Sales Time Series Forecasting
Team 23

Spyros Avlonitis, Dragos Pop, Teun Zwier and Hetian Chen
Applied Forecasting in Complex Systems 2021
University of Amsterdam

ABSTRACT

In this paper, we present the research we did to participate in the "Sales Time Series Forecasting" Kaggle competition. Our goal was to forecast the daily sales for 28 days in the future of 823 products sold in the Walmart TX3 store in Texas, US. First, we present the insights we got from the data exploration, after which we discuss the statistical methods we experimented with to find the best forecasting model. Namely, we used a mean, a naive and a seasonal naive model as a baseline and then experimented with Exponential Smoothing, Arima, TSLM, Dynamic Regression, Croston, Prophet, NNETAR and Bagging. We found that combination models of ARIMA, TSLM and Dynamic regression resulted in the best forecasts. Additionally, even more sophisticated models could not produce significantly better forecasts compared to the simpler ones, and we hypothesize some reasons for this.

1 INTRODUCTION

Last year, the M5 Kaggle Forecasting competition was held, a data science competition to find the most accurate forecast for item level sales in Walmart stores [5]. The goal of the competition was to best forecast item level sales based on a dataset of past item-level sales over multiple departments over multiple Walmart stores in the United States.

Accurately predicting sales for specific products is an important problem for any retail business, especially large-scale chains. Without trying to look into the future, it is difficult to determine how large or small the stock allocation in a given store should be. A common way to try and solve this is by forecasting; utilizing past and present data to predict future sales [4, 10]. Forecasting can help businesses in various ways, such as capacity planning and predicting exception cases. Because of this, much focus is put on optimizing forecasting results.

Forecasting sales brings with it various issues [3]. Future data always has a measure of uncertainty with it, as many outside influences can unexpectedly play a role in product sales (e.g. most recently the pandemic). Specifically for a large chain like Walmart, differences in sales can occur between products, between departments, and between different stores that all have to be taken into account.

For this report, the aim was to forecast on a smaller scale with a subset of the M5 dataset, the dataset used in the Kaggle competition. Instead of multiple departments and stores, only data from the food department in one Texas branch was used. In this way, the forecasting was location and product-specific.

Research was executed by using exploratory analysis, implementing various forecasting algorithms based on this analysis and comparing the results. The main goal of the research was to investigate to what extent future Texan food sales data can be predicted based on historical sales data and which method is the best to predict such data. Next, the data is described further, after which the exploratory analysis and forecasting methods will be presented.

2 DATASET

The dataset available consisted of 4 files;

- sales_train_validation_afcs2021.csv
- sales_test_validation_afcs2021.csv
- calendar_afcs2021.csv
- sell_prices_afcs2021.csv

For specific column-wise descriptions, see the appendix. The sales_train_validation and sales_test_validation files consisted of historical daily sales for 823 specific food products in the Texan WalMart (TX3) between 2011-01-29 and 2016-06-19 (1941 days). Here, the test set consisted of the last 28 days and the train set of the remainder.

The other two files gave additional information pertaining to prices and dates. sell_prices_afcs2021.csv consisted of weekly prices for all 823 products. calendar_afcs2021.csv included information about the date in the sales dataset for different levels of time(day, week, month and year). Additionally, it gave information on special events (e.g. Halloween, Thanksgiving) and if SNAP purchases (customer benefits) were allowed on a specific day.

For effective forecasting, these files were combined into one training dataset and one test dataset. For this, the calendar file was used to structure the data by daily index. The daily sales data was transposed to columns for every food product and additional events and SNAP information were added. Prices were interpolated from weekly to daily format, so every food product also had a daily price tag.

Data Exploration and Decomposition Analysis

During the exploratory data analysis, the first step to be taken was inspecting the missing values from the given datasets. Accordingly, it was discovered that only the event variables, specifically the name and type, from the calendar data had missing values. This indicates a good quality of the data which does not require additional data imputation techniques that would have additional introduced uncertainty in the forecast.

Secondly, the large proportion of zero values in the units sold within the training data was remarked and calculated at 62%. Starting with the assumption that the products which never sold will never generate a sale in the future, these products were sought with the purpose of being eliminated from the forecast. However, each of the 823 products had at least one unit sold.

Then, the evolution of the units sold over time was visualised. As one can see from the graph bellow, the sales of each product vary significantly from one product to another and there are a total of 823 of such series, which makes visually analysing each very demanding. Hence, we decided to aggregate the sales of all products per day and plot the result.

