# Prediction of mood on the basis of smartphone data

An De Rijdt[1]

[1] VU Amsterdam
a.j.de.rijdt@student.vu.nl

**Keywords:** Data mining, Mood, Temporal Data

## 1 Introduction

More and more apps are used to support patients suffering from depression. In this paper we aim to predict their mood on the basis of smartphone data, both from automatic tracking as from manually entered data. This would be valuable for instance for depression patients, enabling quick intervention when a very low mood is expected for the next day.

We are using a dataset consisting of data for 27 individuals for which 19 variables are measured over time.

After transforming the dataset into a regular multidimensional set in order to apply non-time series regression models, we investigate different ways to aggregate in terms of function and aggregation period and compare to the simplest baseline model. Although performing better than the baseline model, there is model found that significantly outperforms the others.

The analysis was performed using python in a jupyter notebook.[1]

## 2 Data exploration and preprocessing

We first examine the dataset as is. The participants mood is expressed in two ways in our dataset, the first is a unidimensional entry of the participants indicating their mood in a scale between 1-10, the other is following the circumplex mood model as described in [3], expressing mood as a linear combination of the valence and arousal dimension. We will focus on the first one as target variable and take the circumplex variables as features.

We note there are two issues:

- Duplicate entries (e.g. mood is logged twice for the same timestamp with a different value): We solve this by taking the latest as it are manually entered values so we assume it is a correction by the user.

- Missing values for valence and arousal. We forward fill those values.

The dataset as it is available to us is not suitable for regular prediction algorithms models. Instead of using timeseries models like ARIMA, we want to capture the time aspect by aggregating the variables as described in chapter 4 of [2].

We define a method to preprocess the data, this entails:

- pivoting the variables so that we get a feature per variable
- aggregating per day to abstract away the time, knowing that this also loses information. However from [4] we know that not much success was reached by using the raw data.
- the aggregation function is depending on the type of variable: mean (excluding NaN values) for mood, activity, arousal and valence, and sum for the duration related variables as screen and app time. For the latter count would also make sense (times we look on the screen), but we start with the sum.
- shifting all features by one day so that we can define the new target variable as the average mood over the day
- Making new features of the temporal variables by windowing over different periods and aggregating over these windows (e.g. average of the last x days for mood)

The latter will be explained further in part 4 as this is part of the model building and evaluation.

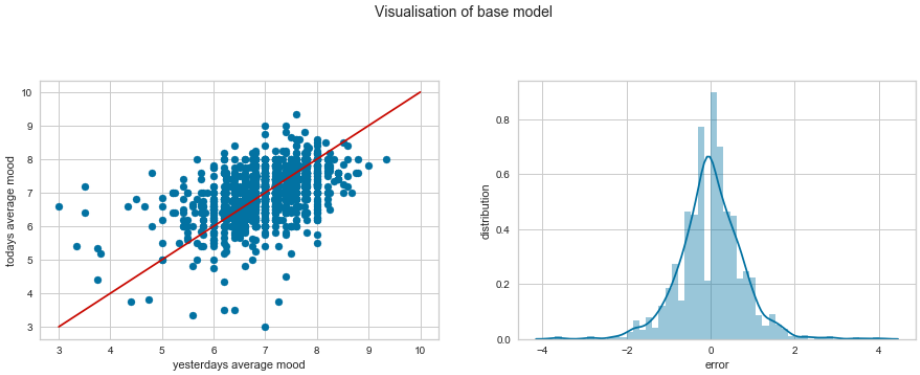
## 3 Base model

To be able to compare more advanced regression models, we first build a basic model which predicts average mood by taking the average entered mood over the past day.

As evaluation metric we calculate the percentage of predictions which is within a treshhold error margin rather than taking a standard metric as rmse. The reason for this is that we mainly want to limit extreme mistakes. The same approach is taken in [4]. However a very big mistake, especially downwards (for instance predicting mood will be the same and then it goes down a lot) is way worse. Setting the treshold of error to 0.6 already gives us 66% accuracy. With a treshhold of 0.5 the accuracy is 0.53.

In figure 1 the base model is visualized together with the distribution of errors.

**Fig. 1.** Distribution of errors in the base model




## 4 Feature engineering

Although we have summarized the different variables per day, we can capture more of the time aspect by using time windows to aggregate over. We know from [2] that it is hard to predict the correct window sizes. Inspired by the approach in [5] we try out different aggregation functions and window sizes to see how they affect the predictive power of our models. We notice the following

- Slope instead of mean gives very bad results
- We varied the window sizes, longer window sizes gave better results. We ended up with 5 day windows for all variables.
- Both for daily aggregation (in the previous step) as for window aggregation we changed between sum, count and mean for the duration related variables. Daily sum and mean over the window gave the best results.

