ML4QS: Assignment 3*

Sven van den Beukel, Erik van den Boogaard, and John (Can) Lokman **

Vrije Universiteit Amsterdam, Department of Computer Science, De Boelelaan 1105, 1081 HV Amsterdam, Netherlands

Date: 2017.07.02

Student Numbers: 2521966, 2582970, 2563162

Abstract. For our final assignment we want to predict sleep stage labels provided by an activity tracking device using classification. In order to do this, a Fitbit was used to gather data on heartrate (HR), activity level and sleep stages (as labeled by the Fitbit algorithm) over a period of 21 days (nights). This report outlines the process we went through during data preprocessing, analysis, and summarizes our results.

1 Introduction

Domain Sensors are increasingly available within most environments, allowing for more data-driven decision making. This does not limit itself to business- and academic environments. General population also has become interested in using electronic devices to measure data that subsequently helps them change their everyday lives. Researchers have dubbed this phenomenon as the *quantified self* [1]. We will be using a more formal definition of the quantified self, as described in [1], as shown in the box below.

> "The quantified self is any individual engaged in the self-tracking of any kind of biological, physical, behavioral, or environmental information. The self-tracking is driven by a certain goal of the individual with a desire to act upon the collected information."

This study focuses on tracking physical data to predict sleep stages by training classifiers. The dataset we used contains only one sensory feature (a users' heartrate in beats per minute [BPM]) and one aggregated feature (activity as measured by steps taken by user per minute). The data was gathered over a period of 14 days with a *Fitbit Charge 2* device. The Fitbit also provided the labels for four sleep stages, being awake, REM-sleep, light sleep and deep sleep; and thus, significantly contributed to our preprocessing and feature engineering endeavours.

* This project was carried out as part of the Machine Learning for the Quantified Self Course of Dr. Mark Hoogendoorn at the Vrije Universiteit Amsterdam

** The order of authors are alphabetical by last name; and each author made an equal amount of contribution to the project.

Pipeline and document's structure First, we explore and introduce the dataset and its characteristics; afterwards, we identify and deal with missing values and outliers in the Data Analysis section. Afterwards, we describe the process of deriving more features from this data, as well as the feature selection process. Section 4 explains which algorithms were used for tackling the classification problem. Finally, we present the results, and discuss them in the Results and Discussion sections.

2 Data Analysis

Data characteristics Our dataset is collected with a *Fitbit Charge 2* device which records the following activity each minute the user wears the device: average heart rate (Beats Per Minute, or BPM when abbreviated), activity level (steps per minute) and - *during bedtime* - sleep stages: wake, light, deep, rem, asleep, restless and awake, see Figure 1.

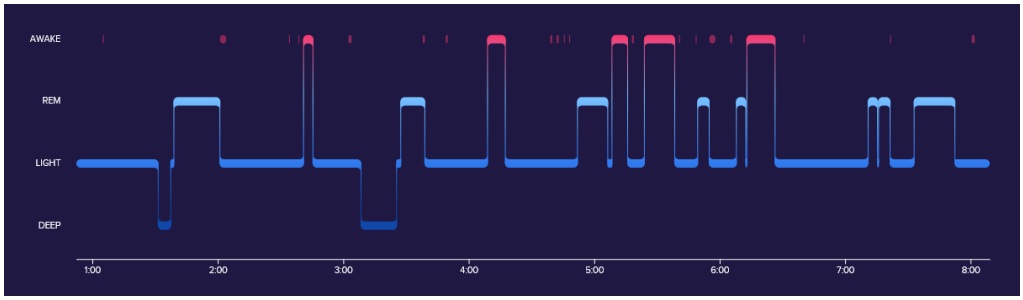

Fig. 1: A chart of sleep stages progressed through a night.

During the night one progresses through a series of sleep stages from light sleep to deep sleep, back to light sleep and into REM sleep. Then, the cycle repeats.

Striking is that there seems to be patterns in the sleep stages; e.g. it seems that *light sleep* (blue) is alternated by a short period of either *REM sleep* (light blue) or *deep sleep* (dark blue) and sometimes even with a shorter *wake* (red) period. During the period of 21 days we collected a total of 675 labeled periods (330 wake, 179 light sleep, 82 REM sleep, 41 deep sleep, 21 restless, 19 asleep and 3 awake). While each period was a multiple of 30 seconds; some were only 30 seconds in length, whereas others were 300 or even 3000. Boxplots (see Figure Boxplots) of HR and activity show that HR stays between 60 and 80 bpm during the night with only a few percent outliers. Activity (steps/minute) during these periods are (not surprisingly) mostly zero with only a few percent of (light) activity. The (short) periods *with* activity may indicate restlessness or being (a)wake.

