On Forecasting Mood from Smartphone Use with Machine Learning

Thomas Bellucci[1 [2710299]], Qingzhi Hu[2 [13167200]], and Chih-Chieh Lin[1 [2700266]]

[1] VU University, De Boelelaan 1105, 1081 HV Amsterdam, The Netherlands
{t.bellucci, c2.lin}@student.vu.nl
[2] University of Amsterdam, Spui 21, 1012 WX Amsterdam, The Netherlands
qingzhi.hu@student.uva.nl

Group 55 DMT-2021

1 Introduction

In this report we present the results of a research project on one-day mood forecasting. The task was to predict the average mood of a smartphone user on a particular day from past days of mood ratings and phone use. To this end, a collection of smartphone logs was provided in which measures of user activity and mood were collected over time. From this time series data, a set of relevant predictors were identified following an exploratory data analysis and literature study, after which two regression methods were examined; a gradient-boosting machine (GBM) and a recurrent neural network (RNN).

In what follows we will elaborate on our data analysis, feature selection, modeling rationale and evaluation, and reflect on the results that were obtained.

2 Exploratory Data Analysis

In order to gain a thorough understanding of the structure and quality of the provided data and identify dependencies among its attributes, an exploratory data analysis (EDA) was performed.[3]

The data was provided in the form of a collection of log files containing time-stamped attribute-value pairs as shown in Table 1. The logs included roughly 7

[3] A test partition was held out as to not inflate evaluation statistics (see section 3.5).

Table 1. Snapshot of the log data for user AS14.02

User ID	Timestamp	Attribute	Value
AS14.02	2014-03-18 20:42:12.348	appCat.social	16.042
AS14.02	2014-03-18 22:00:00.000	circumplex.arousal	0.000
...	...	...	...
AS14.02	2014-03-19 22:00:00.000	mood	8.000
AS14.02	2014-03-19 22:08:31.105	screen	52.920

weeks of mood ratings from 27 users and listed attributes such as screen activity, messages sent, calls made and app usage (e.g. social and weather app usage).

In Table 2, various statistics are shown for each attribute in the data. Here it can be seen that certain attributes contained missing data in the form on NaNs and that some attributes did not comply with their prescribed value ranges (e.g. some *appCat* attributes had negative minimum values despite being defined in terms of time). Furthermore, not all attributes were recorded on every day as is evident from the considerable percentage of missing values.

Table 2. Descriptive statistics of the dataset. Attribute statistics were computed over the development partition of the data (see section 3.5). *AppCat* attributes were combined for the sake of brevity. For the *Missing* column, data is considered missing if no value was recorded for a user on a particular day.

Total records	Total mood ratings	Total users	Total days	Total days (with mood rating)	Avg. days per user	Avg. days per user (with mood rating)	Avg. ratings per user
376.869	5628	27	1973	1268	73.1	47.0	208.4

Attribute	Description	Expected range	NaNs (%)	Missing (%)	Mean	Min	Max
mood	Self-rating of mood	$[1, 10]$	-	31.857	6.936	1.000	10.000
circumplex.arousal	Self-rating of arousal	$[-2, 2]$	0.96	31.857	0.071	-2.000	2.000
circumplex.valence	Self-rating of valence	$[-2, 2]$	3.09	31.857	0.688	-2.000	2.000
activity	One hour user activity score	$[0, 1]$	-	36.870	0.114	0.000	1.000
screen	Duration of screen use	> 0	-	35.498	68.051	0.035	9867.007
call	Call made	1	-	34.547	1.000	1.000	1.000
sms	SMS message sent	1	-	63.548	1.000	1.000	1.000
appCat.*	App use duration (categorized) (e.g. social, weather, game)	> 0	-	65.531	33.322	-82798.871	33960.256

In addition to descriptive statistics, the day-average mood and the seasonal decomposition of the mood ratings were visualized over time for each patient. As can be seen from the examples in Fig. 1, mood was not rated systematically for each user and often several consecutive days of mood ratings were missing. Furthermore, from the seasonality plot we can see that there are repeating short-term cycles in the series of approximately one week in duration.

