Forecasting the M5 Dataset: Clustering or not? Team C5

Yanchao Murong (14090759), Alex Papadatos (13435310) and Theodoros Katsikeros (13955152)
Applied Forecasting in Complex Systems 2022
University of Amsterdam

ABSTRACT

In this study, we focus on finding the optimal technique for forecasting food product sales over 28 days at a Walmart retail store in Texas. We find that ARIMA with predictors and Fourier k = 1 performs better than any other model used in our experiments, giving a final RMSE of 1.807 compared to 2.35 given by our baselines (NAIVE, SNAIVE, RW with Drift). Moreover, the effects of different regressors on different product-based clusters have also been investigated to see whether customized regressors to different clusters can improve the overall model accuracy or not.

1 INTRODUCTION

Forecasting is a crucial challenge in many industries, and accurate forecasts are critical for managing inventory, optimizing production, and making informed business decisions. However, in the retail and manufacturing industries, forecasting can be particularly challenging due to dealing with seasonal and trend effects and the impact of external factors such as economic conditions and market trends.

This research focuses on discovering the optimal model for forecasting the daily retail sales of a Walmart store located in Texas, US. More specifically, our scope entails only the food products contained in the M5 Forecasting dataset [1]. The forecasting horizon is four weeks (28 days) after the training dataset's end. Optimality is measured based on accuracy. More precisely, we use Root Mean Squared Error (RMSE) as the indicator of best forecasting accuracy. The lower RMSE value the better:

$$RMSE = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2} \quad (1)$$

where x_i is the predicted value, y_i is the actual value and n is the number of observations.

During our research, we perform an extensive exploratory data analysis to understand our data's main characteristics, such as seasonality, trends, and data quality. By establishing these main insights, we continue leveraging them to create our features. Our forecasting model then uses these features to make predictions.

During the modeling process, we find that we are limited to having fixed regressors for all products as they must be predefined before training the model. Nevertheless, food products can have intrinsic differences in seasonal availability and sensibility to calendar and events effects. Thus, we have also conducted some experiments to see whether product clustering can give us more insights and information so as to influence the choice of regressors per cluster and improve the overall accuracy.

All of the above supports us in answering the following research question:

- **RQ** - Will clustering products provide useful information in choosing more suitable regressors respectively and hence improve the accuracy compared to applying the same regressors to the overall model?

Literature Review

Several approaches have been developed for forecasting in retail industries, including statistical models, machine learning techniques, and hybrid approaches that combine elements of both. In the M5 competition, many models and techniques were evaluated, including traditional statistical models and more recent machine learning techniques such as gradient boosting and deep learning.

The prizewinner, YeonJun In, a student at Kyung Hee University, used an equal-weighted average of various LightGBM models [14]. LightGBM is a decision tree-based approach [7]. He constructed 220 models, half of which apply a recursive whereas the other half (110) a non-recursive forecasting method [2]. All models were optimised by maximising the negative log-likelihood of the Tweedie distribution [20]. The models acknowledged various identifiers, calendar-related information, special days, promotions, prices, and unit sales data.

In the second place, Matthias Anderer used an equivalent approach. Similarly to YeonJun In, he used an equal-weighted average of various LightGBM models [14]. The main difference in his approach was that he then adjusted by utilising multipliers based on the forecasts produced by N-BEATS [15]. N-BEATS is a deep learning model for time series forecasting and has been applied to various forecasting tasks in different domains. It is based on using a multi-headed deep neural network architecture to capture the long-term dependencies and patterns in time series data. In addition, the models were optimised using a custom asymmetric loss

function. He trained the LightGBM models using basic features of calendar effects and prices and the N-BEATS model was based solely on historical unit sales.

In the third place, Yunho Jeon and Sihyeon Seong used an equally weighted combination of 43 NNs containing multiple Long Short Term Memory (LSTM) [4] layers. LSTMs are a type of RNN [5] that are particularly well-suited for learning long-term dependencies. They do this by using special "memory cells" that can store information for long periods and by using gates that control the flow of information into and out of the memory cells. RNNs are a type of neural network that process sequential data, such as time series or natural language. They do this by using feedback connections that allow the network to maintain a "memory" of past input. This allows the network to incorporate information from the past when processing the current input. In their method, some models considered dropout [17] while others did not. Similarly to the winner, they considered Tweedie regression. They used Adam [8] as the optimiser and cosine annealing for the learning rate scheduling [9]. When it came to the NNs features, the authors used sales data, calendar-related information, prices, promotions, special days, identifiers, and zero-sales periods.

Following the third place, most of the top 50 submissions used a similar method to the prize winner. The majority of the solutions focus on the model itself, however, there are few attempts at optimising the feature engineering. Although the model plays a significant role in maximising forecasting accuracy, we believe that focusing on the features and leveraging as much information as possible from the given data will boost performance.

Competition and Dataset

The task of forecasting the accuracy using the M5 Forecasting dataset was introduced in the M5 competition that took place from the 2nd of March to the 30th of June in 2020[8]. The M5 competition is the fifth of the Makridakis competitions (known as M Competitions) that began in 1982 by forecasting researcher Prof. Spyros Makridakis [10] [11] [12] [13] [14]. They have drawn much attention from both academia and individuals, as they provide an unbiased proof of the best forecasting methods to be used for the specific tasks that are provided.

The M5 competition involves forecasting sales data for products in the retail and manufacturing industries over 28 days. Participants in the competition are given access to a large dataset. The competition's goal is to use this data to develop machine learning models that can accurately predict future sales.

The dataset given for the competition mentioned above was made available by Walmart. It includes the sale volume and price of 3049 products, classified into 3 product types and 7 product departments. The items were sold in ten stores in three states, adding up to 42840 series. The historical data range is from 2011-01-29 to 2016-06-19, summing up to 1941 days or about 5.4 years. For our task, we only use the third category of the *foods* type for the third Texas store. The total products for that are 827.

