2nd Competition - Report

Manuela Bergau
s4543645,

Stergios Morakis
s1047752

June 2020

1 Introduction

A promising application of machine learning is to make predictions on a defined future period based on past observations. While this type of forecasting does not necessarily differ from the 'human-based' forecasting methods, machine learning does bring various benefits. For instance, machine learning may be used to accelerate data processing speed, automate forecast updates based on recent data and most importantly, identify hidden patterns in the data, in cases where humans would struggle to do so.

For the second part of the course 'Machine Learning in Practice' our team, 'MaSterShi', has tackled the Accuracy challenge, one of the two parts that together consist the 5th Makridakis Forecasting competition (M5) on the Kaggle platform. Through this challenge, we are tasked to predict correctly the sales data of the well-known company Walmart, as a 28-day forecast.

The input data of this competition consists of 42.840 time series. These time series represent sales of 3049 unique items in 10 different Walmart stores across Texas, California and Wisconsin. For each item, one out of the 3 categories ('FOODS', 'HOBBIES' and 'HOUSEHOLD') is given. Moreover, up to three departments represent a single category resulting into a total of 7 departments. Files related to prices, promotions, and holidays are available as well.

Consequently, the data can be aggregated on different levels, i.e. item level, department level, category level and state level and is followed by explanatory variables consisting of item prices and calendar events. Submissions for this part of the competition are evaluated by computing the Root Mean Squared Scaled Error (RMSSE) metric for each time series and then averaged across all time series including weights, proportional to the sales volume of the item. Undoubtedly, this competition served as a great introduction to the emerging machine learning trends around time series and reflected the potential of what forecasting technologies can deliver.

2 Our Approach

As a first step, we conducted an exploratory research on background literature regarding the main topics of this competition and analysed the data in order to draw out valuable insight. Our findings, such as the the distribution of sales per category and store, as well as the increased values of 'FOODS_3' across the 'FOODS' category, gave us a good understanding of what we have to work with. Overall, through this analysis we got the impression that the patterns in the sales data would be sufficient for a machine learning model to get trained on to produce good predictions. After examining some popular notebooks for interesting ideas and started implementing our own models.

The distribution of items per category is shown in Figure 1. We can clearly see that items representing the 'FOODS' category are the most frequently sold ones, followed by the 'HOUSEHOLD' and the 'HOBBIES' categories. By taking a closer look at the number of sales per category (Figure 2), we observe the same kind of behaviour. Another thing we may notice is a repetitive pattern over time. For example, there is a striking drop on the sales of all categories on a specific day of the year, the 25th of December. Therefore, it is safe to assume that annual events have a major impact on the sales distribution.