When looking at the correlation between the features and the target variable, it is not surprising that mood and valence have high correlation. Arousal surprisingly has rather low correlation which we wouldn't expect given the circumplex model [3]. Maybe this is caused because entering one self's mood stays a subjective matter.

As we are using 5 day windows, we cannot create features for the first 5 days of mood logging for each patient so those were removed. Some activity scores still are missing after that, which we impute by the mean per participant.

Not surprisingly a lot of features have exponential distribution, as they are related to duration. Hence we will apply normalization during the model building process.

When looking at correlation with the target, we see that the mood for the past 5 days (when using a 5 day window) and valence have the highest absolute correlation if we look patient independent. However, looking at the correlation matrix grouped by pa-

tient id, correlation coëfficients wildly vary. This indicateds that a model per patient might be a better idea, although probably more difficult to maintain.

## 5 Model setup, building and evaluation

While we know from [4] that personalized mood models can attain a high accuracy, we decide to build one model for all patients as this requires less data per patient and is probably less maintenance intensive.

To evaluate models RMSE was used but also percentage which was less than 0.5 off the correct value. As explained above this seems to us more relevant and is in line with earlier research[4]. For this built a custom scoring functions to apply whithin our sklearn pipeline. To cater for the exponential distribution of a lot of variables we use a scaler and normalizer in the preprocessing step of the pipeline.

Different regression models were applied to the dataset, with a 50/50 train test split we get very different results every time we run the algorithm. We see that Decision tree and Random Forest are very prone to overfitting.

**Table 1.** Train and test performances

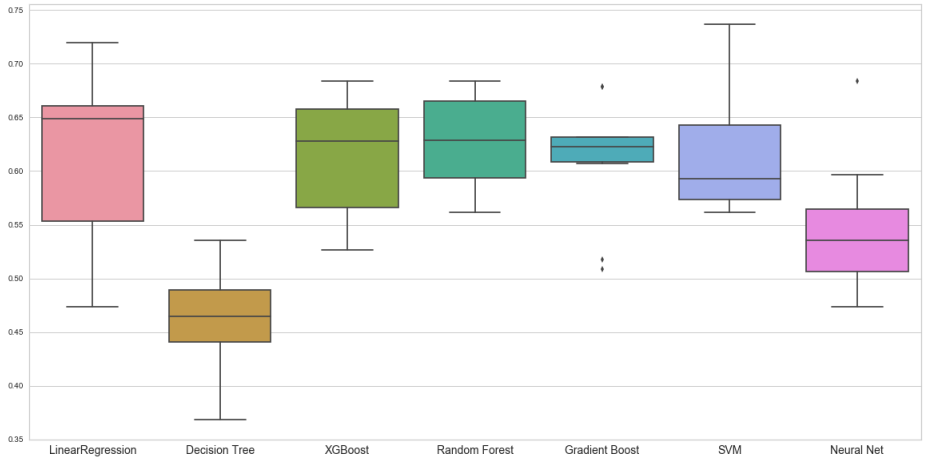
| | test error | train error | test mse | train mse |
|---|---|---|---|---|
| **Decision Tree** | 0.492063 | 1.000.000 | 0.891829 | 3,34E-26 |
| **Gradient Boost** | 0.627866 | 0.800353 | 0.449213 | 1,51E+05 |
| **LinearRegression** | 0.640212 | 0.632509 | 0.398447 | 4,19E+05 |
| **Neural Net** | 0.592593 | 0.614841 | 0.436560 | 4,21E+05 |
| **Random Forest** | 0.634921 | 0.952297 | 0.426079 | 5,96E+04 |
| **SVM** | 0.634921 | 0.628975 | 0.391917 | 4,33E+05 |
| **XGBoost** | 0.631393 | 0.795053 | 0.493563 | 1,56E+05 |

We notice that when we repeat this a few times, we get quite different results. So to assess the performance of our models, we perform k-fold crossvalidation (with fixed seed).

We notice that for our custom scorer function, while most result sets have quite high variance, XGBoost has quite consistent results.

Linear regression had the highest mean accuracy, it also has the highest variance in its results.