The dataset consists of mainly five parts:

(1) The calendar data, *calendar_afcs2022.csv*. This file contains all the dates within the range mentioned above, the weekday (incl. the id), day (incl. the id), month, event names and types and a binary indicator of whether food stamps are allowed for usage during that day.
(2) The sell prices data, *sell_prices_afcs2022.csv*. This file contains the store id, item id, week id and sell price.
(3) The train data, *sales_train_validation_afcs2022.csv*. This file contains the historical daily unit sales data per product and store for the first 1913 days.
(4) The test data, *sales_test_validation_afcs2022.csv*. This file contains the historical daily unit sales data per product and store for the last 28 days.
(5) The submission data, *sample_submission_afcs2022.csv*. This file contains the number of forecasts to be submitted for point forecasts, exactly 28 days (4 weeks ahead), starting at F1, F2, ..., F28.

An overview of the hierarchy of the data can be seen in the following figure:

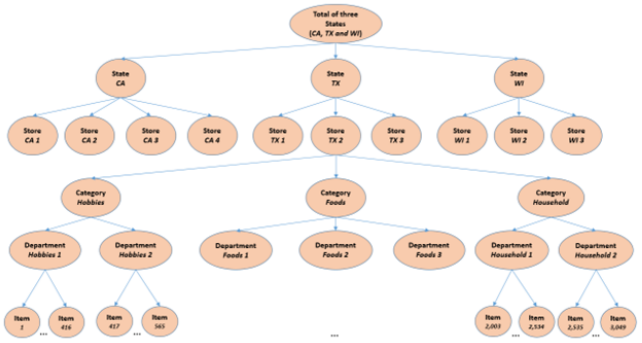

Figure 1: An overview of how the M5 series are organized. Figure by M Open Forecasting Center. (https://mofc.unic.ac.cy/m5-competition/).

2 FORECASTING METHODS

In this section, we will present some of the core activities followed during this research. We start by explaining the techniques we used to extract insightful information from the given datasets. In addition, the reader may find an extensive description of our feature engineering methodology.

Moreover, we present all of the features that our forecasting models use. Lastly, we briefly explain the forecasting techniques employed for this study.

Exploratory Data Analysis

It is important to have an in-depth understanding of the data sets used in the analysis to drive the prediction model and feature selection. By performing the exploratory data analysis, we can identify patterns and trendlines to understand the underlying structure and have a better insight. The analysis is including the data preparation by sorting and merging of the datasets and the creation of several visualization to help us understand the stationary and non-stationary behaviour of the data. Several packages as plotly,ggplot been used to provide information about the weekly sales sliced by attributes as events and seasons. For each visualization, we have included a brief caption.

Our EDA started by loading, merging and cleaning the given datasets. It is apparent from the beginning that the *sell_prices* does not contain prices for all the combinations of products and dates. To overcome this problem, we proceed by back-filling all sell prices. Doing so helps us, as we will see later on, to create lag features associated with sell prices per product. The sell prices of the food products seem to be concentrated around 2-3 dollars as shown in Figure 2.

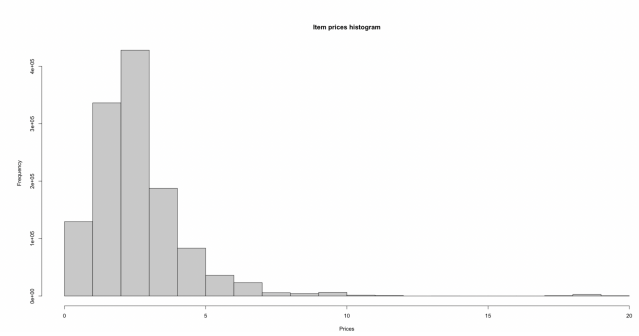

Figure 2: Prices ranges histogram

Continuing, looking at figure 3, one can deduce that there is a strong seasonality associated with sales. Some apparent peaks, few more consistent than others, possibly due to specific events. This is perhaps even more clearly shown by figure 4.

Examples of noticeable occasions are the end of each year, Christmas, and a high peak in August 2015, during the NBA finals. Moreover, it is safe to say that in general there are stable yearly variations. On top of that, following a seasonal and trend decomposition (STL) [3] we are able to strongly confirm the aforementioned seasonality in sales but also a weak trend (figure 5).

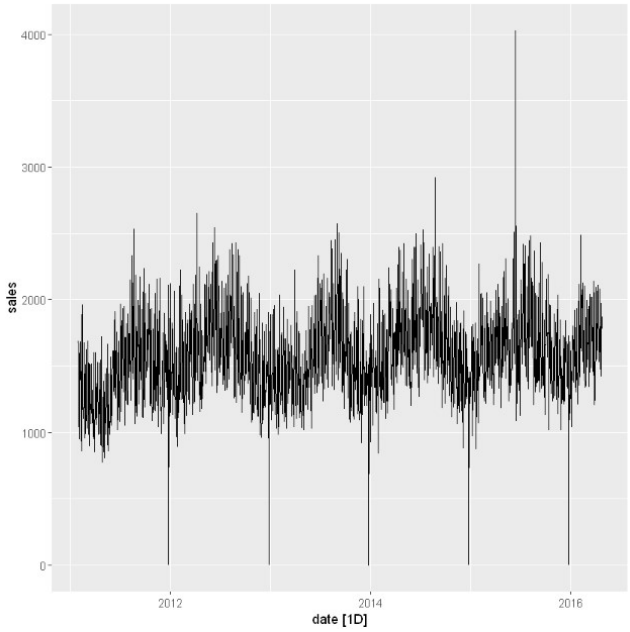

Figure 3: Daily total sales.

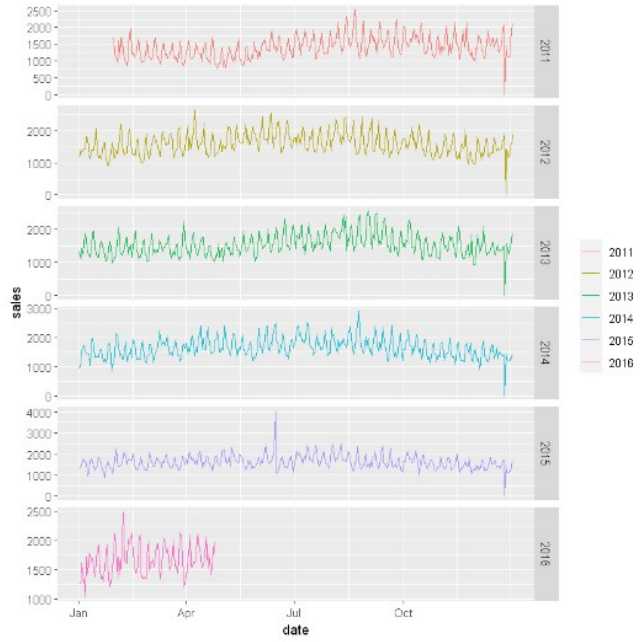

Figure 4: Per year analysis total sales.

