Data Mining Techniques - Advanced Assignment

Andrei Apostol[1,2], Orestis Kompougias[1,3], and Traian Vidraşcu[1,4]

[1] University of Amsterdam, Vrije Universiteit Amsterdam
[2] VU ID: 2656363
[3] VU ID: 2656408
[4] VU ID: 2655587

1 Introduction

In this report, we analyze a mental health dataset which contains records of smartphone activity from the users. Our purpose is to create a model that can predict a user's mood based on the activities and mood of the previous days. To this end, we have implemented two approaches: a model inspired from time-series analysis as well as a method that follows a more classical machine learning approach. A baseline model has been set up for benchmarking purposes.

More details regarding the steps we took in order to bring the dataset to a more suitable form as well as details regarding the implementation of our models and testing methods are given in the following sections.

2 Dataset

Our dataset consists of activity records from 27 patients obtained via a psychology application. These records each contain four columns: the ID of each patient, the timestamp at which the information has been recorded, the type of variable it represents and its specific value at that time. Note that some of the records are automatically generated by the user application, while some are explicitly filled in by the user, as the application periodically queries the user. This knowledge will be of use when we pre-process the dataset.

The specific type of variables that we have in our dataset are given in Table 1.

Table 1: Types of variables in our dataset.

Variable	Description	Input Method
mood	The mood scored by the user on a scale of 1-10	User input
circumplex.arousal	The arousal scored by the user on a scale between -2 and 2	User input
circumplex.valence	The valence scored by the user on a scale between -2 and 2	User input
activity	Activity score of the user (number between 0 and 1)	Recorded
screen	Duration of screen activity (time)	Recorded
call	Call made (indicated by a 1)	Recorded
sms	SMS sent (indicated by a 1)	Recorded
appCat.builtin	Duration of usage of built-in apps (time)	Recorded
appCat.communication	Duration of usage of communication apps (time)	Recorded
appCat.entertainment	Duration of usage of entertainment apps (time)	Recorded
appCat.finance	Duration of usage of finance apps (time)	Recorded
appCat.game	Duration of usage of game apps (time)	Recorded
appCat.office	Duration of usage of office apps (time)	Recorded
appCat.other	Duration of usage of other apps (time)	Recorded
appCat.social	Duration of usage of social apps (time)	Recorded
appCat.travel	Duration of usage of travel apps (time)	Recorded
appCat.unknown	Duration of usage of unknown apps (time)	Recorded
appCat.utilities	Duration of usage of utilities apps (time)	Recorded
appCat.weather	Duration of usage of weather apps (time)	Recorded

3 Initial pre-processing of the Dataset

In order to create a model that can predict subsequent mood values, we have taken some steps to transform this dataset. We define our target as the patient's mood and the features as being the rest of the variables from the dataset. We are interested in transforming this dataset in such a way that will enable our model to learn the temporal dependency between the features and the target.

There are, essentially, two approaches that we can take when building our model. One is when we aggregate the data for each patient individually, therefore creating as many datasets as there are patients. The other one involves building a single dataset where the information recorded for all the patients are summarized in some way (e.g. by taking mean or median values, etc.). We have chosen to do both of them.

3.1 Creating Meaningful Features

The first step in creating a meaningful representation is to turn each type of variable into a column. Essentially, we want each row in our dataset to represent a particular day, and the columns representing the variables recorded in that day with their specific value. In doing so, we can represent the patient's data for a specific day as a vector x and the mood recorded for that day as a target variable y. If a patient has multiple recordings in the same day (for instance, mood) then we use the mean of these values.

When creating the aggregated dataset, we simply average over the values of each patient that has recorded data in that particular day.

In Fig. 2 you can see a bar plot of the number of days that each patient has been using the app to record values. This distribution is relatively balanced and, as such, our models should be able to generalize equally well for each patient individually.

The resulting histogram of the target variable can be observed in Fig. 3. It resembles a sharply peaked normal distribution. The mean is 7.0005 with a standard deviation of 0.6460.