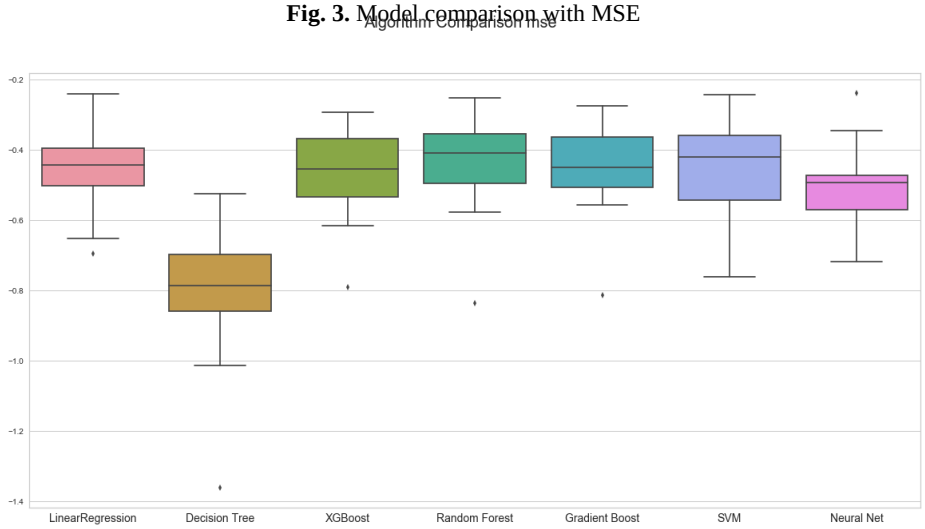
**Fig. 2.**
Algorithm Comparison




Since the results don't seem normally distributed, we use the Wilcoxon test to statistically compare model performances. With a p-value of 0.55, we cannot reject the null hypothesis that there is a difference in performance between the Linear Regression and the next in line, Random Forest. However, the test also shows that the simple decision tree does perform significantly worse than all other models.

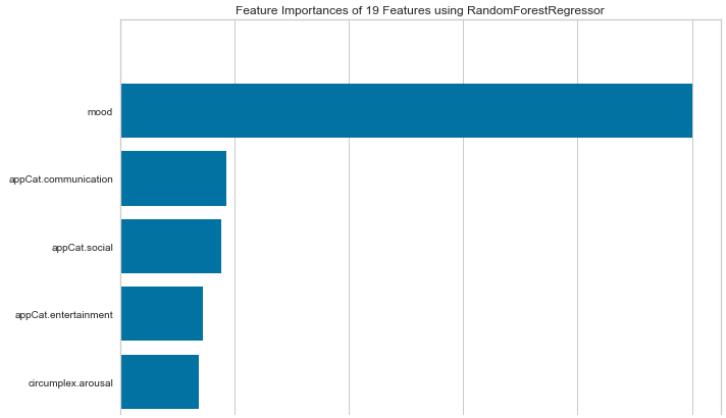
When using MSE as validation criteria, the variances are smaller, but and the Wilcoxon test shows again that the decision tree also in this case performs significantly worse than all other models.

**Fig. 3.** Model comparison with MSE




Looking at feature importances e.g. for Random Forest, not surprisingly mood is the most important:

**Fig. 4.** Feature importance



## 6 Conclusions and further recommendations

We demonstrated that mood over previous 5 days is a strong predictor for next days mood, together with the usage of communication and social apps. More advanced models outperformed the baseline model of just taking previous day's mood.

While users self-entered mood is a already a fairly good predictor for mood of next days, this still requires an effort from the patient which can be seen as a burden. An approach as described in [4] where mood is successfully inferred from automatically gathered data seems less intrusive and promising.

As more data comes available increasingly over time (at least more non-manually entered data), it makes sense to build models which take the whole history into account and not a fixed window size per variable. Also, more advanced aggregation can be taken into account, as there are:

- Looking at time windows to create features, for instance, screen time before bed.
- Using multiple aggregations per variable instead of one as we did.

As next step, we would propose to predict the slope (negative or positive) of mood or sudden changes. One down might be worse than one up, especially from a psychologist perspective. This is not something covered in this analysis but a good next step.

### References


1. An De Rijdt: https://github.com/klimantje/DataMining/blob/master/PredictionOfMood.ipynb
2. Hoogendoorn, M. and Funk, B., Machine Learning for the Quantified Self - On the Art of Learning from Sensory Data, Springer, 2017
3. Russell JA. A circumplex model of affect. J Pers Soc Psychol 1980 Dec;39(6):1161-1178
4. LiKamWa R, Liu Y, Lane N, Zhong L. MoodScope: building a mood sensor from smartphone usage patterns. In: Proceedingsof the 11th Annual International Conference on Mobile Systems, Applications, and Services. 2013 Presented at: MobiSys'13: 11th Annual International Conference on Mobile Systems, Applications, and Services; Jun 25-28, 2013; Taipei, Taiwanp. 389-402
5. Breda, W.v., Hoogendoorn, M., Eiben, A., Andersson, G., Riper, H., Ruwaard, J., Vernmark,K.: A feature representation learning method for temporal datasets. In: IEEE SSCI 2016.IEEE (2016)