Taking it a step further, we would like to understand whether there is a certain trend within the days of the week both on total sales and on individual items. By using the appropriate visuals, it is evident that when it comes to individual product level, each of them maintain their own trends

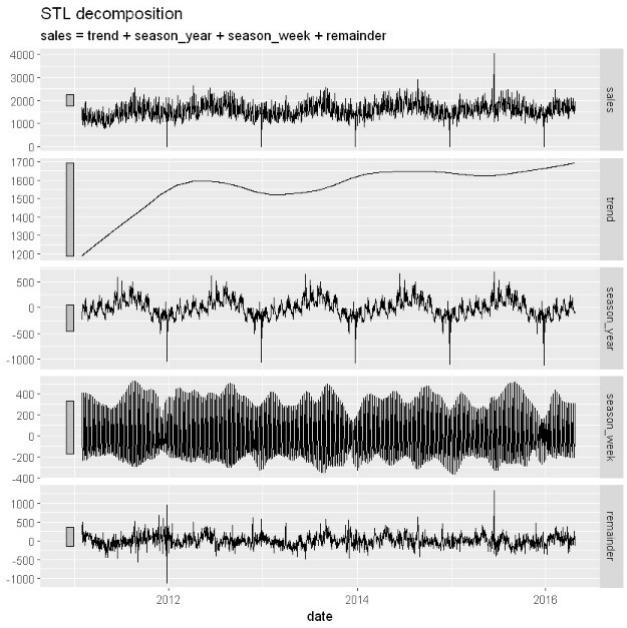

Figure 5: Seasonal and trend decomposition: Sales.

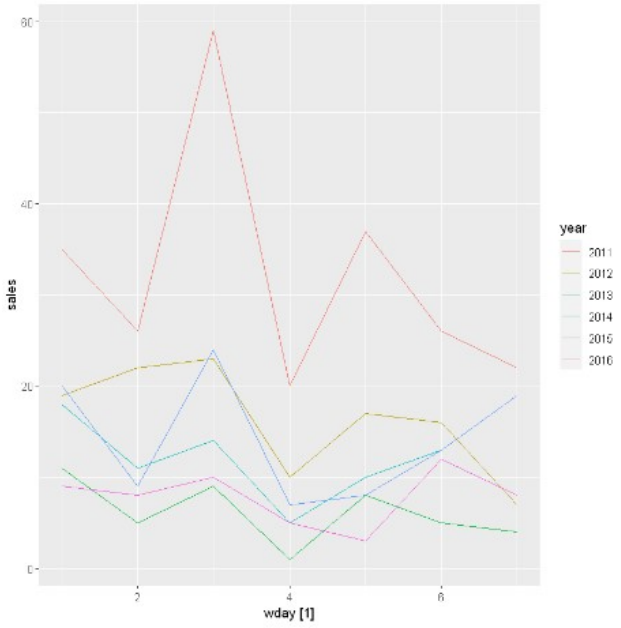

Figure 6: Individual product sales - product 3

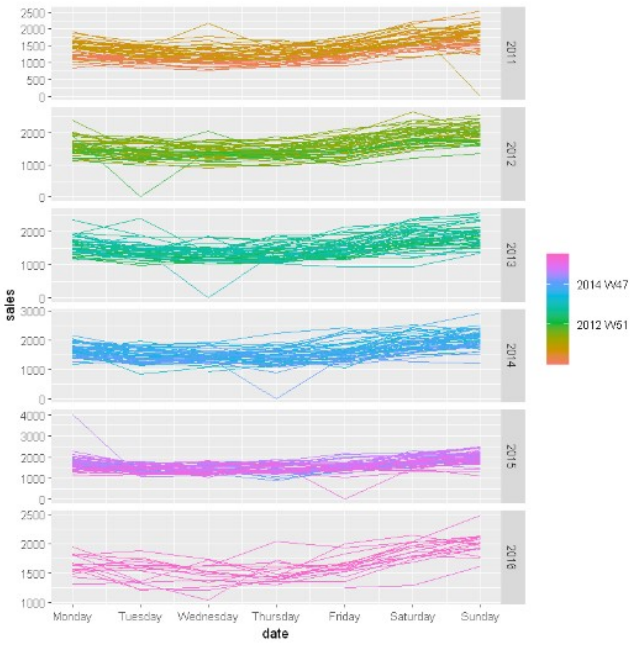

Figure 7: Weekly aggregated sales, per year.

and follow generally a random pattern. On the other hand, one could say that overall weekends perform better than the weekdays. The reader may direct to figure 7 for this information. However, on a daily aggregated level, one could say that there are obvious highs and lows connected to the weekdays. More specifically, Mondays or Wednesdays could be considered more 'profitable' compared to Tuesdays and Thursdays, as indeed shown in figure 6.

Moving forward and looking at the number of sales per month, we see compelling results. More specifically, we see that months close to summer, like June, July and August, are much more popular than other months. This is clearly shown in figure 9. However, if we reduce our scope to individual products, there are cases where we even see increased sales associated with a specific year, as shown in figure 8 .

Besides sales, we find it interesting to investigate the effect of the supplementary nutrition assistance programme (SNAP) and the overall days that allow food stamps to be used compared to days that do not. By counting the number of days SNAP is allowed, we realise that it is in place for most days. The numbers can be seen in figure 10. We see that approximately 66 per cent of the time SNAP is not allowed compared to 33 per cent which is.

Feature Engineering

Based on the previous exploratory data analysis, we have found some other relevant information such as the effects of the calendar and events that could explain some of the historical variations apart from past observations of the series. On top of that, we also find that some products share similar patterns while contradictory to aggregated time series

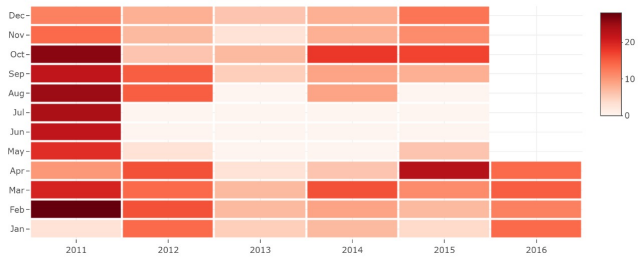

Figure 8: Heatmap of individual product sales across different years for product 3.

