The Alan Turing Institute

Data Study Group Final Report: CatsAi

31 Aug – 25 Sep 2020

Communicating high-street bakery sales predictions using counterfactual explanations

https://doi.org/10.5281/zenodo.5562660

Contents

1 Executive summary **3**
- 1.1 Challenge Overview . . . 3
- 1.2 Data Overview . . . 3
- 1.3 Main Objectives . . . 3
- 1.4 Approach . . . 3
- 1.5 Main Conclusions . . . 4
- 1.6 Limitations . . . 5
- 1.7 Recommendations and Future Work . . . 5

2 Introduction **6**
- 2.1 Challenge Overview . . . 6
- 2.2 Research Questions . . . 6
- 2.3 Predictive Models using Meteorological Data . . . 7
- 2.4 Interpretable Models . . . 7

3 Exploring Explainable AI **8**
- 3.1 Explainable AI: some recent progress . . . 8
- 3.2 Issues with some state of the art methods . . . 9
- 3.3 Beyond correlation based explanations towards counterfactuals . . . 11

4 Data overview **12**
- 4.1 Dataset Summary . . . 12
- 4.2 Data Description . . . 12
- 4.3 Data Quality Issues . . . 12

5 Exploratory Data Analysis **14**
- 5.1 Data Wrangling . . . 14
- 5.2 Data Analysis . . . 15

6 Experiments **26**
- 6.1 Pre-processing Steps . . . 26
- 6.2 Performance Metrics . . . 26
- 6.3 Predictive Models . . . 27
- 6.4 Explanation Methods . . . 38

7 Future Work and Research Avenues **45**

8 Team Members **48**

9 Appendix **50**
9.1 Glossary 50
9.2 Detailed Descriptions for pre-processing steps 54
9.3 Features for building Random Forest Models 54
9.4 Features for Gradient Boosting Regressor (one-hot encoded) 54
9.5 Features for Gradient Boosting Regressor (label encoded) . 55
9.6 Features used for Gradient Boosting Model in R 55
9.7 Features for MLP 55
9.8 Structured timeseries model components weights 55

References **58**

1 Executive summary

1.1 Challenge Overview

This challenge aims to help CatsAi better serve their client (a large wholesaler) to estimate bakery orders to reduce waste and under delivery. The main tasks were to predict high-street sales based on meteorological factors and apply explainability techniques to effectively communicate their outputs to the client.

1.2 Data Overview

During the challenge, we explored data relating to sales for a key client operating in a single country. The data comprised four different sections: location, products, weather and product sales, our target variables. Each group of variables provided several details about particular weather conditions or location (maximum temperature, visibility, competitor index, etc.), providing fine-grained information about sales.

1.3 Main Objectives

The main research questions we wanted to answer were:

1. Which features are predictive of sales (i.e. orders placed to the warehouse)?
2. Powerful predictive models are often difficult to interpret. Can we explain which features are important to a business owner?

1.4 Approach

Keeping in mind our mission to improve product delivery by catsAI and their commitment to predicting product sales and explaining the factors that affect these predictions, we undertook a three-prong approach. We first explored different features that might affect sales, including the weather, location and seasonality. Informed by our initial explorations, we built predictive models. Finally, we used in-built and post-hoc explanation

methods to shed light on which features best explain sales predictions. Our approach was as follows:

1. Exploratory Data Analysis: to better understand what drives sales, we performed spatial and temporal analysis on the data (Section 5), which we describe in more detail in Section 4.
2. Predicting Sales: In Section 6.3, we describe the various models we built and validated, broadly categorised as white-box, which are easier to interpret, and black-box, which are better interpreted using posthoc methods. Since we worked on a limited feature set made available (due to preserving the privacy of a CatsAi client), we did not achieve the highest possible performance. Still, we identified several modelling directions helpful for catsAi.
3. Explaining Predictions: Our final analysis involves applying various explanation methods, broadly categorised as in-built like feature importances, and posthoc, such as LIME. These explanations work on both single instances (local) and the whole model (global). We describe these methods in Section 6.4.