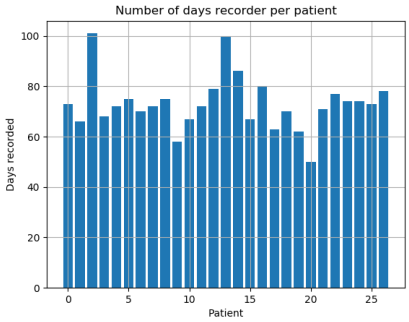

Fig. 2: Number of days recorded per patient.

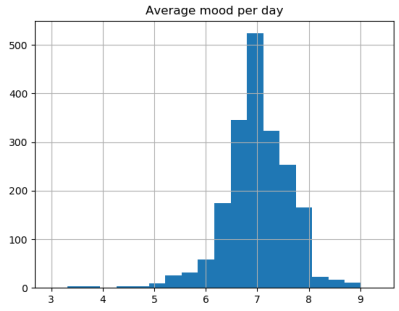

Fig. 3: Average mood per day.

3.2 Handling Missing Values

Some values in our original dataset are missing. For instance, for a particular day, a certain patient might not have sent any SMS messages. As such, there will be no record of that variable. However, having domain information about what our data represents helps us to infer what these missing values are.

We have replaced the missing values of *call* and *sms* with zero, since we know the reason why they are missing is that the event that they measure has not occurred. The variables measuring time spent using certain type of apps or general screen time have also been added with values of zero if they were missing.

For arousal and valence we replace the missing values with the mean. This mean is calculated per person for the datasets that are patient-specific, or globally for the aggregated dataset.

For the missing variables of mood, we have replaced the missing values with the one from the previous day. In case the mood value on the first day is missing, we replace that with the mean, like we did for arousal and valence.

3.3 Feature Correlation

To determine how useful our features are, we study the correlation between them and the target variable mood. Features with a low correlation coefficient are likely to not offer sufficient information for prediction. Two ways of dealing with this are applying transformations to the features (exponential, logarithmic etc.) or removing those features entirely. We have chosen the latter for our experiments.

3.4 Pre-processing the Dataset for SVR

For our classical machine learning approach we have chosen to use Support Vector Regression (SVR) [3]. In order to use SVR with temporal data such as this, we need an additional pre-processing step to create the SVR specific dataset which involves using a sliding window approach [5].

From the previous step, we now have each entry in the dataset per patient representing a single day. In order to use this kind of data with SVRs we also have to average over multiple days, where this number is a hyperparameter. We chose a sliding window of size 10 as this size captured enough of the past mood changes without going too far back and overall producing better results in terms of our evaluation metric, Mean Squared Error (MSE). During this process, we average by feature over a period of 10 days, and then append the 10th day's mood to that new data entry. We repeat this process, incrementally sliding the window by one day until we run out of days for that patient.

Another part of this pre-processing step is removing features that are not correlated with the mood. It is important that we don't include bad features when training in order to train the SVRs more effectively. We chose to do this by calculating the correlation matrices on a per patient basis rather than calculating a general one for all patients. Intuitively, the features in our dataset should influence each patient's mood in a completely different manner. For example, a patient's mood might be highly influenced by the duration of usage of gaming applications but for another patient it might be completely irrelevant. We have thus chosen not to take into account any features that have less than a 0.05 correlation with mood for any given patient.

4 Learning Models

In this section we present the framing of our problem and the details regarding the implementation of our learning models. Our goal is predicting the average mood for the next day based on previous data. To do this we want to create a fit that lies as close as possible to the ground truth data. Therefore, we are solving a regression problem.

Our chosen metric for evaluation is Mean-Squared Error (MSE), which measures how close our fitted curve lies to the ground-truth curve, and can be computed as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_n - \hat{y}_n)^2$$

For benchmarking, we have implemented a baseline model for which the prediction for the next day is just the same value as the current day.

