MLiP Competition Report: M5 Forecasting - Accuracy

Group 18 MLiP eating stroopwafels: Enrico Schmitz s1047521, Linda Schmeitz s1042449, Luca Carotenuto s1047465

June 14, 2020

1 Introduction

The M5 Forecasting - Accuracy[1] competition on Kaggle uses hierarchical sales data from Walmart. With a selection of three US States, and including item level, department, product categories and store details, it is possible to make predictions on different levels. Intermittent demand with zeros in time series can be a challenge for many time-series prediction algorithms. In this report we describe our approach to tackle the competition.

2 Approach

2.1 Model

At the start of the project we did research to decide which model we wanted to use. At the same time we did research to a baseline model and models already implemented in the competition.

Among other papers we found an analyses of the M4-Forecasting competition [1] and the winning model [2]. This model was based on LSTM, so this was an interesting model for us.

Simultaneously, we implemented a baseline model that used the last 28 days as a prediction for the next 28 days. This model scored 0.83770 on the validation set. When we examined the notebooks already implementing LSTM, there was not a single one scoring better than the 0.83770 of our baseline model, giving the impression that the differences in the competitions made LSTM less useful or harder to implement.

Notably most of the (at that time) top scoring notebooks were using gradient boosting, so we did research in boosting methods (Catboost, XGboost, light gradient boosting Machine). At first, we considered catboost (approach in appendix .1), but decided on light gradient boosting Machine (lgbm). Lgbm trains fast, gave less RAM issues compared to catboost and gives state-of-the-art performance [3, 4, 5].

For the lgbm we decided on using the parameters seen in table 1.

Parameter	**Setting**
Metric	rmse
objective	poisson
seed	200
force_row_wise	True
learning_rate	0.1
num_leaves	77
sub_row	0.7
bagging_freq	1
colsample_bytree	0.65
tree_learner	feature
num_boost_rounds	7000
early_stopping_rounds	300

Table 1: Table containing the parameters used in the lgbm. All non-specified parameters are kept at the default setting.

2.2 Evaluation Metric

Submissions are being evaluated on the Weighted RMSSE

$$WRMSSE = \sum_{i=1}^{42,840} w_i * RMSSE \tag{1}$$

where the RMSSE is a variant of the Mean Absolute Scaled Error proposed by Hyndman and Köhler[6] explained in more detail in the M5-Competitors-Guide[2]. The weight w_i is calculated for each of the 42,840 series and represents the cumulative actual dollar sales for each series in the period.

To get a deep insight into the performance of our model we calculate the WRMSSE for each level and plot them along with the overall WRMSSE which should ideally resemble the true score that Kaggle calculates internally at the end of the competition for the leaderboard ranking. Tests during the validation phase showed, that our calculated WRMSSE resembles their score up to a few hundredths. These small differences can be explained by differing weights because we are missing the true sales values for the predicted 28 days, which is why we needed to compute the weights based on the 28 days before.

2.3 Feature Selection

Since the sales around events are very inconsistent we wanted to create a feature to keep track of when the next event was. At first we wanted to add a feature which keeps track of the amount of events in the next 7 days. But then we found a discussion which stated that it would not improve the model [7]. Since the discussion stated that adding the amount of events would not improve the model we decided to see if adding the amount of days until the next event could help our predictions but this feature increased our error with 0.03.

Since it showed that adding one of those feature did not help we had to think of what other features could be added. And we ended up with several unique features.

[1]https://www.kaggle.com/c/m5-forecasting-accuracy

[2]https://mofc.unic.ac.cy/m5-competition/

First we added a feature which states the next upcoming event and a non-working days feature which indicates if the day is a day off for (most) Americans. The following days were considered as non-working days: NewYear, IndependenceDay, LaborDay, MemorialDay, Thanksgiving, Christmas and Saturdays/Sundays. These days where chosen since they are national holiday days or weekend days, of course between states there could be more days on which people don't have to work. But we decided on using nationwide non working days. Beside this, we also decided to add a quarter feature which keeps track of which quarter of the year it is.