Lastly, an all-to-all correlation matrix was computed to infer which attributes were predictive of next-day's mood and to guard against correlated features.[4] The correlation analysis did not show significant correlations among attributes; the highest correlation observed was between the daily averages of the *circumplex.valence* and *mood* attributes with a Pearson correlation of only 0.61 [6].

3 Method

3.1 Feature Engineering

As stated in the introduction, the aim was to predict the average self-reported mood on a day based on mood ratings and phone usage of past days. To achieve

[4] This is important as correlated features may degrade model performance [10].

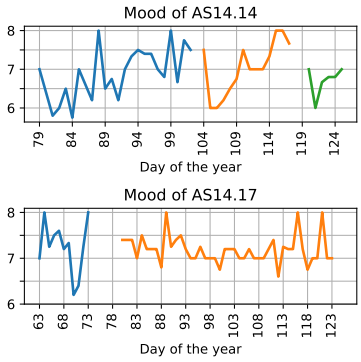

Fig. 1. Average mood over time for patient AS14.14 and AS14.17

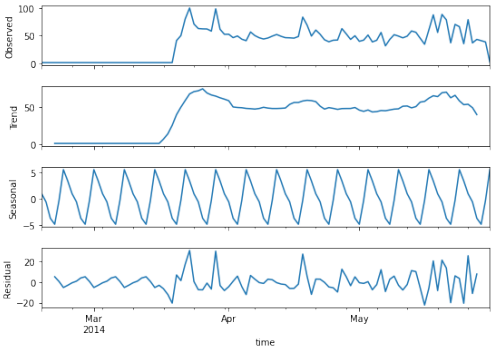

Fig. 2. Seasonality plot of mood w.r.t the day.

this, relevant features were identified by means of a literature study followed by feature selection in which noisy and non-informative features were removed.

The literature study provided insight regarding predictive features to perform mood prediction; for one, recent research has linked mood to irregularities in sleep and the disruption of the circadian rhythm [12,16]. In light of this, features related to user activity throughout the day were incorporated; for each day, the *activity* and *screen* attributes were binned into bins of 4 hours (i.e. from midnight to 4am, 4am to 8am, etc.) and incorporated as 6 additional predictors into the feature set. This way, models would be able to pick up on information regarding user activity at unusual times indicative of a disruption of circadian rhythm.

Moreover, it is well-known that mood is inherently tied to the day of the week and, according to recent empirical findings, mood can be expected to increase on weekends and Fridays [17]. As such, for each day the day of the week was extracted from the timestamp, one-hot encoded and used as a categorical feature.

Social interaction has shown to be a reliable predictor to discriminate between moods [16]. Although no direct measurement of social interaction was available, the total number of text messages over a day, the total number of calls made and social app usage may provide useful proxies and were thus incorporated.[5]

Lastly, weather information from external weather sources was considered as weather is known to correlate with mood [8]; however, as the data was anonymized and the places of residence of the users were unknown, this information was excluded. In the end, 35 features were identified in the literature study which are listed in Fig. 3.

3.2 Feature Selection

In order to assess the usefulness of the identified predictors and rid the feature set of non-informative features, a wrapper-based feature selection was used [1]. In

[5] Sum was computed as the call and sms message variables were binary in the dataset.

this approach, we started with an empty feature set and incrementally extended it with additional features; as the first feature the one was selected which, on its own, achieved the minimum time series cross-validation error when used to fit a GBM model.[6] In later steps, features were chosen from the remaining features that, together with the previously selected ones, decreased the cross validation error the most. This process was repeated until no features were left to add.[7]

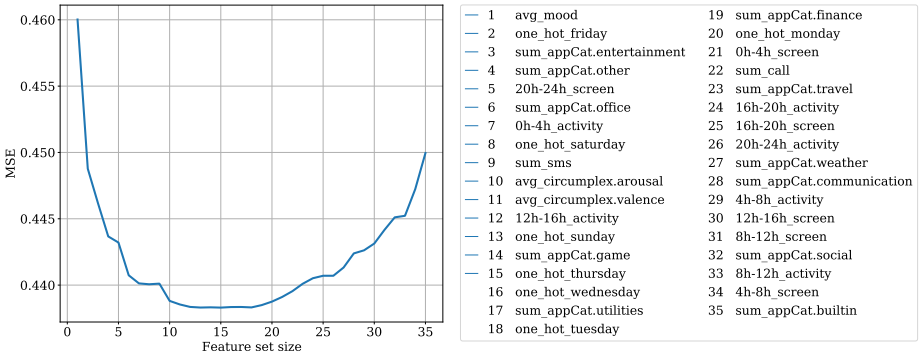

Fig. 3. MSE loss of a GBM model as a function of feature set size. As the ordering of feature insertion varied slightly from trial to trial, the MSE loss was averaged over 10 trials of 10-fold time series cross validation. The final feature set is marked in blue.