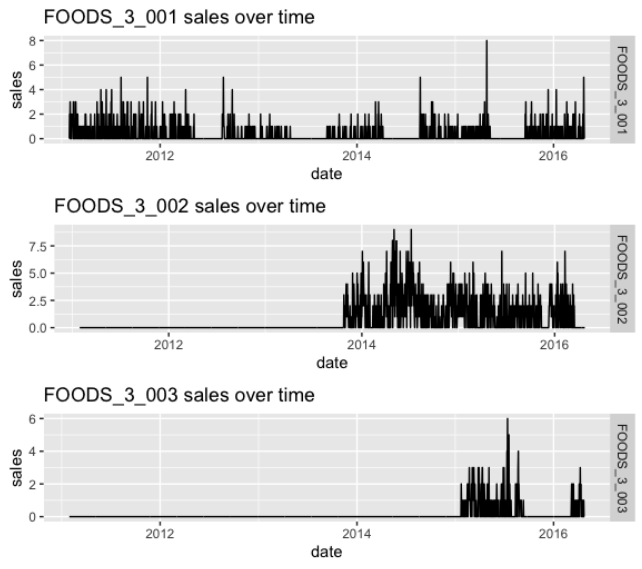

Figure 1: Units sold over time per product

In Figure 2, the yearly seasonality of the sales stands out. The number of sales seems to also follow a slightly upper trend and another interesting annual pattern happens at the end year that causes the sales to drop drastically for a day. This was further inspected and it was discovered that the particular day is the first Christmas day. Therefore, one possible explanation for the lack of sales would be that the store is closed.

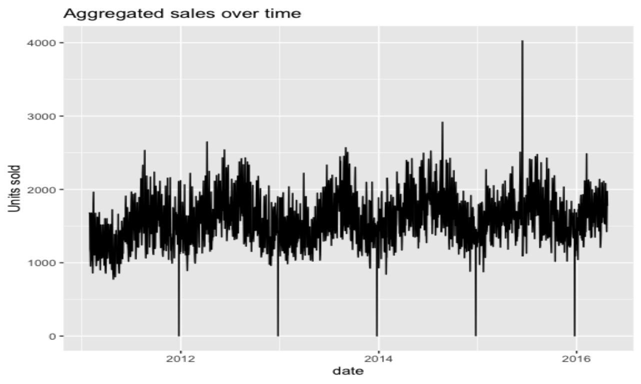

Figure 2: Aggregated daily units sold over time

To get a better insight into the seasonality of the aggregated daily sales, the series was decomposed using the STL method because it has several advantages over other decomposition techniques, including its robustness to outliers and allowing the seasonal component to change over time [8]. Accordingly, in the graphs from Figure 3, it is visible that the number of sales per day has an upward trend, a strong yearly seasonality, and a weaker monthly seasonality which changes over time such that two consecutive months resemble each other, but two that are far apart may have different seasonal patterns.

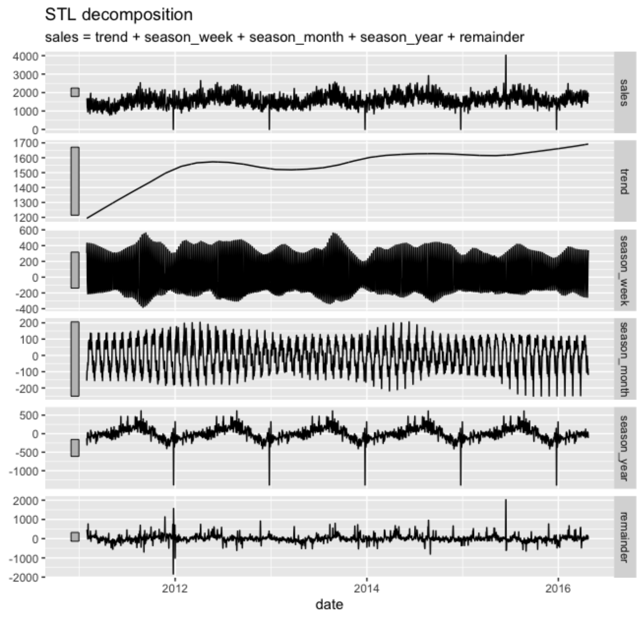

Figure 3: STL Decomposition

Additionally, it was seen that the number of sales during the weekend is larger than during the week days (see Figure 4).

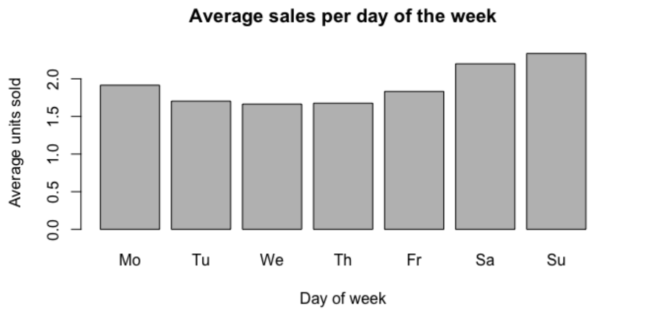

Figure 4: Average units sold per day of the week

Although in Figure 2 we saw that the number of total number of units sold per day follows a slightly upwards trend, Figure 5 shows that the average number of sales per product is declining every year. This implies that other products are introduced over time and their sales cannibalize the old ones.

