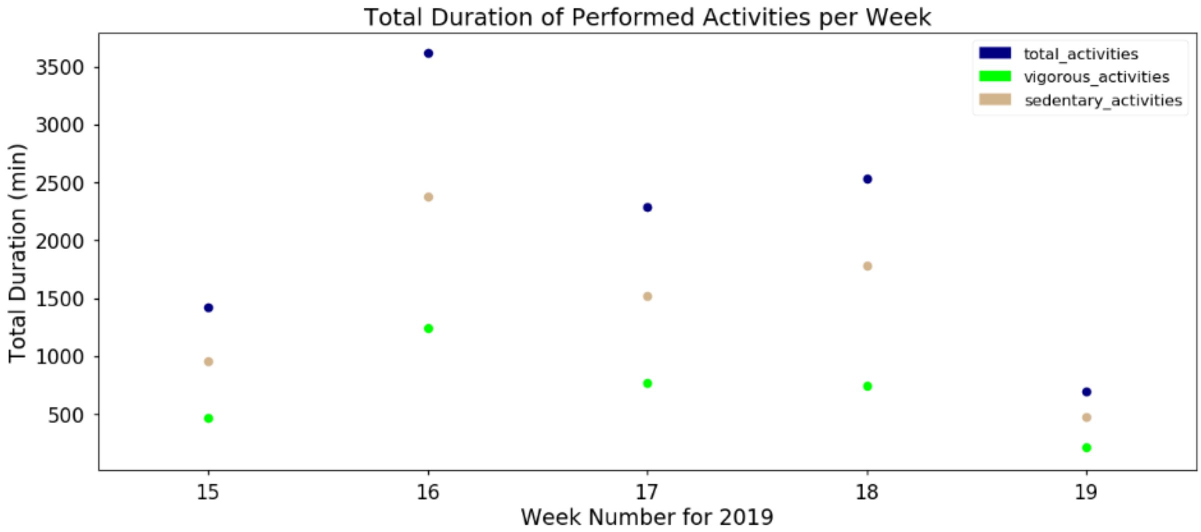

Figure 20: SwarmPlot of the total duration of performed activities per week, comparing all the performed activities (navy colour), with vigorous activities (lime colour) and sedentary activities (tan colour).

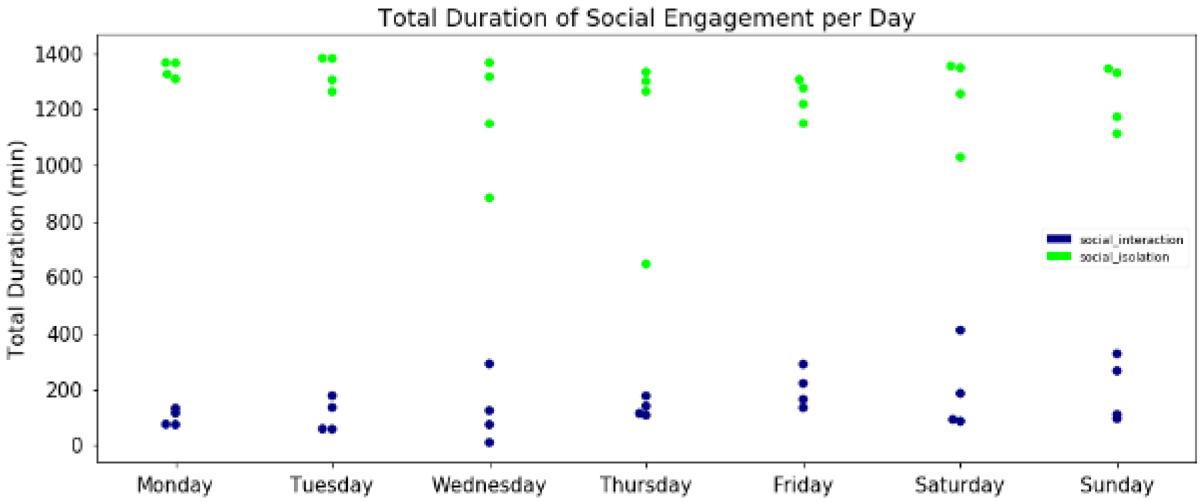

Figure 21: SwarmPlot of the social engagement per day, comparing the total duration of being socially interacted (navy colour), with being socially isolated (lime colour).