For lags and windows every possible combination from the list of lags and windows was taken. Both lists consisted of 7, 14 and 28 days.

2.4 Post-processing

Several ideas where tested for the post-processing of the submission. This includes: an ensemble with higher level predictions, rounding the predictions and setting specific predictions to 0. For validation the last 28 days with a ground truth available (day 1886-1913) were used.

2.4.1 Post-processing: Ensemble

Our implementation of lgbm makes predictions directly on level 12. We argued that a higher level can contain a more consistent time series and is therefore easier to predict correctly. Since the different levels all contribute to the final score, we took the time to investigate the effect of predicting a higher level. This is done by taking the ground truth of a specific level and making an ensemble with the baseline. We choose the baseline (Figure 2) to be the prediction of the last 28 days provided with a ground truth. The ensemble was made by adapting the level 12 predictions per series with an equal amount, so the total of that series would match the higher level series prediction. This is done up to level 9, since 70 time series is still manually tweakable by experts. This is not realistic for thousands of time series anymore.

We planned on making use of this ensemble technique with actual higher level predictions. For this we tried implementing a holt-winters model for level 1 and one series of level 9. We also aggregated the input of the lgbm model to accept higher level input. Unfortunately, these simple modifications did not result in a lower MSE for that specific level and we ran out of time for this project to dive in optimizing models.

2.4.2 Post-processing: Rounding

A characteristic of the sales we noticed, was that it were always integer values. Our lgbm model predicted floats, so we tested what would happen if we rounded the output. We also tested what would happen if we only rounded the output that was already close (at most 0.1) from the rounded integer value.

2.4.3 Post-processing: Out of assortment

Another phenomenon we noticed was that a product at some time in the series could stopped being sold. The output of our lgbm model did not always catch up on this, so we decided to add post-processing to set the predictions to 0 if there were no sales for that product id in the last days before the prediction. The amount of days taken into account are 7, 14, 28 and 56.

3 Results

3.1 Feature selection

Comparison of Event features are made in table 3.

Furthermore, we tested the effect of adding lag features. Specifically, we tried adding 1 day and 365 days as lag features (See Table 2). However, both lag features worsened the score significantly.

Lags	WRMSSE
7, 14, 28	0.5326
1, 7, 14, 28	0.6573
7, 14, 28, 365	0.5731
7, 28	0.5477

Table 2: Table stating the WRMSSE for different used lags. The first row is what was used in our final model. Models are trained with dropping 1500 days due to memory isues. With 2000 rounds.

In table 3 the WRMSSE can be seen where different features where used to run the model. Here baseline is where no extra features besides lags and windows where used. In Figure 3 (in appendix) we can see all the features on the y-axis. UpcomingEvent, NWD and quater are considerd to be extra features. No events is different from the rest since no extra features were added but also the event names were removed. Showing small improvements when using extra features, except when using upcoming event.

Event features	WRMSSE
Baseline	0.5255
Quarter	0.5230
Upcoming Event	0.5408
Non working days	0.5133
Upcoming, NWD, Quarter	0.5326
No events	0.5434

Table 3: Table stating the WRMSSE after adding extra event features. 'Baseline' gives the performance of the submission without post-processing and feature selection. Models are trained with dropping 1500 days due to memory isues. With 2000 rounds.

3.2 Post-processing

The results for the post-processing can be found in the Appendix in table 5. Here the score of WRMSSE and the score split per level are presented. The first row 'Baseline' gives the performance of the submission without post-processing and feature selection. The ensemble results are given for an ensemble with the ground truth of level 1 till 9. In the 'Rounding' section, the results are presented where all predictions where rounded to integers ('All') and where only values where rounded that where in a range of 0.1 from an integer value ('<0.1'). Lastly the 'Out of Assortment' section gives the errors after setting the predictions to zero when the last x days nothing was sold.

3.3 Final Validation Results

Table 1 shows the validation results for the final model that we are using for our final Kaggle submission. We increased the number of rounds to 7000, early_stopping to 300 and we drop the first 1000 days. As you can see, the model performs better on lower levels than on higher levels.