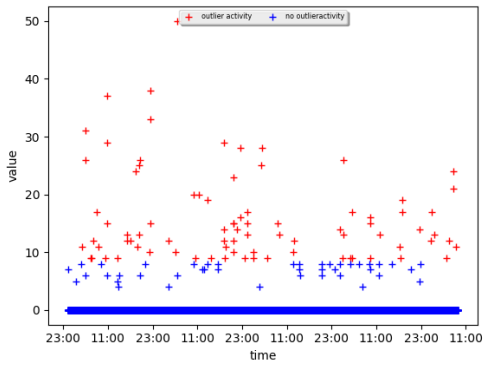

Fig. 2: Activity during sleep

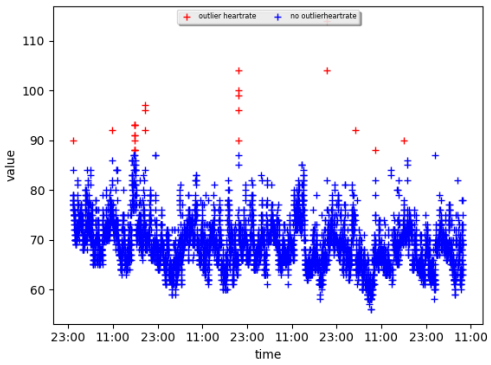

Fig. 3: HR during sleep stages

Outliers Based on Figure 2 and 3 there seems to be outliers in the measurements of activity and HR. But since we know the context - which is lying down in bed in some sleep stage - these so-called outliers might actually be good indicators (predictors) of being in (or transitioning to) a certain sleep stage! Therefore we assumed these values may not really outliers and should therefore not be discarded.

Preprocessing Because we are only interested in labeling *sleep* stages, we discarded all irrelevant data (HR and activity) collected during the *waking* periods of the day - typically between 07:00 and 24:00. These awake periods can easily be spotted by NaN-values (i.e., *Not a Number*) in the sleeping stage labels. For each night, the 10 instances with NaN values were kept in the dataset, so that the information required for aggregating temporal heartrate values for the first 10 instances of each night, remained intact. These values were added

Column	% Missing Values	Mean	SD	Min	Max
activity	0	0.26	2.13	0	50
heartrate	0	69.14	4.8	56	114
sleep_stage	0.02	1.59	0.90	0	3
sleep_stage_0	0	0.19	0.39	0	1
sleep_stage_1	0	0.11	0.31	0	1
sleep_stage_2	0	0.59	0.49	0	1
sleep_stage_3	0	0.09	0.29	0	1
sleep_stage_nan	0	0.02	0.14	0	1

Table 1: Characteristics of the data and its key features.

right after the data of each previous night. Since the instances with NaN values are not used in the final classification, the right temporal order is retained this way. For all sleep time points, we calculated aggregated values with a maximum window size of 10. A higher window size would have had resulted in aggregating over the wrong data (a window size of 11 means using the final data point of the previous night in the aggregation). Eventually, approximately 6300 labeled instances (minutes) remained that primarily consisted of blocks of sleep level labels, and of the additional data described in-between these blocks.

Imputation Because 1 percent of the HR measurements were missing (possibly due to difficulties measuring HR in a lying down position) we imputed the missing HR data using linear interpolation between two measurements with a gap of 2 minutes. We chose linear interpolation due to measurements generally being stable and not varying significantly within a time span of 2 minutes. This method indeed filled in 98 one-instance 'blank spots' in the data. For blocks of missing data (e.g., caused by user not wearing the device for a few hours) no imputation was applied, because the reliability of such imputed values would be low, and thus extra noise would be added into the dataset.

Granularity We used a granularity of 1 minute for our temporal dataset because most (Fitbit) data was only available at a 1 minute granularity. Exception was the reported Fitbit data about sleep stages which was at a 30 second granularity. To be able to predict the Fitbit sleep stage labels using only data on HR and activity (1 minute granularity) we decided to set all granularity to 1 minute for convenience. Fitbit uses only 4 distinct sleep stages: *wake*, *light sleep*, *deep sleep* and *REM sleep*, whose measurements are highly likely based on HR and accelerometer (activity) sensor measurements.