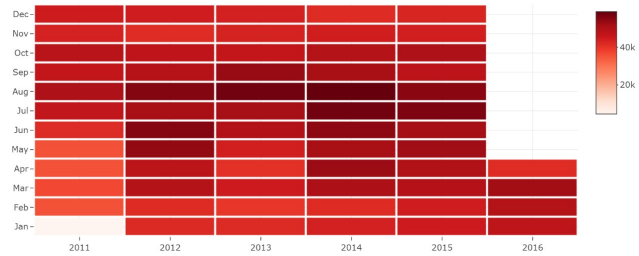

Figure 9: Heatmap of overall mothly sales across different years

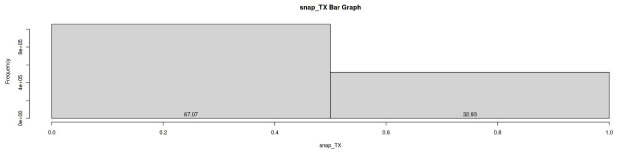

Figure 10: Bar chart of the frequency of SNAP

in seasonality. Consequently, we produce the following feature engineering in order to have more features that explain better the sales variations.

The calendar and events features aim to better explain some sudden changes which do not follow overall trends and seasonality in order to better fit the sales forecasting model.

The clustering results would not directly be used as a predictor as we actually fit each product with its own model. We mainly use the clustering results to do experiments to see if we could come out with more adaptable predictors per cluster due to their differences in seasonal availability and sensibility to calendar and events effects.

Calendar and Events Feature.

(1) ***Month***: This feature is self-explanatory. Following out EDA, we extract the month easily from the date and we check whether it is of importance when it comes to different models.

(2) ***Weekend***: This is a boolean indicator of whether a sale is made during the weekend. We expect this feature to be of high importance since, as discussed in the EDA, we observe that overall there is a higher number of sales during weekends compared to weekdays.

(3) ***SNAP_TX***: This feature stands for the supplemental nutrition assistance program (SNAP). In summary, this feature indicates whether using food stamps is possible during that day. This feature will be of great importance, as it will naturally drive more people to the store and increase sales. It is kindly given to us directly in the provided datasets.

(4) ***Has event near***: Naturally extending the information handed to us through the datasets, we leverage the event information by checking whether there is an event on a specific day and whether a day is near an upcoming or past event. The window used here was chosen arbitrarily to be of size two. We use *event_name_1* and *event_name_2* as the indicators of a future or past event around a day.

(5) ***Has two events***: Following the same logic as in the calculation of the previous feature, we check whether there are two events on the same day. Again, it is logical to expect two events to drive more sales compared to a one-event day or a typical day.

Clustering Feature. There are many features that could contribute to the clustering of supermarket products. Based on the dataset, we have initially selected the ‘total sales‘, ‘sales q1 percentage‘, ‘sales q2 percentage‘, ‘sales q3 percentage‘, ‘sales q4 percentage‘, ‘mean price‘.

Normalization is applied as we use euclidean as cluster distance. Log transformation is applied to mean price and total sales as they are positively skewed. After testing the co-linearity, some feature are dropped. Figure 2 and Figure 3 show the distribution and pairwise correlation before the preprocessing and after the preprocessing respectively. The three selected features ‘sales q1 percentage‘, ‘sales q4 percentage‘, and ‘mean price‘ demonstrate low correlation and relative normal distribution.

Hierarchical clustering [6] and GMM(Gaussian Mixture Model) [16] have been chosen for clustering experiments. Hierarchical clustering has been chosen for its advantage of handling large datasets without any assumption of data distribution. GMM has been chosen for its flexibility and ability to choose cluster number automatically.

(1) Hierarchical Clustering
Herbert Index (Figure 4) and D index [18] (Figure 5) both show that 6 is the best clustering number with a significant knee that corresponds to a significant increase in the value of the measure. Silhouette [19]

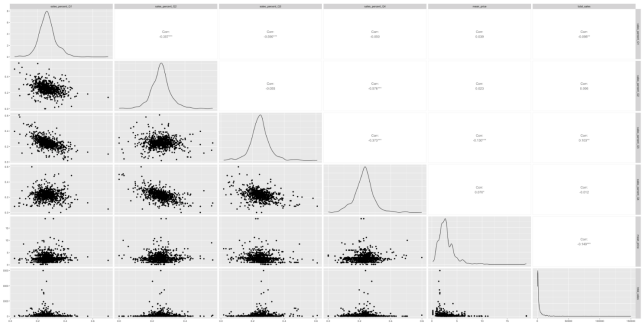

Figure 11: Initial features before preprocessing

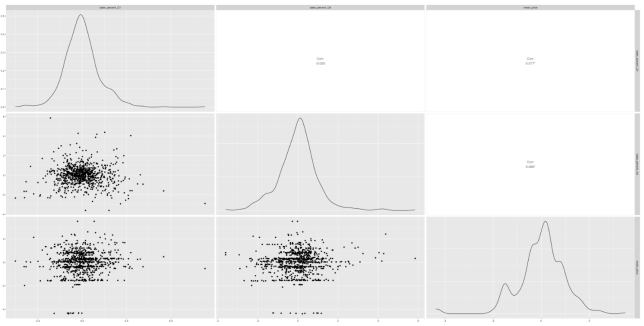

Figure 12: Selected features after preprocessing

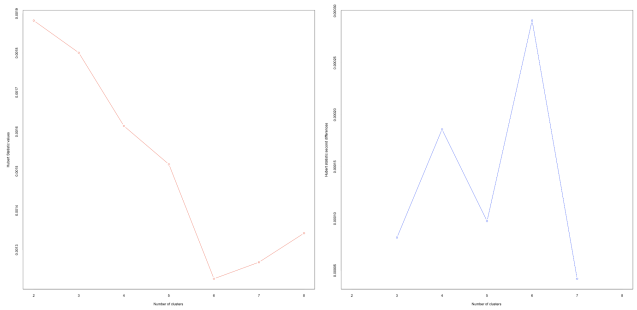

Figure 13: Herbert Index: a graphical method of determining the number of cluster