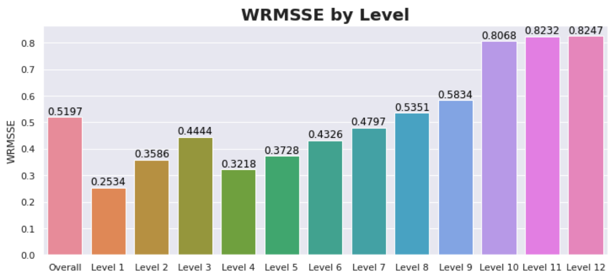

Figure 1: Final validation results

4 Conclusion & Discussion

4.1 Feature engineering

Feature engineering was done to improve our baseline model. The results seen in table 3 show that the feature upcoming was increasing the WRMSSE but when this was discovered there was not enough time left to rerun the final prediction. But in further research the upcoming could better be removed since it is likely the reason why our final prediction has an higher error than the baseline model.

4.2 Post-processing

Looking at table 5, rounding is not desired. As the WRMSSE either remains unchanged or deteriorates compared to the baseline and our final model produces comparable results.

For the out of assortment measurements the most successful one is based on the last 28 days. The improvement is visible in level 1 to 9, but not in level 10 to 12. We expected the opposite, as we make changes to specific ids not related to sales of other products. Table 4 presents the MSE on the subset of lines affected by the measurements before and after the changes. In all situations the MSE gets worse, further indicating that the measurement is not an improvement. We expect that the improvements seen in the higher levels is due to poor forecasting on the higher levels, that by chance should be lower than predicted.

The results gathered by the ensemble experiments indicate that when a better forecast on a high level can indeed increase the performance. A better forecast on level 1-9 has a positive effect on the WRMSSE score for level 1-9. Level 10-12 are mostly unaffacted, or have a slight decrease in performance.

4.3 Validation ideas

In this project we mainly validated on the last 28 days with a ground truth. We had some other validation ideas, as validating on the data exactly one year before the final submission month or use more data for training event features (as they often only happen once a year). Another idea was applying incremental learning with batches of 28 days, so the settings would change as time passes and more validation is possible. This would also make the model durable, which is often appreciated by a company.

	MSE before	MSE after
last 7d	1.47	1.60
last 14d	1.82	1.89
last 28d	2.08	2.17
last 56d	3.37	3.70

Table 4: The MSE calculated on the subset of rows in the validation data (day 1886-1913) that have all zeros in the last x days.

5 Evaluation of the Process

This project we all worked from home. Instead of a weakly working day, we only had a weekly meeting scheduled. We devided the work more and kept eachother up to date with new insights via Whatsapp in between meetings. The thing we noticed going wrong moreoften than the last project, was the quick fix of little details. One had commented a line out of the code, because suspicions of overfitting. A couple weeks later another had commented it in again. Also little mistakes could go unnoticed for weeks. As opposite to last project, where we checked each others screens regularly while sitting together.

None of use was familiar with time series on this big scale. We were disappointed that we did not manage to make a good comparison between catboost and lgbm and that we did not manage to significantly increase the score from our starting notebook, but definitely learned a lot about time series and are confident that we have gathered enough knowledge to tackle another time series project with a big dataset from scratch ourselves.

5.1 Evaluation of Supervision

The meetings with the teachers were very short and when we encountered big problems, there was not enough time to dive into it. However, we did really appreciated the meetings and the search directions that were given.

The meetings with the coach were a bit disappointing. We had one at the start and one at the end. At the start most of the classmates (including us) were not sure how to tackle the problem, thus making it hard to get a discussion going. Afterwards we had not spoken to any classmates until the final meeting with the coach, where the meeting was not very useful since everyone had different problems and another approach.

6 Contributions

Below are the contributions per team member listed.