4.1 SVR

As mentioned earlier, the classical machine learning approach we have chosen for this problem is Support Vector Regression (SVR) [2][3]. We have created a separate dataset from the one that will be used in the ARIMA model which has been explained in detail in Section 3.4. To summarize, for SVRs we averaged over a window of size 10, instead of using the dataset in a day by day format as can be seen below.

	period	attribute1	attribute2	...	mood(target)
patient1	t=1-10	value for t=1-10	value for t=1-10		value for t=10
patient1	t=2-11	value for t=2-11	value for t=2-11		value for t=11

Table 2: SVR dataset of a single patient.

We trained SVRs in a per patient basis and kept the 10 last datapoints of each patient as a validation set. In order to perform predictions we did a rolling forecast [4] by first trying to predict the first label of the validation set, then adding the first datapoint of the validation set to the end of the training set and retraining the SVR. We repeated this process until we had gone through the validation set for a specific patient and then computed the MSE for the predictions made for that patient.

After repeating the above process for each patient we averaged the MSEs and it resulted in a MSE of 0.449. The plot for the MSEs per patient can be found in Fig. 4.

4.2 ARIMA

Since our data is highly temporal, a natural choice for a predictor would be a time-series analysis technique. For this purpose we have chosen to implement the autoregressive integrated moving-average model (ARIMA) [1].

Patient-specific ARIMA We want to train one model per patient. In order to find parameters that are suitable for each individual, we have used a method that tunes the parameters dynamically by minimizing the Akaike Information Criterion and the Bayesian Information Criterion. We are enforcing stationarity on the data and constrain the values of the AR and MA parameters to not be higher than 10, although most trained models tend to have AR and MA components less than 3.

We have kept the last 10 reported days of each patient as our test set and, since we are only interested in the prediction of the following day, we do a

rolling forecast [4], which involves retraining our algorithm after each time step. The MSE values for these forecasts are then computed for each patient and finally averaged over all patients to get the overall MSE for this method. Before training, we have dropped the features that are below the 0.05 threshold of correlation with respect to the target variable. This step has also been performed dynamically.

Multi-step forecast on the aggregated dataset In addition to our previous models which only predict one step at a time, we have implemented an ARIMA model for the aggregated dataset, to test whether we can make a forecast based on data from all patients.

We have kept 80 percent of our data for the training set and the rest for the testing set. Our goal is to predict the remaining test set as a single forecast.

In order to determine a good set of parameters for this model, we ran an autocorrelation analysis. The autocorrelation and partial autocorrelation plots for up to 25 lags can be seen in Fig. 3.

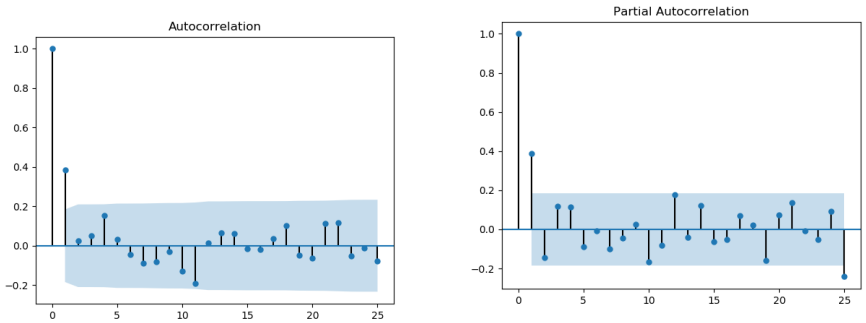

Fig. 3: Autocorrelation and partial-autocorrelation plots.

The sharp cutoff after the 1st lag suggests that our series is underdifferenced and, as such, we will add an I term of 1. These graphs also suggest that a lag of 2 should be sufficient to produce good results, since the autocorrelations beyond the 2nd lag are close to zero. Furthermore, since the effects of AR and MA terms can cancel each other out, we choose to not include any MA terms for our model. As such, the final configuration is an autoregressive component of 2 and an integration component of 1, with no moving average. The forecast generated by this model can be seen in Fig. 5.