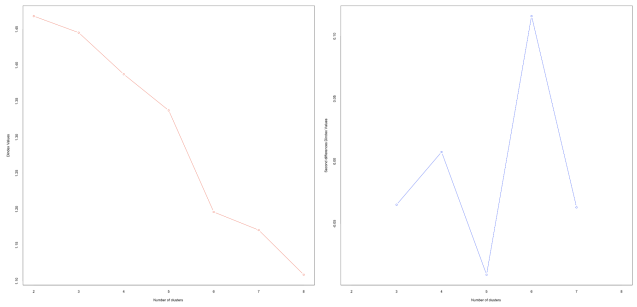

Figure 14: D Index: a graphical method of determining the number of cluster

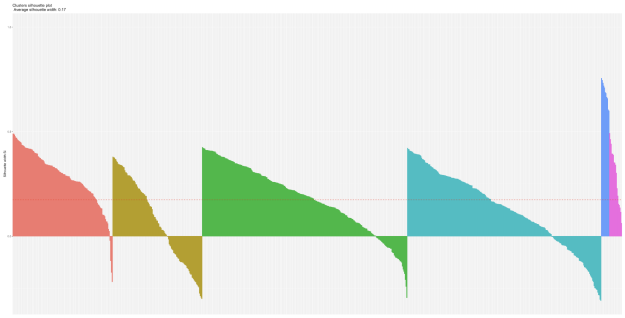

Figure 15: Silhouette Plot

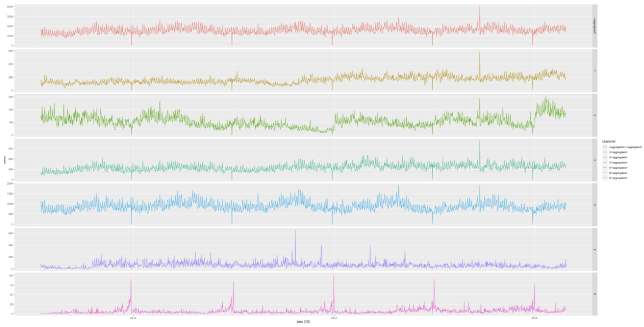

Figure 16: Hierarchical Time series after Hierarchical clustering with 6 clusters: The first red time series is the original aggregated time series without clustering, the subsequent 6 time series corresponding to the group time series for 6 clusters

plot (Figure 6) also shows that 6 is a proper number of clusters.

Table 1 shows the result of clustering. Cluster 5 and 6 are small in size compared to the other groups. By observing Figure 8, the major contributing factor of cluster 5 (green points) is the mean price that is lower than -4 (scaled value) while the major contributing factor of cluster 6 (yellow points) is the ‘sales q4 percentage‘, which means it has really high sales during quarter 4. These points may initially seem outliers but are actually important to take into account by stores as they represent important sales patterns.

From Figure 7, we observe some interesting seasonality segmentation. The green time series (cluster 2) demonstrate higher sales during the first quarter. The blue time series (cluster 4) demonstrate higher sales during spring and autumn but lower sales during the summer and winter. The purple time series (cluster 5) shows relatively weak quarter seasonality except for some peaks during special events. The pink time series (cluster 6) demonstrates extremely higher sales

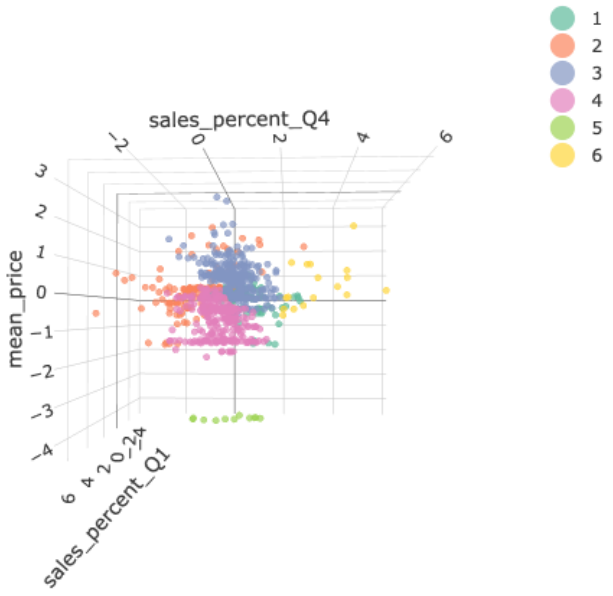

Figure 17: Hierarchical clustering: Clustering Factors 3D Graph

Table 1: Hierarchical clustering: Number of Products per Cluster

Cluster Number	Size
1	135
2	121
3	277
4	262
5	11
6	17

at the end of the year compared to its usual sales for the whole year.

(2) GMM Clustering

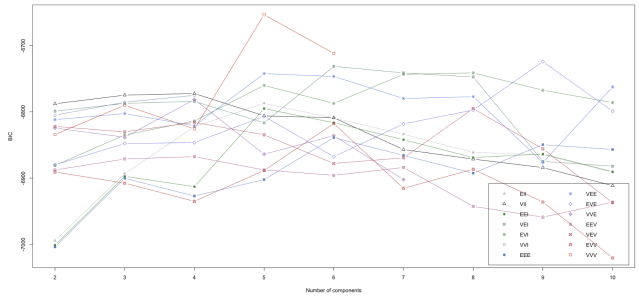

Figure 18: GMM: Bayes Information Criterion (BIC) based model family selection plot

Table 2: GMM: Number of Products per Cluster

Cluster Number	Size
1	510
2	52
3	126
4	87
5	48

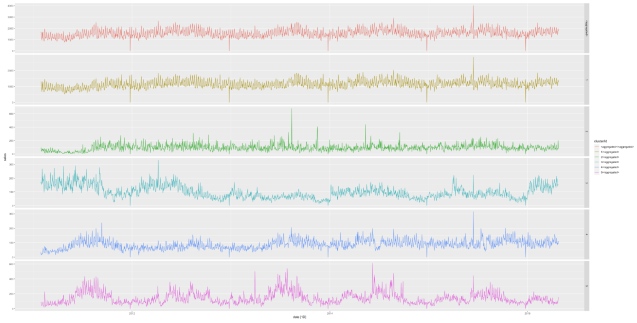

Figure 19: Hierarchical Time series after GMM clustering with 5 clusters: The first red time series is the original aggregated time series without clustering, the subsequent 5 time series corresponding to the group time series for 5 clusters

GMM clustering [16] has chosen VVV (Variable Volumn, Shape and Orientation) as its model family and 5 as its cluster number (Figure 9). Table 2 shows the distribution of cluster frequency based on GMM and each cluster has more equivalent size compared to hierarchical clustering.