The resulting loss curve can be seen in Fig. 3, showing the MSE loss when adding features to the feature set in order of predictive value. Here it can be seen how many of the previously identified features, such as the 10h to 24h screen time, were useful in predicting mood, whilst other attributes did not contribute much to the decrease in loss (e.g. *circumflex.valence*) or even caused overfitting (e.g. *appCat.travel*). In the end, the first 15 features reducing the loss were kept.

3.3 Data Wrangling and Imputation

As stated in the introduction, machine learning algorithms were applied to the dataset in order to forecast mood; however, to use the identified features in a supervised setting and perform feature selection, the data needed to be transformed into an instance-based format first. To achieve this, timestamps were rounded to days, sorted by user and time, and feature values were computed for each measured day to obtain a data matrix as shown in Table 3. As the data contained missing values and the number of data points was limited, features were imputed with zeros in case no value was available on a particular day; zeros were used as missing values indicate that some action was not done by the user. If a mood rating was missing time points were discarded.

[6] Time series cross validation is explained in section 3.5.
[7] This is often referred to as *bottom-up selection* or *sequential forward selection* [1].

Table 3. Simplified snapshot of instance-based matrix with time-shifted target

User ID	Target date	GBM + RNN				GBM only				Target
		avg_mood	sum_sms	...	one_hot_sunday	arousal_lag_1	arousal_roll_mean_3	arousal_ewm_lag_7	...	
AS14.01	8	6.250	1.0	...	0.0	5.000	0.991	5.323	...	6.200
AS14.01	9	6.200	0.0	...	0.0	6.250	0.832	6.520	...	6.500
...	...	...	...	...	...	...	...	...	...	...
AS14.33	149	5.400	3.0	...	1.0	6.333	-1.012	6.771	...	5.500
AS14.33	150	5.500	2.0	...	0.0	5.400	-0.153	7.031	...	5.666

For the non-temporal GBM model, additional lag features were incorporated for all features considered in Fig. 3 as the value at time t was often strongly associated with the value at previous time steps [15]. In order to leverage seasonality, moving average [3] and exponentially-weighted moving average[8] were created for specified time intervals (i.e. past 3 days, 5 days and 7 days). As the number of features increases exponentially and limited data was available, Standard Gaussian Random Noise [7] was added to the features to avoid overfitting.

As recurrent networks are inherently capable of handling the temporal dependencies within the data, the RNN was trained on the reformatted data without rolling average or lag features. However, as we can assume independence between samples across users, the data matrix was split into individual time sequences belonging to the individual users. Furthermore, as RNNs are sensitive to the scale of the input data [11], the numerical features were standardized by z-scoring.

3.4 Models

We built a gradient boosting regression model using LightGBM (Light Gradient Boosting Machine) [9]. Gradient boosting is a type of boosting in machine learning and is based on the assumption that the best next model, when compared with previous ones, minimizes the total prediction error. The main concept is to define the desired outcomes for this next model in order to minimize error. LightGBM is a leading candidate for gradient boosting implementations produced by Microsoft in 2017 [9]. It has much of the same benefits as the commonly used XGBoost [4], such as sparse optimization, parallel preparation, various loss functions, regularization, bagging, and early stopping; however, LightGBM is generally faster. Whereas other implementations grow trees row by row, LightGBM grows trees leaf by leaf, selecting the leaf that it claims would result in the greatest loss reduction. This technique will produce more complicated and elaborate systems, resulting in a lower loss but face the risk of overfitting. However, techniques were implemented such as early stopping and bagging, which makes this implementation applicable even to smaller datasets.