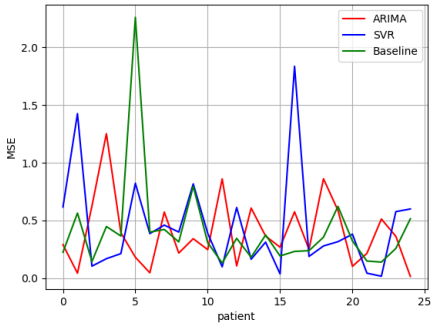

Fig. 4: MSE of patient-specific ARIMA, SVR and Baseline models.

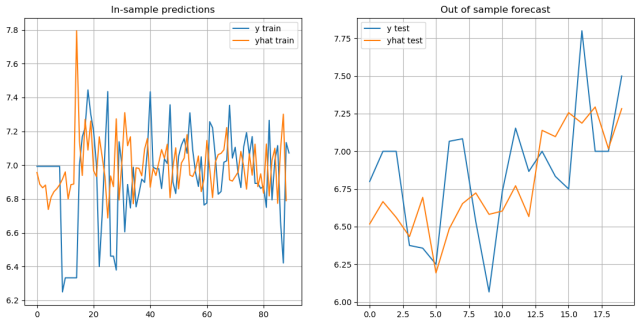

Fig. 5: In-sample and out-of-sample forecast for the multi-step ARIMA model.

5 Results and Interpretation

Overall, ARIMA seems to be the model that is most closely fit to the non-aggregated datasets with the Baseline and SVR in second and third place respectively in terms of mean MSE. Looking at Fig.4 however, ARIMA and SVR are both outperforming the Baseline model in terms of being the best predictors in a per patient basis and it can be observed that the Baseline model is better only for the first patient.

Our results for ARIMA when training on the aggregated dataset and doing multiple step predictions using an evaluation set from that dataset are a training MSE of 0.070 and a testing MSE of 0.121.

In retrospect, we found no model that vastly outperformed the others and as a result both ARIMA and SVR as well as a Baseline model should be considered when modelling a forecast system.

	mean	variance	std. dev.	min	median	max
Baseline	0.412	0.167	0.409	0.132	0.345	2.258
Arima	0.396	0.084	0.291	0.016	0.362	1.251
Svr	0.451	0.174	0.417	0.017	0.387	1.834

Table 3: MSE statistics over individual patient datasets

6 Conclusion

To summarize, we first pre-processed the dataset we were given and then evaluated the performance of three different patient-specific models. No model seemed to be distinctly better than another one for this dataset and that's perhaps due to human nature being mostly unpredictable by this feature set. Given more and better correlated features could possibly provide us with a richer dataset to train models that are much better at predictions.

Sharp increases or decreases in mood that are unprecedented for a specific patient, however, pose a big problem in predictions. Thus, using a more sophisticated machine learning approach, such as LSTM [6], that is able to detect and predict patterns over a large dataset of patients might be a better option than both ARIMA and SVR if predicting such a deviation is crucial to an application.

The results we got are, in any case, not to be dismissed and for most patients the predictions made were very close to the ground truth.

References

1. Ratnadip Adhikari, R. K. Agrawal "An Introductory Study to Time Series Modeling and Forecasting" https://arxiv.org/pdf/1302.6613.pdf
2. Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.
3. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A. and Vapnik, V. Support vector regression machines. Advances in Neural Information Processing Systems, 9:155–161, 1997 https://papers.nips.cc/paper/1238-support-vector-regression-machines.pdf
4. Zhihua Wang, Yongbo Zhang, and Huimin Fu, "Autoregressive Prediction with Rolling Mechanism for Time Series Forecasting with Small Sample Size," Mathematical Problems in Engineering, vol. 2014, Article ID 572173, 9 pages, 2014. https://doi.org/10.1155/2014/572173.
5. Yuya Suzuki, Hirofumi Ibayashi, Yukimasa Kaneda, Hiroshi Mineno, Proposal to Sliding Window-based Support Vector Regression, Procedia Computer Science, Volume 35, 2014, Pages 1615-1624, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2014.08.245.
6. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Comput. 9, 8 (November 1997), 1735-1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735