From Figure 10, we also observe some different seasonality segmentation. The blue time series (cluster 3) demonstrate higher sales in spring and autumn and lower sales during winter. The purple time series (cluster 4) has relatively higher sales during autumn. The pink time series (cluster 5) has higher sales during spring and autumn compared to its spring and winter counterparts.

Both clustering methods have different interesting segmentation and will be tested during the forecasting experiments of different models to see whether they provide effective segmentation

Cluster-Based EDA

Before doing experiments, we examine whether some calendar/events factors have different impacts on different clusters. The sales represent the average sales quantity per factor for each cluster.

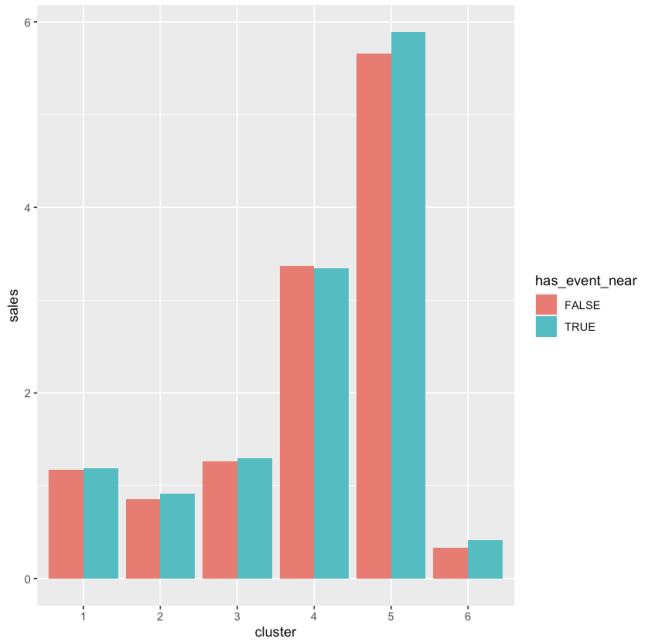

Figure 20: has_event_near factor

From Figure 20, we could see that cluster 5 is probably more sensitive to has_event_near factor compared to cluster 4, but they do have really small differences.

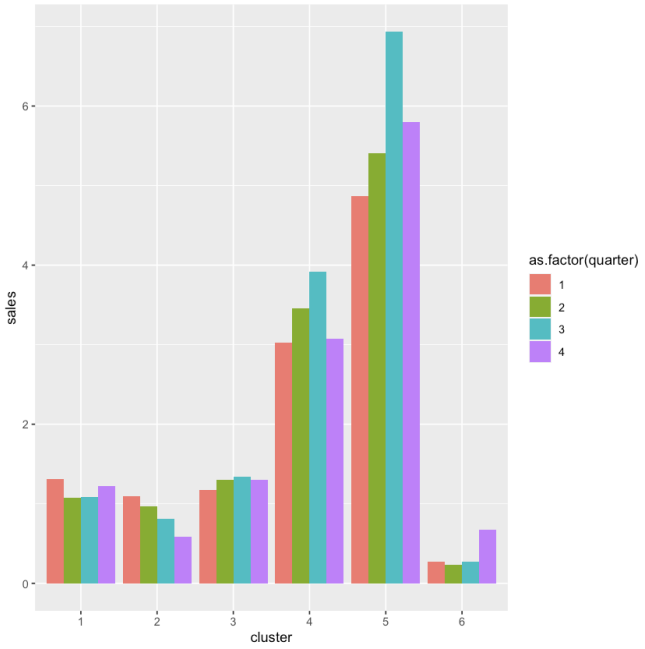

Figure 21: quarter factor

From Figure 21, we could see that cluster 1 and 3 are not that sensitive to quarter factor compared to other clusters.

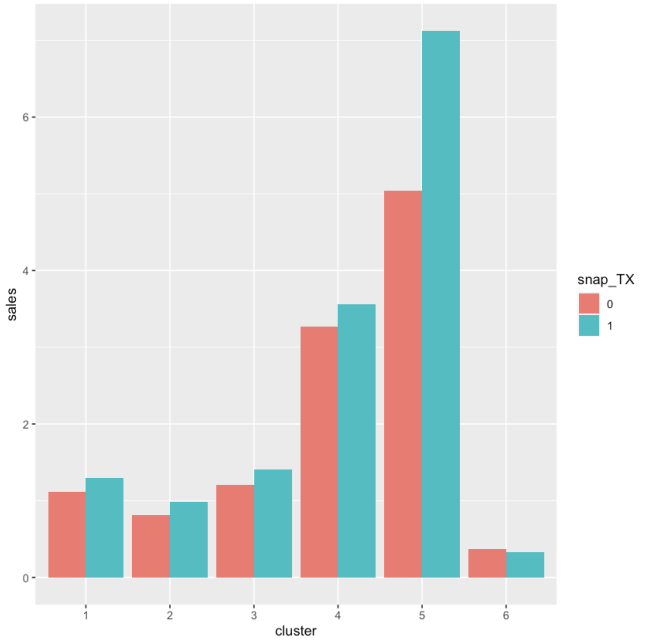

Figure 22: snap_TX factor

From Figure 22, we could see that cluster 5 is very sensitive to snap_TX factor compared to other clusters.