3 Feature Engineering & Selection

Adding features Firstly, we created additional features to choose from by performing a Principle Component Analysis (PCA). Because we only had 2 features (HR and activity) only 2 Principle Components could be found, which

explained about 80 percent of the variance. Resulting from this step, 2 features were added for each instance.

Secondly, we created temporal features. To do so, we calculated the mean, standard deviation, median, and slope of the heartrate with a window size of 5 and 10 minutes, in order to create a large set of features. It is our belief that this increase in dimensionality improves the chance of finding a good performing subset of features.

Additionally, we applied Batal algorithm for categorical abstraction and identified 9 more features with a threshold of 0.03. However, since adding them resulted in a heavily overfitted model (accuracy of 1 on the decision tree algorithm), these were emitted. In total, 17 temporal features were added, of which 8 were eventually used for classification. Additionally, the basic frequency abstraction was applied to the heartrate data as provided through the course materials.

Finally, we performed *k-means clustering* to find any patterns in the data space. To select the best value for k, *silhouette* values were calculated. The algorithm shows an optimum with a k of 4 clusters (equal to the amount of sleep stages) with a silhouette coefficient value of 0.55 (see Figure **??**). The k-means clustering approach had the most optimal performance; better than k-medoid or hierarchical clustering. Each instance was extended with one feature describing the cluster the instance was assigned to.

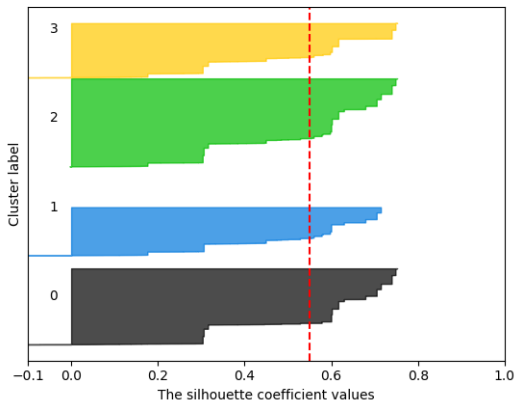

Fig. 4: The silhouette plot for various clusters.

For feature selection we chose to use the one provided for Chapter 7 of the course, which uses a decision tree classifier for testing performance given a set of features. The 7 features that were found to be most informative can be found in Table 2. It appears that besides PCA 2, only temporal aggregations of heartrate data add extra predictive power for the decision tree algorithm, in which temporal data over a larger window size seems to add most valuable data. Whether this is

also the case for other algorithms, we need to find out by feeding the selected features as a feature set to the same classifiers as the complete feature set. To make the comparison complete, all intermediate datasets should be fed to the classification algorithms, a step after which accuracy can be compared. There was no time left for us to apply regularization to lower the chance of overfitting.

Rank	Feature
1	pca_2
2	heartrate_temp$_mean_ws$_10
3	heartrate_temp_std_ws_10
4	heartrate_temp_slope_ws_10
5	heartrate_temp_mean_ws_5
6	heartrate_temp_slope_ws_5
7	heartrate_temp_std_ws_5

Table 2: Most informative features

4 Algorithms

We initially selected the feedforward neural networks, support vector machines, random forests, decision trees, naive Bayes and k-nearest neighbour for classification. However, due to time restrictions resulting from difficulties in obtaining and preprocessing the fitbit data, we could not run these within the available time. Therefore, we chose to drop the three non-deterministic classifiers: The neural network, random forest and the SVM. Due to the high computation costs associated with these three classification algorithms, it has not been possible to complete their runs before the project deadline. However, we do believe that, if their runs were finished, these would result in higher accuracy than the currently used deterministic classifiers.

5 Results

In order to increase generalizability of the obtained results in the performed experiments, the training set and test set were created through a cut-off at 70% of the data. The remainder 30% was used as test data. Although normally stratified K-fold cross validation would have been our preferred choice, the use of aggregated temporal data would then cause overfitting, by allowing the training set to learn on aggregations of test data. The final results can be found in Table 3. Adding new features did not help improve the accuracy of the decision tree or the naive Bayes classifiers significantly. However, the K-nearest neighbor performed better when using more features.