In addition to the GBM, a recurrent Elman network [5] was implemented using the Pytorch library [14]. The rationale for an Elman network comes from previous research and the limitations of the dataset at hand; recent research has shown how recurrent neural networks are well-suited for mood prediction

[8] The moving average is constructed in such a way that it gives less weight to older findings. As a data point ages, the weights decrease exponentially.

due to their flexibility in modeling and ability to handle arbitrary length time series data [13,18]. However, as neural networks are often highly-parameterized they require lots of training data to mitigate overfitting. In light of this, a 2-layer Elman network was chosen as this model is architecturally simple and thus likely best equipped to cope with the limited data availability. Moreover, as techniques such as dropout have been introduced overfitting could be further minimized.

In order to gauge the performance of our models against a baseline, a persistence benchmark was implemented as well; that is, a benchmark predicting the average mood rating of the day prior to the day of prediction.

3.5 Validation and Evaluation Setup

The validation and evaluation setup was as follows; first, the data was split into a development and test set, using the last 20% of each user's data as testing data.[9] On the development set a hyperparameter tuning was then performed after which the final prediction performance was measured on the test set.

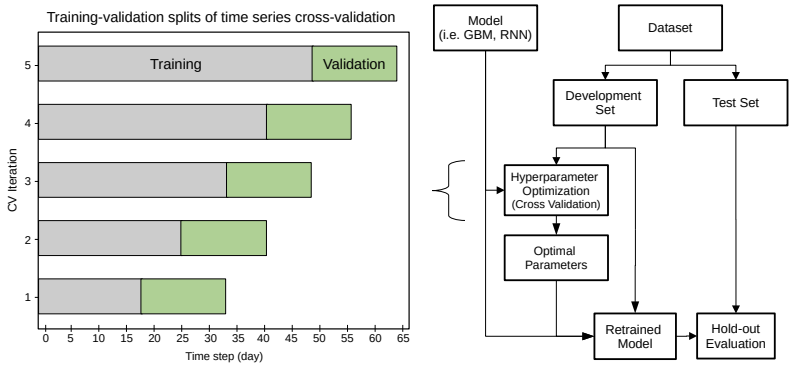

Fig. 4. Graphical representation of evaluation setup and hyperparameter tuning with forward-chaining time series cross-validation.

K-fold cross-validation (CV) [1] is a common technique to perform parameter tuning by randomly splitting the data into K mutually-exclusive folds, and holding out one fold when training the model for validation. However, when applied to time series, standard cross-validation can be problematic; because samples are randomly split, we may train on future samples and test on past samples and temporal coherence is disrupted. In order to mitigate this, Sklearn's time series cross validation[10] was used for hyper-parameter tuning as shown in Fig. 4. Contrary to K-fold cross-validation, successive training sets are supersets of

[9] This was possible as the data displayed reasonable stationarity (see Fig. 2).

[10] Ref: `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html`

the previous ones. Additionally, it populates the first training partition with a surplus data, which is always used to train the model.[11]

To find the optimal model parameters a grid search was used, which performed an exhaustive search over a large number of parameter combinations [1]. For the GBM model several parameter configurations were examined including variations in bagging fraction, number of leafs and early stopping. In the section above, several advantages of the GBM model were mentioned as well as its tendency to overfit. If the model starts to memorize the training data instead of learning general patterns, the training error will continue to decrease while validation error will slowly start to increase. One measure to regularize the model is to stop training early on, a practice known as *early stopping*.

For the RNN the number of hidden units and dropout rate were optimized. The model parameterization performing best on average over CV trials was selected. A full list of tuned hyperparameters can be found in Table 4.

Table 4. Hyperparameter settings obtained by hyperparameter tuning

Parameters	Meaning	Reason	Range	Value
Bagging fraction	Randomly select data without resampling	Reduces training time and overfitting	0.2 to 0.9	0.836
Bagging frequency	Sets bagging to every K steps	Reduces training time and overfitting	$[1, 3, 5]$	5
Early stopping	Halts training when after N rounds validation performance has not improved	Regularizes to prevent overfitting	$[100, 200, 300]$	200
Feature fraction	Percentage of features to select when training tree	Reduces training time and overfitting	$[0.2, 0.5, 0.8]$	0.5
Learning rate	The step size for performing gradient descent	Helps escape local/global minima	$[0.015, 0.02, 0.025]$	0.025
Max tree depth	The max depth for tree model	Mitigates overfitting with limited data	$[5, 7]$	5
Min data in leaves	Minimal number of data in one leaf.	Mitigates overfitting	$[20, 25, 50]$	25
Number of leaves	Max number of leaves in one tree	Mitigates overfitting	$[20, 25, 35]$	25
Hidden units	Number of units in the hidden state	Influences model complexity	1 to 10	3
Dropout rate	Probability of 'disabling' neurons during training	Avoids overfitting	$[0.1, 0.2, 0.3, 0.4]$	0.2