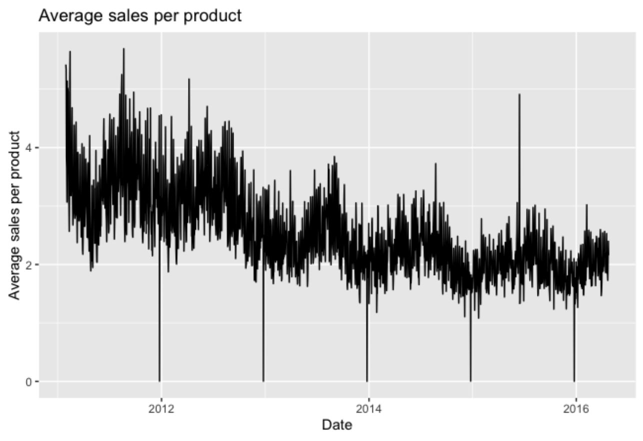

Figure 5: Average sales per product

Afterwards, the selling prices of the products were analysed. More exactly, they were averaged per day and plotted over time, as illustrated by Figure 6). From this image, one can see the average selling price of a product ranges from approximately \$2.75 to upwards of \$2.9. At the beginning, the average rises drastically until the start of 2012, followed by a flat period until 2014, and ending with an ascending trend. This suggests the inflation over the food products in this period was not constant.

Finally, we checked the correlation between the sales and the predictors we used for our forecasting. As can be sees

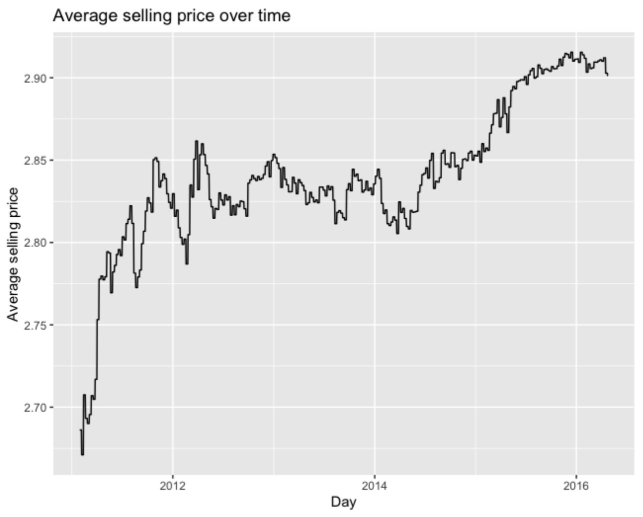

Figure 6: Average selling price over time

from Figure 12 there is a small negative correlation between the sale price and the sales. On the other hand, there is a small but positive correlation between sales and discount. The same stands for weekends and snap days. Finally, we can see that there are less sales during the event days and more sales during the weekends.

Feature Engineering

Considering the previously discovered insights, extra features were derived and added to the data with the scope of improving the forecast. An additional dummy variable, depicting whether the day is part of the weekend or not, was created. This was followed by two binary features that were computed with regard to the events, the first one denoting if there is an event in the respective day and the second one considering if there is an event in the day before or the one after. Moreover, the event types variables were one-hot-encoded to allow modeling with methods that do not have a functionality to process categorical data.

The last variable to be derived is called discount and is defined as the proportion of the selling price of the current week to the average of prices for the last four weeks. Finally, we also used the selling price and snap_TX features, which was present in the dataset, for our forecasts. snap_TX indicates if the day was part of the SNAP[1] program.

3 STATISTICAL METHODS

To find the best forecasting method, we experimented with multiple methods which theoretically can provide us with quality results. First, we started with simple methods such

[1]https://www.fns.usda.gov/snap/supplemental-nutrition-assistance-program

as Mean and Naive in order to create a baseline. Below we discuss the methods we utilized.

Models

Mean/Naive/Seasonal Naive/Drift. To forecast the sales in the 28 days of the prediction period, we naturally start with the four benchmark methods introduced in the lectures and the textbook, namely *Mean, Naive, Seasonal Naive* and *Drift* methods. We start with these benchmark methods for two reasons: firstly the sales time series of individual products show various characteristics, some with a trend or strong seasonality while some has no clear pattern. These benchmark methods should be able to deal with different characteristics. Secondly these widely used forecast methods could also be used as benchmarks to other more sophisticated methods. Here we fit the four methods to the training data and calculate the error metrics, namely *MAE* and *RMSE*, with the 28-day data of the testing set.

Exponential Smoothing. As sales for individual products might be correlated in a certain period of time, we could use moving averages to forecast future sales based on past data. The most common method for moving averages would be Exponential Smoothing, as the weights decay exponentially for older observations. There are various models for Exponential Smoothing, either with trend or with seasonality or with both, and either an additive or a multiplicative model. In the project, we simply apply *ETS()* function in R, as the algorithm would automatically select the best model by AICc values. By this means, we could fit different models to sales of different products with different patterns. To compare the fitted ETS models with those from other methods, we then calculate the *MAE* and *RMSE* with the 28-day data of the testing set.