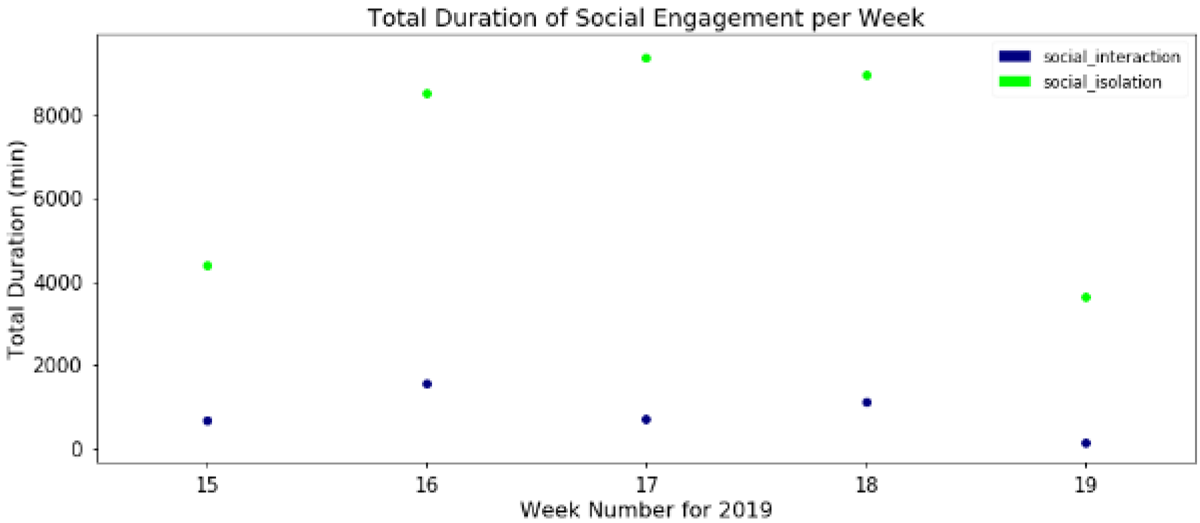

Figure 22: SwarmPlot of the social engagement per week, comparing the total duration of being socially interacted (navy colour), with being socially isolated (lime colour).

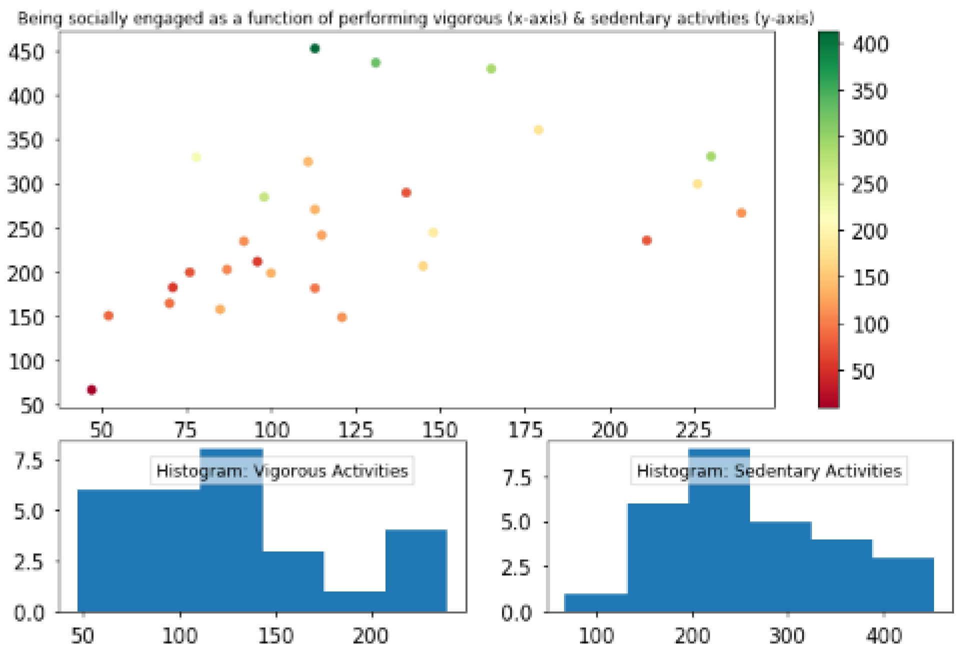

Figure 23: Scatter plot comparing the social interaction with the vigorous & sedentary activities.