Once the optimal model hyperparameters were found, the models were retrained on the entire development data and evaluated on the held-out test set. Model performance was quantified primarily by the MSE metric due to its increased sensitivity to large errors.[12] In addition, MAE, maximum error and (S)MAPE were computed to obtain a complete view on model performance.

To test for statistical significance of model performance relative to the benchmark a one-tailed Wilcoxon signed-rank test was performed on the models' errors relative to the benchmark [20]. A non-parametric test was required as a paired test was needed (i.e. the residuals were from the same samples) and normality of the residuals could not be assumed. By means of this test the statistical significance of model performance could be established relative to the benchmark.[13] A standard significance threshold of $\alpha = 0.05$ was used.

Lastly, features were compared by using the permutation feature importance measure [2] and mood predictions were visually compared to their ground truths.

[11] Ref: `https://robjhyndman.com/hyndsight/tscv/`

[12] It is desirable to make sure the model accommodates the extreme mood values when fitting (e.g. 2 or 9) as these are indicative of large changes in mood.

[13] Under the hypothesis H_0: $E_{model} \geq E_{bench}$ and H_1: $E_{model} < E_{bench}$, where E is the residual error.

4 Results

Table 5 shows the performance of the models on the development and test set. Despite considerable differences in architecture, only minor differences in test set performance are observed between the GBM and RNN; on the training data, the differences in performance are more pronounced. Both the RNN and GBM yield performance approaching 0.400 MSE on the held-out test set. However, both do display some signs of overfitting to the development data.

Despite their minimal differences in performance, it can be seen that both models outperform the persistence benchmark considerably on both datasets. According to the Wilcoxon signed-rank test the observed improvements in performance from the RNN and GBM relative to the benchmark are statistically significant ($p = 0.032$ and $p = 0.041$, respectively), assuming $\alpha = 0.05$.

Table 5. Model performance on development and test set

Partition	Model	Metric					p
		MSE	MAE	MAPE	SMAPE	Max Error	
Development	Benchmark	0.589	0.563	8.520	8.388	4.000	-
	LightGBM	0.242	0.354	5.408	5.233	2.66	4.862e-22
	RNN	0.344	0.431	6.639	6.376	3.384	9.462e-18
Testing	Benchmark	0.538	0.530	7.541	7.467	2.800	-
	LightGBM	0.403	0.471	6.941	6.844	2.107	0.041
	RNN	0.391	0.470	6.507	6.635	2.615	0.032

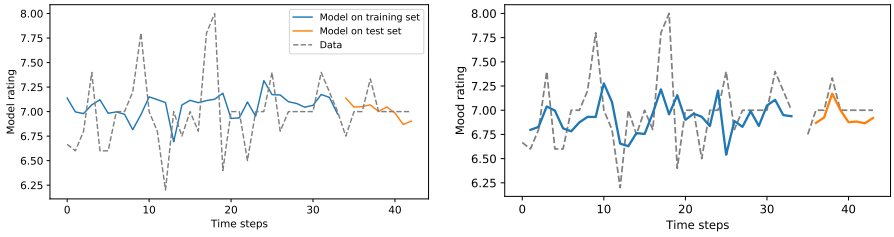

Fig. 5. Prediction of mood for user AS14.31; (left) GBM and (right) RNN

To gain insight into the predictions made by the model, the mood forecasts were visualized against the ground truth mood ratings over time in Fig. 5. Note the offset of the RNN model; as the model starts predicting one day ahead from day zero, there is no prediction for day zero itself. Both models show to have captured the underlying mood pattern and follow the fluctuations in mood approximately. However, one may notice how the model predictions tend to be offset by one day, showing a bias towards the mood of the day prior to the day of prediction.