ARIMA. For similar reasons that we use Exponential Smoothing, we could also fit different *ARIMA* models to sales time series of each product. The *ARIMA()* function in R would automatically select the best model by AICc values.

Linear Time Series Regrssion Model (TSLM). In the training dataset, apart from the date and sales data, there are other information available, such as the unit prices of the products, day of the week of the sales days, whether the sales day was a holiday, the type of holiday (religious/cultural/sporting event/national) and whether *Supplemental Nutrition Assistance Program (SNAP)* benefits were available on that day. With these inputs, it is possible to fit a time series regression model using *TSLM()* function to the training dataset.

Before we fit the model, we need to transform categorical features into binary dummy variables. We use one dummy variable for weekdays/weekend and one for each of the event types (religious/cultural/sporting event/national). The unit prices are numerical values and the *snap_TX* values are already binary so we do not need to transform them.

It is also common sense that grocery stores usually open for shorter time or even close on holidays such as Christmas. And customers tend to shop before the holidays. So there could be lagged impact of predictors such as religious and cultural holidays. We would also use the *lag()* function for these predictors in the regression models. As the unit prices and the calendar for the 28-day forecast period are provided, we could simply use the fitted *TSLM* model to forecast the sales. Then we could calculate the *MAE* and *RMSE* values.

Dynamic Regression. After fitting linear regression models to the training data, dynamic regression models were considered as well, as the error terms might not be white noise. In this project, we use *ARIMA* errors to fit dynamic regression models, with the same (dummy) variables we use for linear regression models.

Croston. All methods discussed in this paper assume that the data have a continuous sample space. However, in our case the dataset consist of discrete number of orders. Additionally, our prediction need to be on product level, and thus we have to predict very small numbers. Through our analysis, we found that the mean number of times a product is sold per day is 5.69. A method which is used for such cases is the "*Croston*'s method," named after its British inventor, John Croston, and first described in [2].

With *Croston*'s method, we construct two new series from our original time series by noting which time periods contain zero values, and which periods contain nonzero values. Then it is using simple exponential smoothing forecasts on the two new series. This method will not always fulfill our requirements sufficiently because a product's sales can have multiple periods of zero and non-zero sales.

Prophet. A recent proposal is the *Prophet* model. This model was introduced by Facebook [11], originally for forecasting daily data with weekly and yearly seasonality plus holiday effects. It was later extended to cover more types of seasonal data. It works best with time series that have strong seasonality and several seasons of historical data. Prophet can be considered a nonlinear regression model. We decided to try this model because it takes into account 1) a piecewise linear trend, which could be useful to detect periods with zero sales 2) seasonality and 3) holiday effects, which seem to be present in our series.

Neural Network. Another possible approach to forecasting is the utilization of Neural Networks [6] . A Neural network is a network of layers of nodes that train weights for these nodes based on the difference between generated values and actual data/values. In the case of Time Series forecasting, there are multiple advantages to Neural Networks compared to other models. First, the model is more robust to noise compared to other models. Second, it does not assume a linear relationship between input and output, which means that it allows for nonlinear relationships between the response variable and predictors. This could be especially helpful for predicting product sales, as there are so many factors that play a role in purchasing products.

Specifically, for this experiment, the *NNETAR* model was used. This Neural Network is a feed-forward multilayer perceptron, meaning the network contains a layer of input nodes, at least 1 hidden layer and output nodes. Feedforward refers to the fact that information travels only forward from input to output nodes. In the figure below, a small-scale Feedfoward NN is shown.

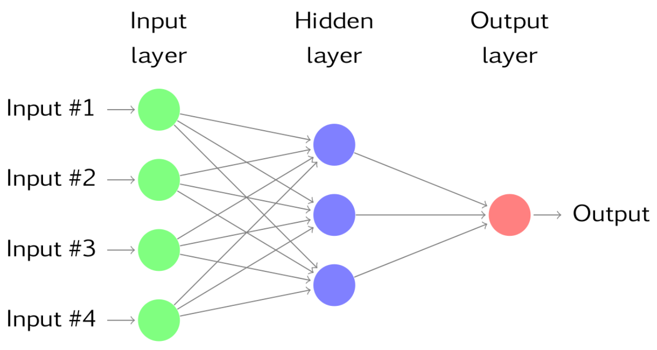

Figure 7: Small-scale Feedforward Neural Network (https://otexts.com/fpp2/nnetar.html).

The *NNETAR* model consists of 1 hidden layer (P=1) and uses lagged training data as input for the model. Depending on the number repeats N number of networks are instantiated and trained, of which the average is taken for the final model. To generate a forecast, the times variable is specified as the number of t forecasts that are generated. Forecast values are generated from the network, but to get a reliable average, t has to be sufficiently high. Here, repeats was set to 20 (which is standard) and times to 30 (to make the method computationally viable). p (and therefore also k) were kept at default values to be automatically decided based on AIC.