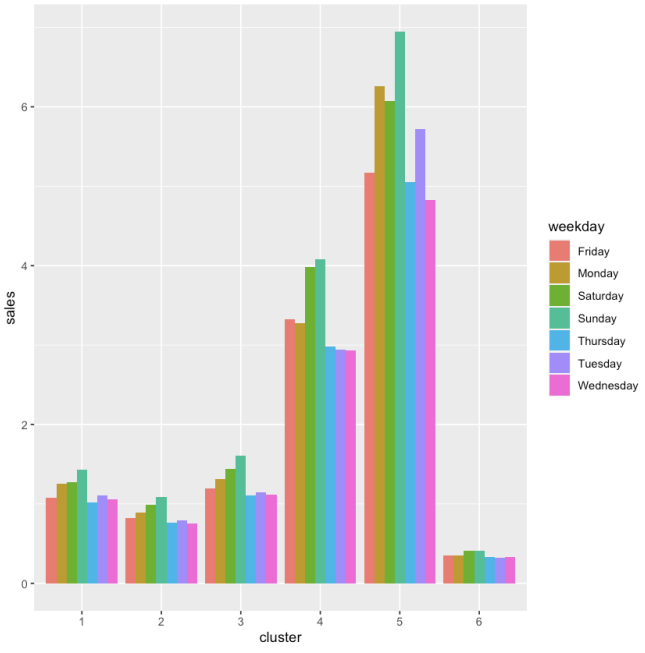

Figure 23: weekday factor

From Figure 23, we could see that cluster 5 is very sensitive to the weekday factor compared to other clusters.

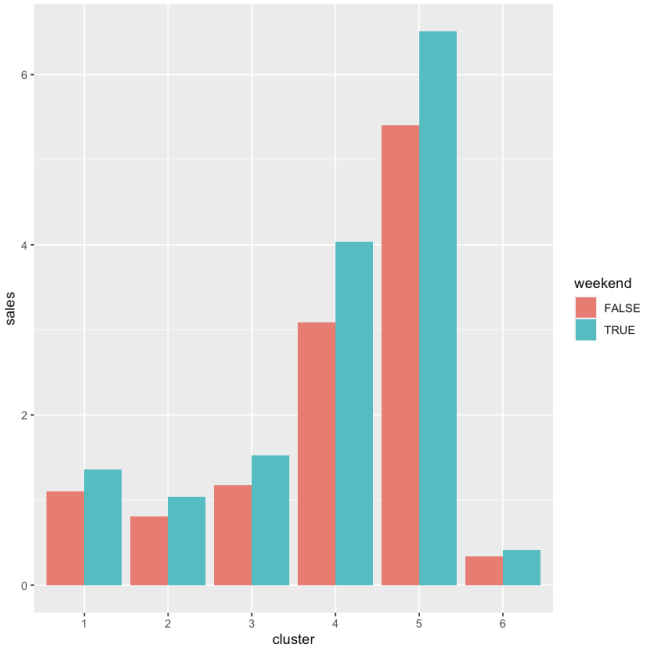

Figure 24: weekend factor

From Figure 24, we could see that all clusters similarly reacted to weekend factor.

3 RESULTS

Results with fixed regressors

In table 3, we listed all the models that we have experiments with their RMSE. The best result is 1.807. A step-forward method has been used to see which predictor or which combinations work better. The predictors used have been mention in section 2, and their final effect can be seen below:

- **fourier**: Adding a fourier predictor in the ARIMA models, seemed to improve the results. The most effective one, was with K=1. When we used K=2 or K=3, we got worse results, as they might have overfitted the training data.
- **snap_TX**: It improved the RMSE for every combination it was added on. It makes sense, as is a crucial factor on weather people buy products or not.
- **has_event_near**: When it was added to models, it generally showed a negative effect. But when it was added with predictors like *weekends or has_two_events* it showed improving the model.
- **has_two_events**: Again, when this one was added, it generally showed worse results. When added with as a combination with *has_event_near*
- **weekend**: Showed increasing results when added. With a combination with the quarter dummy though, the accuracy was decreasing.
- **month dummy**: This dummy predictor seemed the most promising one. It was not tested properly though, as it was increasing the calculation time by multiple times.
- **quarter dummy**: Overall, it showed to be adding noise to the model and decreasing the results. In practice it improved the results in some models where they had specific predictors, and helped the model capture the seasonality better.

In the following table, the experiments can be seen. A group of predictors that had very good results and was commonly used, was: *"snap_TX + has_two_events + has_event_near + quarter_2 + quarter_3 + quarter_4"*. It is mentioned as *"Group 1"* in order to save space.

Table 3: Models and RMSEs

Model	Predictors	RMSE
Random Walk + Drift	-	2.35
Naive	-	2.34
Seasonal Naive	-	2.32
Mean	-	2.11
TSLM	trend + season + Group 1	2.02
Prophet	-	2.005
Prophet	day=1 week=1 year=1, Group 1	1.99
Prophet	day=2 week=7 year=1, snap_TX	1.984
Prophet	Group 1	1.973
Prophet	day=10 week=5 year=3, Group 1	1.959
ARIMA	snap_TX + month_dummy	1.868
ARIMA	snap_TX + quarter_dummy	1.824
ARIMA	Group 1	1.82
ARIMA	fourier(K=2)	1.82
ARIMA	snap_TX + has_two_events	1.82
ARIMA	snap_TX + has_event_near	1.82
ARIMA	-	1.82
ARIMA	fourier(K=1)	1.819
ARIMA	snap_TX	1.819
ARIMA	snap_TX + weekend	1.815
ARIMA	fourier(K=1) + snap_TX + weekend	1.812
ARIMA	fourier(K=1) + snap_TX + weekend + has_event_near + quarter_dummy	1.811
ARIMA	fourier(K=1) + snap_TX + weekend + has_event_near	1.809
ARIMA	fourier(K=1) + Group 1	**1.807**

Results with cluster-based regressors

Following the fixed regressors model, we try to find whether some predictors have a different influence on different product clusters. Particularly we focus on the quarter dummy variables due to time and computation limitations. We use

fourier(K=1) + snap_TX + weekend as the baseline model and compare the results with/without quarter dummy regressors (one-hot encoded).