Although both models capture the valleys and peaks in the mood time series, the models tend to smooth (average) the effects. This can be seen in Fig. 6 where the predicted mood values cover a smaller range than the actual mood values and the distribution peaks around 6 to 8.

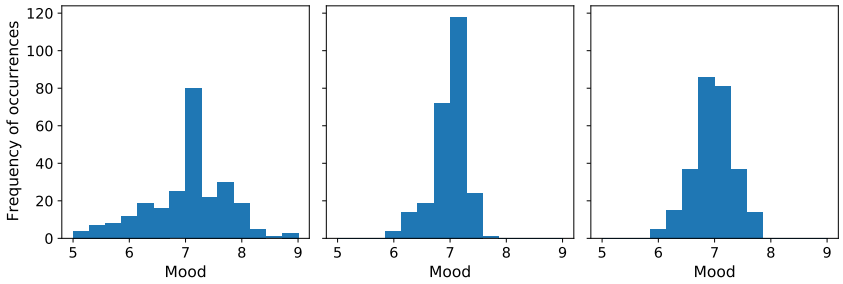

Fig. 6. Mood distribution in test set by (left) actual data, (middle) GBM, (right) RNN

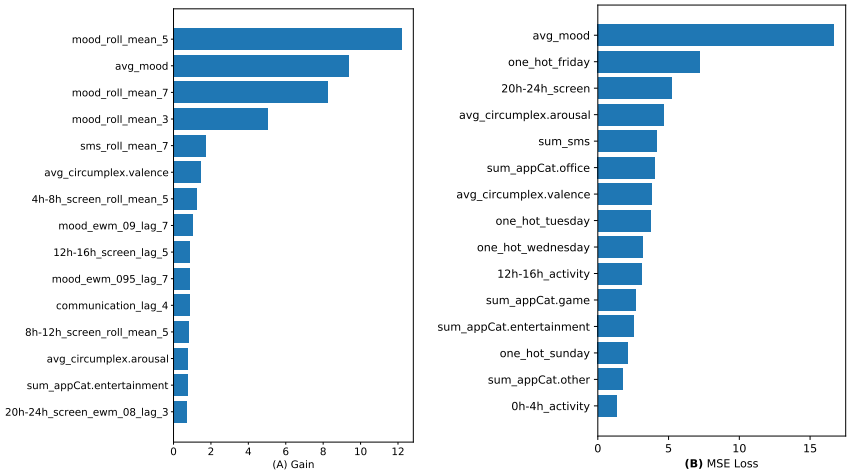

Fig. 7. Permutation feature importance according to (A) our gradient-boosting machine and (B) our RNN. MSE estimates were averaged over 200 trials.

The importance of features for the GBM and RNN models are shown in Fig. 7. Both models stress the importance of including the average mood of the previous day (*avg_mood*) into the model, explaining the bias seen in Fig. 5. As for the GBM, most information is gained through learning rolling-mean features of mood in the previous 3 to 7 days. This was to be expected as the average mood for most of the users during a week is often quite stable (see Fig. 1). The messages received during a week (i.e. *sms_roll_mean_7* and *sum_sms*) also show to correlate with mood. This is in line with previous findings supporting the claim that social

contact is a strong predictor for mood. Another important feature was found to be the screen time during the 4am to 8am period over the past 5 days, which may be due to its correlation with one's sleeping schedule. As for the RNN model, besides average mood in the previous day, it tends to favor using the features *screen time* and *activity* during 8pm to 12pm interval indicating that time spent before sleep may influence one's mood on the following day(s). In line with the GBM, the total number of messages and calls over previous days and time spent on entertainment apps were also used by the RNN.

5 Discussion

Despite showing promising results, the models have shown to have inherent limitations, limiting their applicability in mood forecasting. As for the RNN, the model showed to be highly susceptible to overfitting on the development data. By careful tuning of hyperparameters (i.e. dropout rate and hidden units), overfitting was largely minimized, however performance remained subpar on novel time points. In order to further minimize overfitting a feature selection was required, which added considerably to the complexity of the analysis and computational resources required; as such, the use of RNNs may have been prohibitively expensive if even fewer data samples were available. Lastly, as the RNN is inherently a blackbox prediction method, it was necessary to use external permutation-based feature importance measures in order to gain insight and gauge feature importance; as other algorithms may provide feature importance insights without external measures (e.g. linear regression and decision trees), these algorithms may be more appropriate when the primary objective is to gain insight.