6.1 Linda

- Attend weekly meeting with project group (+ thinking along with group members)
- Research existing models (e.g. M4 competition, kaggle notebooks, gradient boosting, LSTM).
- Research suitability and applicability of catboost.
- Investigate feature usage in kaggle notebooks.
- Create evaluation plan with dashboard. (Note: I outsourced the implementation to Luca.)
- Rewrite code to predict validation data and the final submission.
- Post-processing
- Write report

6.2 Luca

- Attend weekly meeting with project group (+ thinking along with group members)
- Develop baseline versions (Copy last 28 days, Exponential smoothing)
- Include WRMSSE Evaluation Dashboard
- Feature testing
- Setting up Google Cloud Platform for final prediction for final submission
- Create slide and present Flashtalk
- Write report

6.3 Enrico

- Attend weekly meeting with project group (+ thinking along with group members)
- Feature engineering
- Feature testing
- Hyper parameter tuning
- Comparing notebooks using lgbm (e.g. notice how different features were created and how trend was included).
- Write report

References

[1] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: Results, findings, conclusion and way forward," *International Journal of Forecasting*, vol. 34, no. 4, pp. 802–808, 2018.

[2] S. Smyl, "a hybrid method of exponential smoothing and recurrent neural networks for time series forcasting," *International Journal of Forecasting*, vol. 36, pp. 75–85, 2020.

[3] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," *arXiv preprint arXiv:1810.11363*, 2018.

[4] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," in *Advances in Neural Information Processing Systems 31* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 6638–6648, Curran Associates, Inc., 2018.

[5] J. Grohmann, N. Herbst, A. Chalbani, Y. Arian, N. Peretz, and S. Kounev, "A taxonomy of techniques for slo failure prediction in software systems," *Computers*, vol. 9, no. 1, p. 10, 2020.

[6] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, 2006.

[7] "Upcomming events as features." `https://www.kaggle.com/c/m5-forecasting-accuracy/discussion/147733`. Accessed: 2020-05-14.

[8] L. Sanzhapova and G. Silkina, "Machine learning: A retrospective, current state, and prospects for business use," in *SPbPU Science Week*, pp. 337–340, 2017.

Appendices

Appendix

.1 Catboost approach

In the first week we did research to the boosting methods and noticed that catboost was a recent method first released June 2017 [8]. It has been proven to potentially outperform other boosting methods as lgbm and XGboosting [3] [4] [5] and was specialized in handling categorical features without the need to pre-process. This was interesting as there were quite a few categorical features in the dataset: events, weekday, month, store, state. Although there where also important continuous features: price, sales and modifications of the sales (e.g. rolling means or lags). There where only 2 notebooks implementing Catboost, of which one 'm5_catboost' with a public score (0.57) below our baseline. Besides there where many notebooks implementing lgbm with a descent public score.

With the increased interest in catboost we made the plan to compare catboost to lgbm. We noticed the possible parameter settings from the two models were nearly identical. As there were already notebooks that had included some optimization for the parameters for lgbm, we tried applying the setup for the lgbm to the catboost model. Unfortunately, catboost with similar settings as the lgbm model required more RAM than available in Kaggle. Running catboost locally gave errors concerning drivers. Next, we did a step back and ran the notebook with the public score of 0.57 (as this one should function without RAM errors). When our run of that notebook resulted in a score of nearly 3, we decided to focus on lgbm. At the end of the project we did manage to get a score of 0.56 with the catboost algorithm, but we decided to not further include this in the report.

.2 Code

We published the code on Github:
https://github.com/lucacarotenuto/machine-learning-in-practice.
The code used for post-processing can be found in 'PostProcessing.ipynb'. The submission file used in this notebook is 'Postprocessing-baseline.csv'.
The notebook 'run_validation.ipynb' shows our validation run for the days 1886 to 1913.
The notebook 'run_evaluation.ipynb' shows our evaluation run for the days 1941 to 1968 which creates the final 'submission.csv' that we submitted to Kaggle.

.3 Figures/Tables

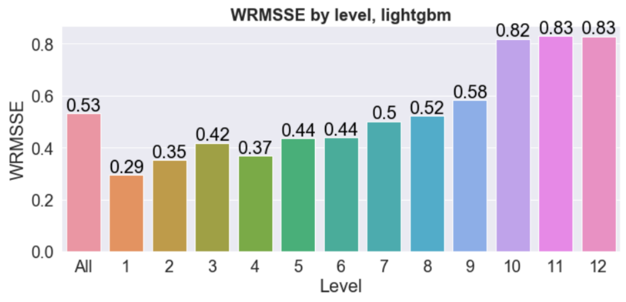

Figure 2: Baseline used for post-proccessing. (Note: The predictions are from a previous version of our model.)