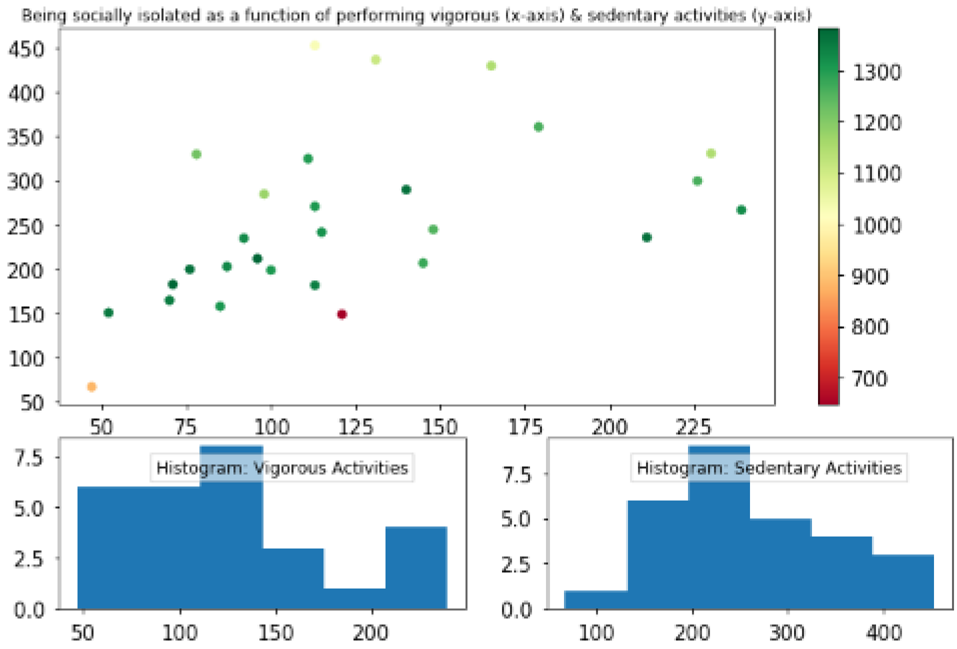

Figure 24: Scatter plot comparing the social isolation with the vigorous & sedentary activities.

6 Bibliography

Python 3.5.0. (2015). Retrieved from Python: https://www.python.org/downloads/release/python-350/

(2017). Retrieved from Datatracker: https://datatracker.ietf.org/doc/rfc8259/?include_text=1

(2017). Retrieved from ecma-international: https://www.ecma-international.org/publications/standards/Ecma-404.htm

(2018). Retrieved from MySQL: https://dev.mysql.com/doc/relnotes/mysql/5.7/en/

Activity Recognition API. (2018). Retrieved from Google: https://developers.google.com/location-context/activity-recognition/

Baltrusaitis, T. (2015). Retrieved from https://github.com/TadasBaltrusaitis/OpenFace

Camurri. (2004). Retrieved from http://www.infomus.org/eyesweb_eng.php

Google Brain. (2015). Retrieved from https://github.com/tensorflow

Keras. (2015). Retrieved from https://keras.io/

Microsoft. (2012). Retrieved from https://docs.microsoft.com/en-us/previous-versions/office/developer/speech-technologies/dd266409(v%3doffice.14)

Slim 3.9.0 released. (2017). Retrieved from Slim Framework: https://www.slimframework.com/2017/11/04/slim-3.9.0.html

Ubuntu 14.04.5 LTS Trusty Tahr. (2018). Retrieved from Ubuntu: http://releases.ubuntu.com/trusty/

YouTube - Demonstrator for 'On-Body' Sensors. (2018). Retrieved from YouTube: https://youtu.be/Mr0afhSKrVk