Table 4: Quarter Dummy Influence on Different Hierarchical Clusters

Cluster	Predictors	RMSE
1	fourier(K=1) + snap_TX + weekend	1.489
1	fourier(K=1) + snap_TX + weekend + quarter	1.483
2	fourier(K=1) + snap_TX + weekend	1.455
2	fourier(K=1) + snap_TX + weekend + quarter	1.456
3	fourier(K=1) + snap_TX + weekend	1.458
3	fourier(K=1) + snap_TX + weekend + quarter	1.460
4	fourier(K=1) + snap_TX + weekend	2.305
4	fourier(K=1) + snap_TX + weekend + quarter	2.307
5	fourier(K=1) + snap_TX + weekend	8.298
5	fourier(K=1) + snap_TX + weekend + quarter	8.279
6	fourier(K=1) + snap_TX + weekend	0.883
6	fourier(K=1) + snap_TX + weekend + quarter	0.878

Table 4 summarized the experiment's results. We can see that quarter dummy regressor indeed has different effects on different clusters. Quarter dummy has positive effects on cluster 1,5,6 while it has negative effects on cluster 2,3,4, indicating that different regressors could indeed have different impacts on a different cluster. It is worth noting that cluster 1,2,3,6 demonstrates much lower RMSE result than cluster 4 and 5, which might be due to the fact that these two clusters have not well generalized the products in them. We would discuss the reasons later in discussion part.

In practice, we would prefer fewer regressors when the RMSE shows a similar result. Hence, compared to the best predictors that we found which lead to 1.807, we prefer Auto ARIMA with fourier(K=1), snap_TX, weekends, has_event_near that not only use fewer predictors but also computationally less demanding.

4 DISCUSSION

In general, we find that calendar/events regressors could improve the overall accuracy of predictions. We also find that by using unsupervised clustering, we could come up with groups of products that share similar characteristics in terms of seasonality and sensibility towards different predictors. Our experiment results show that the quarter dummy predictor indeed had a different influence on the accuracy.

Particularly, we found cluster 4 and cluster 5 have really high RMSE compared to other clusters, indicating that the model for these two clusters did not generalize well the characteristics. Since we only use 3 features in clustering which might have underfitted for these two clusters, more complex clustering could be useful here by using more features to further separate these clusters into sub-clusters in order to decrease model errors. If we had more features about the products apart from sales and price that would be also helpful in distinguishing products into different clusters.

Due to time and computational constraints, we did not finish fitting different clusters by using different regressors and come up with a final result as further analysis is still required to investigate more different clusters with different regressors.

It would also be beneficial to experiment with other models of different natures such as the ensemble boosting machine learning method or deep learning method to compare the results and whether clustering would be beneficial to these different methods.

Moreover, we also need to point out that as the test data only cover one month between April and May, the performance of the model could be diminished when we come to different seasons.

5 CONCLUSIONS

In conclusion, we indeed find that supermarket product sales have really strong seasonality and differ a lot from each other in their trends, seasonality and sensibility to different regressors. The introduction of extra intervention variables such as calendars and events is of great importance in order to catch the variations in its forecasting.

Different models and auto ARIMA have experimented with different combinations of predictors. Of all the models, auto SARIMA behaves the best in catching the subtle trends and differences in seasonality.

On top of that, in order to catch the individual differences between products, we also introduced unsupervised clustering so that we could apply different regressors to different clusters so as to reduce the RMSE errors accordingly. The features that are used for clustering could of great importance in the generalization of similarities between products. Further sub-clustering could also be helpful in further distinguishing the products in order to have more precise predictors for each sub-cluster.

REFERENCES

[1] Spyros Makridakis vangelis Addison Howard, inversion. 2020. M5 Forecasting - Accuracy. https://kaggle.com/competitions/m5-forecasting-accuracy

[2] Gianluca Bontempi, Souhaib Ben Taieb, and Yann-Aël Le Borgne. 2013. *Machine Learning Strategies for Time Series Forecasting*. Springer Berlin Heidelberg, Berlin, Heidelberg, 62–77. https://doi.org/10.1007/978-3-642-36318-4_3

[3] Rb Cleveland, William S. Cleveland, Jean E. McRae, and Irma J. Terpenning. 1990. STL: A seasonal-trend decomposition procedure based on loess (with discussion).

[4] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[5] J J Hopfield. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* 79, 8 (1982), 2554–2558. https://doi.org/10.1073/pnas.79.8.2554 arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.79.8.2554

[6] Stephen C Johnson. 1967. Hierarchical clustering schemes. *Psychometrika* 32, 3 (1967), 241–254.

[7] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3149–3157.

[8] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980

[9] Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic Gradient Descent with Warm Restarts. https://doi.org/10.48550/ARXIV.1608.03983

[10] S. Makridakis, A. Andersen, R. Carbone, R. Fildes, M. Hibon, R. Lewandowski, J. Newton, E. Parzen, and R. Winkler. 1982. The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting* 1, 2 (1982), 111–153. https://doi.org/10.1002/for.3980010202 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.3980010202

[11] Spyros Makridakis, Chris Chatfield, Michèle Hibon, Michael Lawrence, Terence Mills, Keith Ord, and LeRoy F. Simmons. 1993. The M2-competition: A real-time judgmentally based forecasting study. *International Journal of Forecasting* 9, 1 (1993), 5–22. https://doi.org/10.1016/0169-2070(93)90044-N

[12] Spyros Makridakis and Michèle Hibon. 2000. The M3-Competition: results, conclusions and implications. *International Journal of Forecasting* 16, 4 (2000), 451–476. https://doi.org/10.1016/S0169-2070(00)00057-1 The M3- Competition.

[13] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2020. The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting* 36, 1 (2020), 54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014 M4 Competition.

[14] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2022. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting* 38, 4 (2022), 1346–1364. https://doi.org/10.1016/j.ijforecast.2021.11.013 Special Issue: M5 competition.

[15] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. https://doi.org/10.48550/ARXIV.1905.10437

[16] Douglas A Reynolds. 2009. Gaussian mixture models. *Encyclopedia of biometrics* 741, 659-663 (2009).

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 56 (2014), 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[18] Wikipedia contributors. 2022. Dunn index — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Dunn_index&oldid=1124562963 [Online; accessed 23-December-2022].

[19] Wikipedia contributors. 2022. Silhouette (clustering) — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Silhouette_(clustering)&oldid=1126468040 [Online; accessed 23-December-2022].

[20] Gang Zhou, Yafeng Wu, Ting Yan, Tian He, Chengdu Huang, John A. Stankovic, and Tarek F. Abdelzaher. 2010. A multifrequency MAC specially designed for wireless sensor network applications. *ACM Trans. Embed. Comput. Syst.* 9, 4, Article 39 (April 2010), 41 pages. https://doi.org/10.1145/1721695.1721705