Bagging. One way to improve forecast accuracy is the method of bagging [7]. Bagging is a shortened version of **B**ootstrapped **Agg**regat**ing**. The idea behind bagging is to generate multiple slightly different time series, forecast using these time series and take the average of these forecasts. Accordingly, by changing the series slightly and running models on slightly different training sets, the eventual forecasts are less influenced by outliers in the training data.

More specifically, we utilized bagging for exponential smoothing forecasts, which consisted of 3 steps. First, the time series was decomposed into trend, seasonal components and the remainder. Then, slightly different time series were bootstrapped using the remainder data. Here, it was decided to generate 10 time series and use bootstrap blocks of size 7 (to capture a week).

Lastly, exponential smoothing was executed as explained in the previous section. After forecasts were obtained, the means were aggregated resulting in a combined forecast of 10 models.

Model Combinations. Finally, combining models is a possible way of improving forecasting [1, 9]. Different models may capture different aspects of the forecast more accurately, so computing the average can combine the best aspects of the various forecasting methods.

The combination model were mostly focused on less computationally intensive models, as multiple fits had to be calculated to create a combination model. Multiple combinations were executed between *ETS*, *ARIMA* and *TSLM* models with various different permutations (different additional variables, fourier seasonality etc.). The most focus was put on the following combinations: *ETS* & *ARIMA*, Multiple *TSLM* models, Dynamic Regression & *TSLM*.

Metrics

To evaluate our models, we used the test dataset provided. Given that we only experimented with series with the same units of measure, we used scale-dependent errors. Namely, MAE and RMSE.

Mean absolute error (MAE). When comparing forecast methods applied to a single time series, or to several time series with the same units, the MAE is popular as it is easy to both understand and compute.

Root mean square error (RMSE). Another metric which is useful when comparing methods on the same units, as we do, is RMSE. Compared to MAE, RMSE is more difficult to interpret, however it is often used because minimizing it is pushing forecasts toward the mean; compared to MAE which is leading them towards the median.

4 RESULTS

Table 6 shows the *MAE* and *RMSE* scores of the statistical methods evaluated on the test dataset. For each method, we choose the best model with the lowest *RMSE* score.

Figures 8, 9, 10 and 11 also shows the actual sales of item *FOODS_3_370* against the forecast from each method. First, we gain an intuition of the performance of every method from the plots. We could clearly see that *Mean*, *Naive* and *Drift* perform poorly for items with strong seasonality such as *FOODS_3_370*. This also applies to the forecasts from *Croston* method, which is a level line. *ETS* and *ARIMA* perform slightly better but ignore most of the variation of the seasonality. Forecasts from *TSLM*, *Dynamic Regression*, *Prophet* and *NNETAR* look quite close to the observed sales values, with the latter two models provide higher forecasts values than the former two. *Seasonal Naive* captures the seasonality perfectly, but its performance on items without seasonality might not be as good. We further investigate the performance of each model by looking at the *MAE* and *RMSE* values calculated on the testing dataset.

As the *Kaggle* competition evaluates forecasts solely on *RMSE* score, we use the same metric for method comparison. As we could see on Table 6, the four benchmark methods all give *RMSE* scores over 3. Exponential smoothing also gives an *RMSE* score close to 3. *Prophet*, *NNETAR* and *Bagging* give relatively high scores. Given the complexity of these models, we do not choose them for the forecasts. *ARIMA* and *Croston* give good scores but these models use only the time values as input, so we prefer to use regression models, either linear/dynamic or a combination of both, for the forecast.
We further investigate which type of model and which type of predictors we should use for the forecast. To speed up the process, we randomly choose 300 products, use their sales data in the training dataset to fit the regression models, and then use the testing data for the same 300 products for validation. Results are shown in Table 2 and Table 3.

	MAE	RMSE
Mean	1.58	3.32
Naive	1.69	4.09
Seasonal Naive	1.55	3.52
Drift	1.70	4.12
Exponential Smoothing	1.35	2.98
ARIMA	1.26	2.68
Comb. of ETS & ARIMA	1.29	2.75
TSLM	1.50	3.06
Dynamic Regression	1.29	2.74
Comb. of TSLM & Dynamic Regression	1.35	2.76
Croston	1.33	2.79
Prophet	1.35	2.77
NNETAR	1.40	2.81
Bagging	1.35	2.81

Table 1: The measured results from the experiments we tried.