Although the GBM enables faster training and lower memory usage, it faces similar issues of overfitting on the small dataset. We have applied cross validation with hyperparameters tuning to tackle this issue, however the time consumed in performing grid search cross validation can not be overlooked.

Despite its increased training time relative to the GBM model, the RNN does yield slightly better results than the GBM in terms of the MSE and MAE metrics as shown in Table 5, although not significantly; as both models showed signs of overfitting, both could not generalize well enough to unseen time points.

Moreover, Fig. 6 shows the distribution difference between GBM model prediction and the actual data. The frequency peak was at around 7 in the GBM predictions, which correspond with the one in the actual data. However, our model tends to be conservative on prediction which leads the deficiency on edge values (such as moods of 5 or 8). Although both the GBM and RNN were deemed statistically superior to the benchmark model, the former is still inferior to the latter by the perspective of the distribution. In order to capture extreme events and provide more accurate predictions for peaks and valleys in the time series, anomaly detection methods and data augmentations for time series data [19] could be considered in the future.

References

1. Alpaydin, E.: Introduction to machine learning. MIT press (2020)
2. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
3. Brownlee, J.: Introduction to Time Series Forecasting With Python: How to Prepare Data and Develop Models to Predict the Future. Machine Learning Mastery (2017)
4. Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., et al.: Xgboost: extreme gradient boosting. R package version 0.4-2 **1**(4) (2015)
5. Elman, J.L.: Finding structure in time. Cognitive science **14**(2), 179–211 (1990)
6. Freedman, D., Pisani, R., Purves, R.: Statistics (international student edition). Pisani, R. Purves, 4th edn. WW Norton & Company, New York (2007)
7. Graning, L., Jin, Y., Sendhoff, B.: Generalization improvement in multi-objective learning. In: The 2006 IEEE International Joint Conference on Neural Network Proceedings. pp. 4839–4846. IEEE (2006)
8. Howarth, E., Hoffman, M.S.: A multidimensional approach to the relationship between mood and weather. British Journal of Psychology **75**(1), 15–23 (1984)
9. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems **30**, 3146–3154 (2017)
10. Kuhn, M., Johnson, K., et al.: Applied predictive modeling, vol. 26. Springer (2013)
11. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.R.: Efficient backprop. In: Neural networks: Tricks of the trade, pp. 9–48. Springer (2012)
12. McClung, C.A.: How might circadian rhythms control mood? let me count the ways... Biological psychiatry **74**(4), 242–249 (2013)
13. Mikus, A., Hoogendoorn, M., Rocha, A., Gama, J., Ruwaard, J., Riper, H.: Predicting short term mood developments among depressed patients using adherence and ecological momentary assessment data. Internet interventions **12**, 105–110 (2018)
14. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703 (2019)
15. Pena, D., Tiao, G.C., Tsay, R.S.: A course in time series analysis, vol. 322. John Wiley & Sons (2011)
16. Sano, A., Amy, Z.Y., McHill, A.W., Phillips, A.J., Taylor, S., Jaques, N., Klerman, E.B., Picard, R.W.: Prediction of happy-sad mood from daily behaviors and previous sleep history. In: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). pp. 6796–6799. IEEE (2015)
17. Stone, A.A., Schneider, S., Harter, J.K.: Day-of-week mood patterns in the united states: On the existence of 'blue monday','thank god it's friday'and weekend effects. The Journal of Positive Psychology **7**(4), 306–314 (2012)
18. Umematsu, T., Sano, A., Taylor, S., Picard, R.W.: Improving students' daily life stress forecasting using lstm neural networks. In: 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). pp. 1–4. IEEE (2019)
19. Wen, Q., Sun, L., Song, X., Gao, J., Wang, X., Xu, H.: Time series data augmentation for deep learning: A survey. arXiv preprint arXiv:2002.12478 (2020)
20. Wilcoxon, F.: Individual comparisons by ranking methods. In: Breakthroughs in statistics, pp. 196–202. Springer (1992)