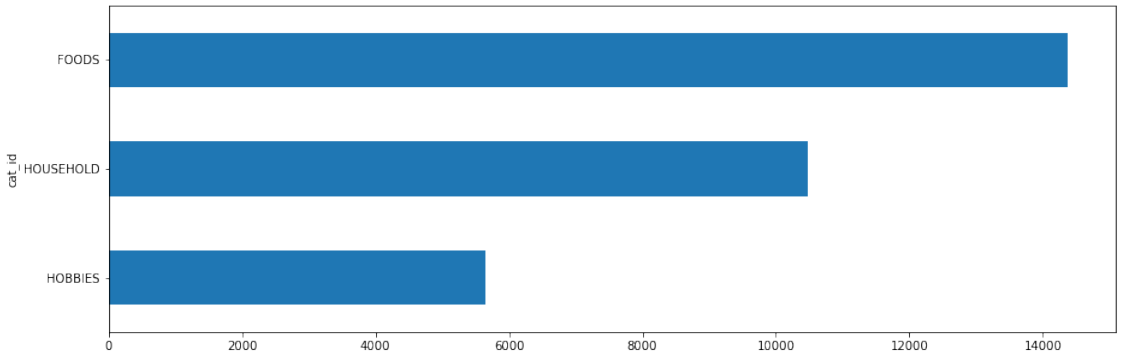

Figure 1: Number of Items per category

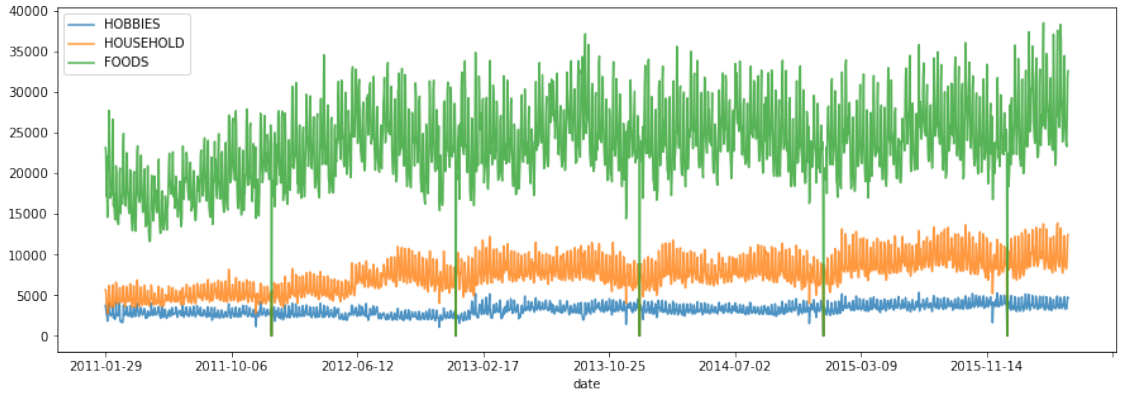

Figure 2: Number of sales over time

2.1 LightGBM

LightGBM was proposed by [1] and is a variant of a Gradient Boosting Decision Tree (GBDT). The advantage of using LightGBM over other GBDT algorithms is its core design, focusing on fast computational procedures on large datasets resulting in a complexity reduction. It is a Tree-based algorithm that offers various improvements in comparison to other tree-based approaches, i.e. the Gradient-based One-Side Sampling, which takes advantage of the fact that data points with larger gradients will contribute more to the information gain and the Exclusive Feature Bundling, which uses the sparseness of the feature space.

2.1.1 LightGBM Features

As features for the LightGBM, we used the existing columns of the available data, such as 'item_id', 'dept_id', 'cat_id', etc., and combined them with some of our own, as described below:

- *Time Features:*

  Features 'year', 'quarter', 'month', 'week', 'day', 'dayofweek' and 'is_weekend' consist the 'Time' feature group using for our model.

- *Demand Features:*

  The 'Demand' feature group was generated based on a 28 day shift setting of lags and rolling features. Different measurements were retrieved for the sliding windows of different sizes ranging from 7 up to 180 days. These features consisted of the standard deviation, mean, max, min, skew (defined as the unbiased skew over the requested axis) and curt (defined as the unbiased kurtosis over the requested axis) for each rolling window.

- *Price Features:*

  This feature group contains lags and rolling features fixed to a 1-day shift setting regarding the item price.

Figure 3 visualises the influence of each feature on the final result. Features 'sell_price', 'day' and 'item_id' played a major role, as expected, while the addition of 'rolling_max_t60' and 'rolling_mean_t7' features had a positive impact on the resulting score as well.

2.1.2 loss functions

As a kind reminder, the performance of a model displayed on the leader board is calculated by the 'Root Mean Squared Scaled Error' metric. In our first attempts, we used the RMSE, as most of the other kernels we found. Once we felt content with LightGBM's state, we started exploring other loss functions we could use and investigated their influence on the test set results. The use of the 'Tweedie' and 'Poisson' loss functions proved critical as we witnessed significant differences in terms of performance.

- *Root Mean Square Error*:

  Root Mean Square Error (RMSE) has been one out of the three loss functions we used for training our model and is considered a standard way to measure the error of a model in predicting quantitative data. It is measured as the square root of the average of squared difference between predictions and actual observations.

- *Tweedie*

  The Tweedie distribution is a special case of an exponential distribution. It has a power parameter p, which characterises the power relation between distribution mean and variance. To use the distribution as a loss function we need to maximise the likelihood of data. That is often done by applying the negative log-likelihood [2].

- *Poisson* Poisson loss function is often used to model count data [3]. The poisson distribution is a special case of the tweedie distribution [2].