	AICc
TSLM	1166
TSLM w lagged predictors	1171
Dynamic Regression	4502
Dynamic Regression w lagged predictors	4587

Table 2: AICc scores of regression models on a random subset of data

	MAE	RMSE
TSLM	1.08	1.62
TSLM w lagged predictors	1.09	1.63
Dynamic Regression	0.97	1.54
Dynamic Regression w lagged predictors	1.06	1.65

Table 3: MAE and RMSE scores of regression models on a random subset of data

Dynamic Regression model is clearly the best model in terms of error metrics, although it has a higher AICc score than *TSLM*. However, *Dynamic Regression* using *ARIMA()* function in R sometimes generates big errors or even fails, probably due to low sales numbers for some products. To deal with this issue, we decide to use a combination of *TSLM* and *Dynamic Regression* as the final model. The lagged predictors, namely lagged*Religious/Cultural/National* holidays, do not improve the accuracy but make the model more complex (higher AICc scores). So we decide not to use any lagged predictor in our final model for forecast.

Example forecasts

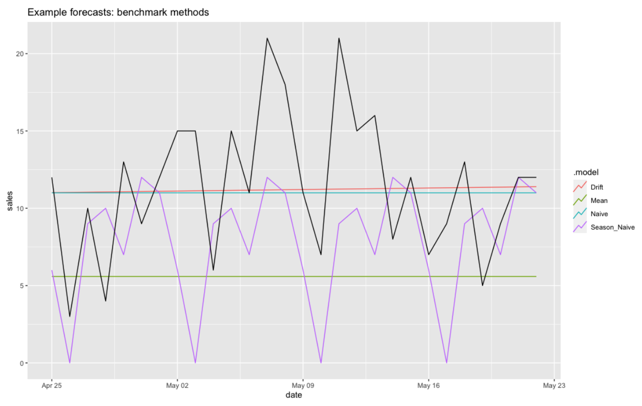

Figure 8: Example forecasts: four benchmark methods

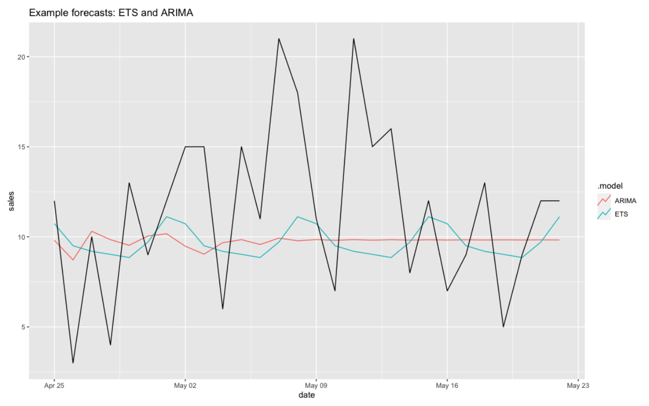

Figure 9: Example forecasts: ETS and ARIMA

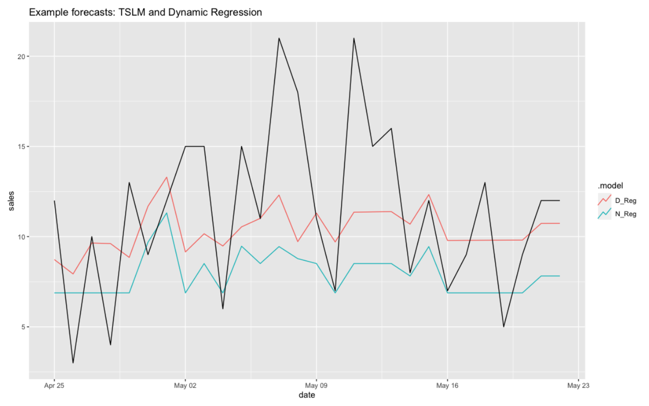

Figure 10: Example forecasts: Linear Regression (N_Reg) and Dynamic Regression (D_Reg)

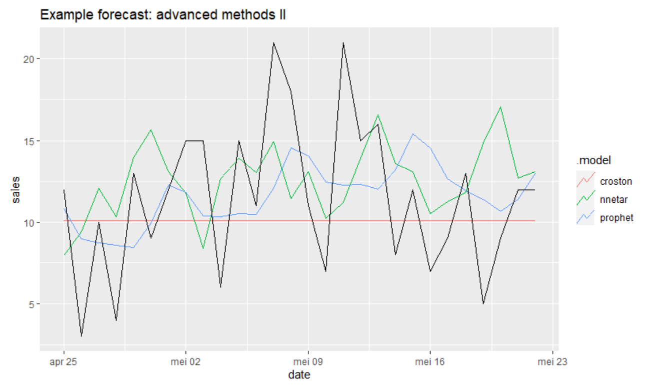

Figure 11: Example forecasts: Croston, NNETAR and prophet

5 DISCUSSION

Interesting findings

First, it was found that the baseline methods produce a significantly worse result than the other forecasting methods: *Mean* , *Naive Seasonal Naive* and *Drift* had the bottom 4 MAE and RMSE scores. While this is expected, it is still an important finding. The models are likely too simplistic for the quickly changing product level sales. While these methods can result in accurate forecasts for specific products, they perform worse overall.