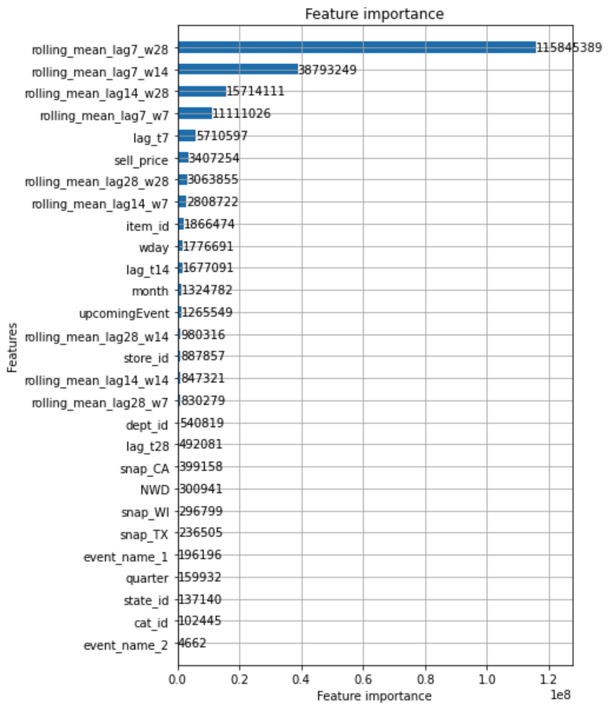

Figure 3: Feature importance given by the lgbm. With 7000 rounds

Score/level:		ALL	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12
Baseline		0.53	0.29	0.35	0.42	0.37	0.44	0.44	0.5	0.52	0.58	0.82	0.83	0.83
Ensamble	L1	0.46	0	0.22	0.35	0.24	0.36	0.36	0.45	0.48	0.56	0.82	0.83	0.83
	L2	0.42	0	0	0.28	0.24	0.36	0.28	0.41	0.44	0.54	0.82	0.83	0.83
	L3	0.39	0	0	0	0.24	0.36	0.28	0.41	0.37	0.5	0.82	0.83	0.83
	L4	0.42	0	0.22	0.35	0	0.27	0.3	0.42	0.44	0.54	0.81	0.83	0.83
	L5	0.38	0	0.22	0.35	0	0	0.3	0.34	0.44	0.49	0.81	0.83	0.83
	L6	0.35	0	0	0.28	0	0.27	0	0.3	0.36	0.49	0.81	0.83	0.83
	L7	0.29	0	0	0.28	0	0	0	0	0.36	0.41	0.81	0.83	0.83
	L8	0.28	0	0	0	0	0.27	0	0.3	0	0.36	0.81	0.83	0.83
	L9	0.21	0	0	0	0	0	0	0	0	0	0.81	0.83	0.83
Rounding	All pred.	0.64	0.43	0.44	0.5	0.54	0.64	0.55	0.65	0.61	0.7	0.85	0.86	0.85
	rest <0.1	0.53	0.29	0.35	0.42	0.37	0.44	0.44	0.5	0.52	0.58	0.82	0.83	0.83
Zero	last 7d	0.61	0.45	0.46	0.5	0.51	0.57	0.53	0.59	0.58	0.64	0.82	0.83	0.83
	last 14d	0.57	0.37	0.4	0.46	0.43	0.5	0.48	0.54	0.55	0.61	0.82	0.83	0.83
	last 28d	0.55	0.33	0.37	0.44	0.4	0.46	0.46	0.52	0.54	0.59	0.82	0.83	0.83
	last 56d	0.54	0.31	0.36	0.42	0.38	0.45	0.45	0.51	0.53	0.59	0.82	0.83	0.83

Table 5: Table containing the results obtained with the post-processing experiments.