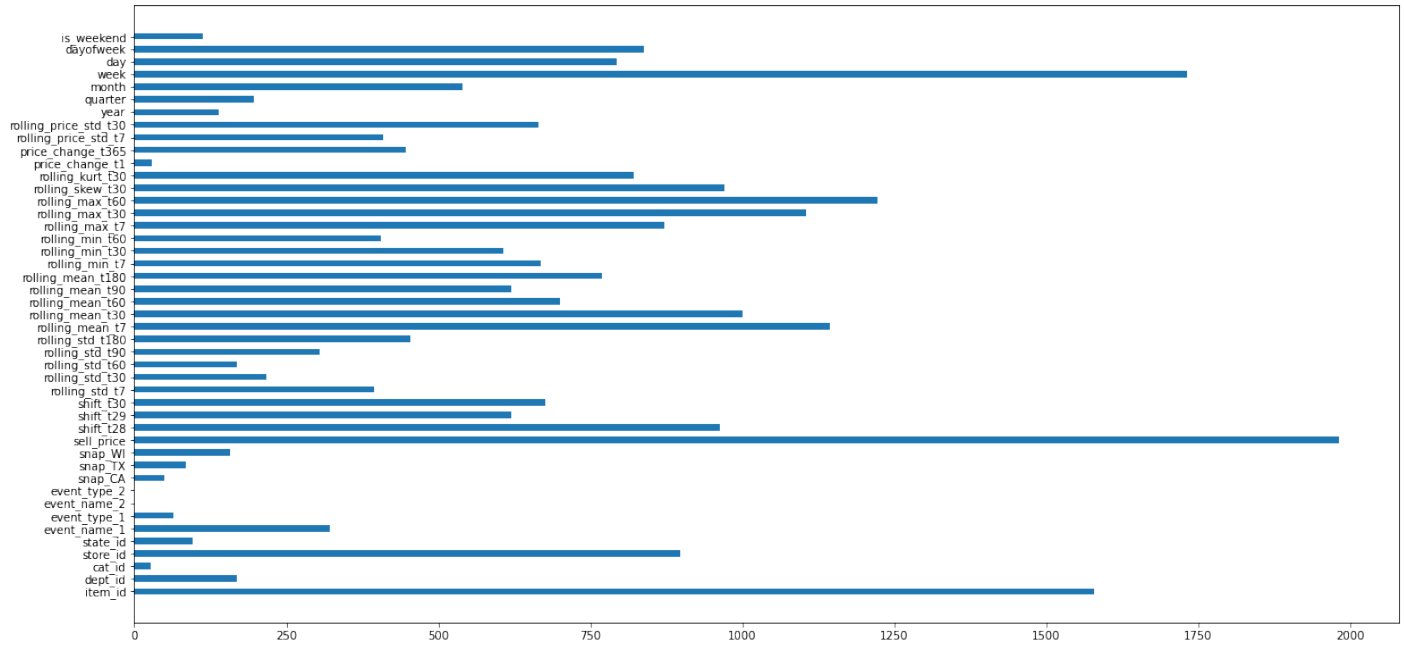

Figure 3: Feature importance of all features used

2.2 LSTM

Long short-term memory (LSTM) is a recurrent neural network (RNN) architecture, that is especially suited for sequence data [4]. LSTM is able to 'forget' long term data connections and thus deals with the vanishing gradient problem that normal RNNs can have during back-propagation. We used a model with three LSTM layers.

2.2.1 LSTM Features

Due to time pressure we were not able to include the same amount of features as we did in the LightGBM model. Nevertheless, we included the most important ones regarding the calendar information, i.e. the 'events' and 'weekday' values. There are up to two events per day given. The calendar contains the name of the event and the type (sport, religious ...). We encoded the strings using a label encoder. The 'weekday' feature consists of an integer ranging from 1 (Saturday) to 7 (Friday). The computed values are then merged to the sales data and will jointly compose our LSTM model's features. Surprisingly, 'weekday' feature had a negative influence on the score and if we had more time, we would proceed by using an One-Hot-Encoder.

3 Evaluation of the Model

We ran our experiments using the environment provided by Kaggle. The platform's CPU proved sufficient for running our scripts. Towards the end of the course, we tried to improve the computational speed with the provided GPU but it turned out that the use of LightGBM on GPU required quite a complex installation process. As the computational time was less than an hour on the platform's CPU, we did not proceed with the alternative option.

4 Best working model

Our best working model is the LightGBM model. There was an improvement on the resulting score when using either 'tweedie' or 'poisson' loss function over the 'root mean square error'. Specifically, we achieved the best result (0.63174) using 'Poisson' and the worst using RMSE (0.65884) on the same exact model. The code of our best working model can be found at https://github.com/ManuelaRosina/MLiP/blob/master/Project_2/m5-forecasting-lightgbm.ipynb

5 Discussion

It came as no surprise that the LightGBM model performed better than the LSTM model. The two submitted models have a difference in their scores of just about 0.09, indicating that additional features on the LSTM model would result in further decreasing that gap. Another noteworthy difference is that LightGBM has a runtime of 1 hour whereas the LSTM model runs in 20 minutes. A minor difficulty we encountered has been the 'out of memory' error caused by the huge size of the DataFrame, even when using memory reduction techniques. We could not figure out the reason for the said error, as rerunning the notebook would solve the problem most of the time.

6 Author Contributions

The following table displays each team member's main contributions throughout the M5-Accuracy competition.

Stergios Morakis	Manuela Bergau	Shima Yousefi
LightGBM model setup	LightGBM model features	Data Preprocessing
LightGBM parameter tuning	LSTM model setup	LightGBM model setup
Loss Functions	LSTM model features	

7 Evaluation of the Process

In general, our group worked well in this last competition. The current situation, i.e. working remotely, resulted in some minor miscommunication problems, such as overlapping responsibilities. Because of this, we invested more time than originally anticipated on our LightGBM model. We recall that at some point we were uncertain on how to proceed, but the latest teacher meeting accompanied by our TA's advice pushed us towards the right direction and we are thankful for this. Also, one team member dropped the course before we were done with programming our models. Thankfully, we were able to stay on track and despite the issues, we achieved some satisfying results. Ideally, we would like to sit together once a week within a team setting and focus on the given task, as in the first part of this course. Nevertheless we are proud of what we achieved given the situation.

References

[1] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 3149–3157, Curran Associates Inc., 2017.

[2] W. Shi, "Tweedie Loss Function." https://towardsdatascience.com/tweedie-loss-function-for-right-skewed-data-2c5ca470678f. [Online; accessed 12-June-2020].

[3] "Poisson." https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/poisson. [Online; accessed 12-June-2020].

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.