Then, through our experiments, we found that the best model for forecast is a combination of *TSLM* and *Dynamic Regression* on predictors such as unit price, weekend, holidays, *SNAP_TX* program, etc. This might be due to the heavy impact of the predictors on the sales, or the simple time series patterns shown in the sales of individual products.

In theory, and given the nature of our dataset of counts, we would expect Croston method to outperform the rest of methods which are primarily used for continuous data. However, that is not the case.

More sophisticated, methods such as Prophet, NNETAR, Bagging are not providing significant improvements compared to Arima and Dynamic Regression. Likely, the Neural network models had difficulty with the sparsity of Product-level time series. That could be because the information the selected features contained was better utilized in models like *TSLM* and *ARIMA*.

Limitations

First, a limitation of the study is the dataset available. When predicting the sales of products, a lot of factors may play a role on the sales that are not present in our dataset. For example, the available stock of specific products, the category/subcategory, the location of the product inside a store, or the weather. As such, possibly more accurate predictions could be made with a more complete dataset or a different approach in feature engineering.
Second, there were computational limitations to the research. Due to the high computational complexity of some of the approaches, most notably NNAR and Bagged ETS models, compromises had to be made regarding the features number and models' size. Bigger neural networks or Bagged ETS with more resources may result in more accurate forecasts, which could be an interesting direction for future research.
Finally, we tried to find the best forecasting method which can fit all the products' sales series. Specific models may result in quality forecasts for one product, but underperform for other products. For example, neural networks could be suitable for high-selling products, but are less effective on sparse datasets, so low-selling products are harder to forecast effectively. An alternative approach for future research could be to look for the best fitted model for each product individually.

6 CONCLUSION

Concluding, the aim of this research was to find the best method and use it to generate products' sales forecasts. Through data exploration, the main characteristics of the data were obtained. Additionally, we extracted insights about the data and created features that will help us with forecasting. Next, we tested various statistical methods and compared them to each other. The results suggest that the most accurate methods for our dataset are TSLM, ARIMA and combination of the two. These models yielded significantly better results than naive methods and slightly better than the more sophisticated methods. Finally, we discussed the limitations of our approach and suggested promising alternatives for future research, such as product-specific models.

REFERENCES

[1] John M Bates and Clive WJ Granger. 1969. The combination of forecasts. *Journal of the Operational Research Society* 20, 4 (1969), 451–468.

[2] J. D. Croston. 1972. Forecasting and Stock Control for Intermittent Demands. *Operational Research Quarterly (1970-1977)* 23 (1972), 289–303. http://www.jstor.org/stable/3007885

[3] Anita S Harsoor and Anushree Patil. 2015. Forecast of sales of Walmart store using big data applications. *International Journal of Research in Engineering and Technology* 4 (2015), 51–59.

[4] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice.* OTexts.

[5] University of Nicosia. 2020. M5 Forecasting - Accuracy. 1 (2020). https://www.kaggle.com/c/m5-forecasting-accuracy/data Accessed: 2021-12-12.

[6] Zeydin Pala, İbrahim Halil Ünlük, and Erkan Yaldız. 2019. Forecasting of Electromagnetic Radiation Time Series: An Empirical Comparative Approach. *Applied Computational Electromagnetics Society Journal* 38 (2019).

[7] Fotios Petropoulos, Rob J Hyndman, and Christoph Bergmeir. 2018. Exploring the sources of uncertainty: Why does bagging for time series forecasting work? *European Journal of Operational Research* 268 (2018), 545–554.

[8] Cleveland Robert, C William, and Terpenning Irma. 1990. STL: A seasonal-trend decomposition procedure based on loess. *Journal of official statistics* 6, 1 (1990), 3–73.

[9] Thomas D Russell and Everett E Adam Jr. 1987. An empirical evaluation of alternative forecasting combinations. *Management Science* 33, 10 (1987), 1267–1276.

[10] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72 (2018), 37–45.

[11] Sean J. Taylor and Benjamin Letham. 2018. Forecasting at Scale. *The American Statistician* 72 (2018), 37–45. https://doi.org/10.1080/00031305.2017.1380080 arXiv:https://doi.org/10.1080/00031305.2017.1380080

A DATASET DESCRIPTION

File 1: calendar_afcs2021.csv: contains information about the dates the products are sold.

Label	Description
date	The date in a "y-m-d" format.
wm_yr_wk	The id of the week the date belongs to.
weekday	The type of the day (Saturday, Sunday, . . . , Friday).
wday	The id of the weekday, starting from Saturday.
month	The month of the date.
year	The year of the date.
event_name_1	If the date includes an event, the name of this event.
event_type_1	If the date includes an event, the type of this event.
event_name_2	If the date includes a second event, the name of this event.
event_type_2	If the date includes a second event, the type of this event.
snap_TX	A binary variable (0 or 1) indicating whether the stores of TX, allow SNAP3 purchases on the examined date. 1 indicates that SNAP purchases are allowed.