Feature set	KNN Train	KNN Test	DT Train	DT Test	NB Train	NB Test
Initial set	0.483 *(0.465-0.501)*	0.477 *(0.450-0.505)*	0.608 *(0.590-0.626)*	0.607 *(0.580-0.635)*	0.607 *(0.590-0.625)*	0.606 *(0.579-0.634)*
Initial+ PCA	0.483 *(0.464-0.501)*	0.477 *(0.449-0.505)*	0.608 *(0.591-0.626)*	0.609 *(0.582-0.636)*	0.599 *(0.581-0.617)*	0.606 *(0.579-0.634)*
Initial+ PCA (+temporal)	0.674 *(0.656-0.691)*	0.583 *(0.555-0.611)*	0.603 *(0.585-0.621)*	0.613 *(0.585-0.640)*	0.441 *(0.423-0.460)*	0.559 *(0.531-0.587)*
Initial+ PCA+ temporal (+cluster)	0.673 *(0.655-0.690)*	0.585 *(0.557-0.612)*	0.603 *(0.585-0.621)*	0.613 *(0.585-0.640)*	0.447 *(0.428-0.465)*	0.560 *(0.533-0.588)*
Selected features	0.710 *(0.693-0.726)*	0.556 *(0.528-0.583)*	0.657 *(0.639-0.674)*	0.583 *(0.555-0.611)*	0.483 *(0.464-0.501)*	0.563 *(0.536-0.591)*

Table 3: Results for K-NN, Decision Tree and Naive Bayes

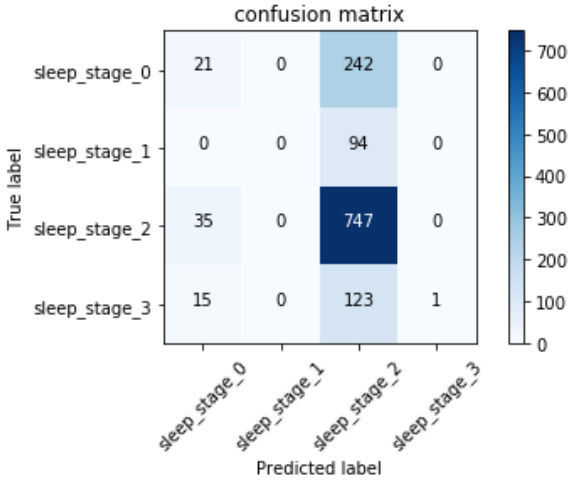

Fig. 5: Confusion matrix

See Figure 5 for the confusion matrix which shows the accuracy of the predicted labels. The only high score comes from *sleep stage 2* (light sleep) but still has an accuracy of only 62 percent.

6 Discussion

A classification accuracy of only +- 61 percent as a best overall classification result, shows that classifying sleep stages using only aggregated data about heartrate (bpm) and activity (steps per minute) poorly predicts the provided Fitbit labels. This is probably due to Fitbit using fine-grained and raw sensor data (accelerometer, HR sensor) to annotate sleep stages. However, the fact that the small amount of two features allow a decision tree to achieve this accuracy, seems to confirm our suspicion that these are informative features for determining a user's sleep phase. From the confusion matrix it becomes apparent that our classifiers could not train properly due to light sleep being heavily over represented

in the dataset (see Table 4, causing the classifiers to simply predict light sleep in 62% of the classifications, while predicting deep sleep (stage 1) and awake (stage 3) respectively 0 and 1 times.

Sleep phase	Code	Occurrences
REM-sleep	0	1190
Deep	1	693
Light	2	3707
Wake	3	566

Table 4: Occurrences per sleep phase

The results of the naive Bayes classifier were initially very surprising. While usually the training error is equal to -or higher than- the test error, this time it is the other way around. It is unclear why, but the naive Bayes classifier performs (up to 12%) better on the test set than on the training set. The most plausible explanation for this, is the possibility that simple variety in the data may have accidentally benefited the training data, for example by overfitting on a label that was disproportionally represented in the test set.

For future work, we suggest using a smartwatch to gather more fine-grained data (e.g., with an accelerometer and gyroscope), as Fitbit was only able to provide aggregated data (i.e., blocks of sleep stages or activity levels inferred from movement patterns, instead of actual gyroscope and accelerometer values). Although Fitbit uses some of the above-mentioned sensors itself, unrestricted access into sensor measurements could certainly prove to be useful for testing whether different algorithms than ones used by Fitbit could perform better. Therefore, if this data is gathered, one should consider actually competing with fitbit by classifying on a dataset in which sleep phases were labeled using a different device, such as EEG measurements or real-time MRI scans. Then, the applied techniques may yield the potential to outperform the Fitbit software in sleep phase prediction, and possibly be transferred to future devices for better results. Furthermore, a more evenly distributed dataset must be compiled, containing a roughly equal amount of occurrences of all sleep phases.

References

1. Hoogendoorn, M., Funk, B.: Machine learning for the Quantified Self: On the art of learning from sensory data. Springer (2017)