Table 4: Calender dataset

File 2: sell_prices_afcs2021.csv: Contains information about the price of the products sold per store and date.

Label	Description
store_id	The id of the store where the product is sold.
item_id	The id of the product.
wm_yr_wk	The id of the week.
sell_price	The price of the product for the given week/store. The price is provided per week (average across seven days).If not available, this means that the product was not sold during the examined week. Note that although prices are constant at weekly basis, they may change through time (both training and test set).

Table 5: Prices dataset

File 3 & 4: sales_train_validation_afcs2021.csv and sales_test_validation_afcs2021.csv: contains the historical daily unit sales data per product and store.

Label	Description
item_id	The id of the product.
dept_id	The id of the department the product belongs to.
cat_id	The id of the category the product belongs to.
store_id	The id of the store where the product is sold.
state_id	The State where the store is located.
d_1, d_2, . . . , d_i, . . . d_1913	The number of units sold at day i, starting from 2011-01-29 (train).
d_1914, d_1925, . . . , d_i, . . . d_1941	The number of units sold at day i (test).

Table 6: Product dataset (train and test

B PREDICTORS' CORRELATION

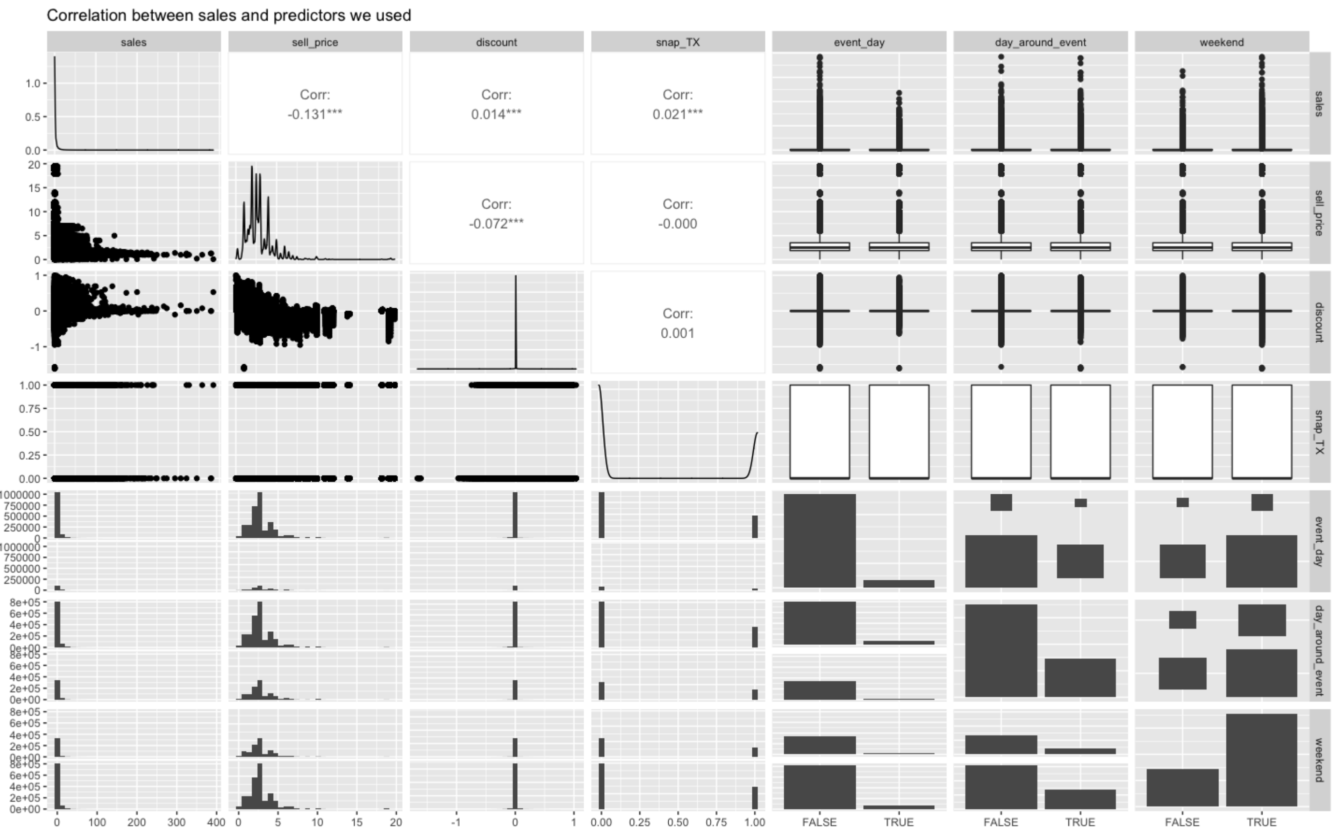

Figure 12: Correlation Table