1.5 Main Conclusions

Our analysis highlighted several valuable structural insights for model developers which the CatsAi team can leverage to improve their predictive models and explanations. For example, we found some correlation with weather data, like wind, temperature and visibility. However, further analysis would benefit from a more granular data set and further data availability, both of which would enable the incorporation of more features into machine learning models.

For explanation methods, we found that different methods yield similar explanations, implying that our analysis is stable and valid. Model agnostic post-hoc methods had overlaps with in-built methods for explaining predictions, which shows that both of these options are viable for generating client-friendly and accessible explanations.

1.6 Limitations

As mentioned in “Predicting Sales”, we worked on a limited feature set to preserve the privacy of the CatsAi customer and, therefore, could not experiment with other valuable features such as demographics of the customer base, information on competitors and fine-grained location information. Future work can exploit these features and build more powerful predictive models. Further work could also incorporate a larger customer base, facilitating the use of the above features while retaining anonymisation.

1.7 Recommendations and Future Work

1. There are several positive implications from our analyses. We show that some weather features play a role in sales, more so for certain products. A natural extension of our predictive models would be disaggregating by product type and building individual models for different products.
2. One could also explore additional modelling techniques such as agent-based modelling and directed acyclic graphs (DAGs). The latter is especially attractive for the scenario of explaining predictions, since they go beyond correlation captured by other post-hoc methods and encode causality.
3. We also found that location plays an important role. Therefore, using more location-related features such as distance to the city centre, the number of tourists and presence of competitors could lead to better predictions.
4. Finally, we showed through quantitative evaluations that several explanation methods can work with regression-based models we built. Future work can explore qualitative evaluations with these explanations, especially to gauge how helpful they are for end-users.

2 Introduction

2.1 Challenge Overview

The CatsAi challenge focus is on 'Communicating high-street bakery sales predictions using counterfactual explanations'. A central tenet of their project is the interpretability of machine learning models that predict the sales of a large wholesaler to smaller individual bakeries. Models that predict sales in the real-world need to be explainable to build trust with a large audience of business-minded stakeholders. While strong model performance is valuable, interpretability is critical - if the output of a model is not easily understandable, it will find very little use in practice. For this reason, CatsAi encouraged a focus on approaches that enable explainability of the sales predictions produced, particularly through 'counterfactual explanations'. They provided participants with a comprehensive dataset of historical sales and meteorological data across thousands of bakery sites. While catsAI anticipated a clear link between weather and sales, they understood that this would simply provide a start to understanding the key drivers of bakery sales.

2.2 Research Questions

We investigated several research questions throughout the data study group:

1. Can we use machine learning to develop models that are capable of predicting bakery sales from geo-locational and temporal data?
2. Can bakery sales be predicted with meteorological factors?
3. Can we interpret the output of the machine learning models we have developed?
4. What is the trade-off between the performance or complexity of a model and its explainability to a lay audience?
5. How can we evaluate the explanations produced by explainability techniques?

Our subsequent work centres around developing predictive models using

meteorological data (Section 2.3) in addressing Research Questions 1 and 2, and in the development of interpretable models (Section 2.4) in answering Research Questions 3, 4 and 5.

2.3 Predictive Models using Meteorological Data

Meteorological features are time-varying variables that originate from global circulation of the Earth, a world-wide system that drives different meteorological phenomena at different geological locations. To understand the weather pattern of a particular location, one must look at its past history (temporal variation) and local geological characteristics (spatial variation). Thus, a causal inference framework is required to verify the causal assumptions of the data generating process. Further details about this approach are given in Chapter 6 (Future Work and Research Avenues).

2.4 Interpretable Models

The complexity of the meteorological data and associated models required to capture underlying patterns, currently utilised by CatsAI, do not allow for easy interpretation. The multi-faceted machine learning pipeline, which includes clustering, anomaly identification and dimensionality reduction, achieves good performance, however it results in an opaque AI system. However, it results, in an opaque AI system. This performance-interpretability tradeoff is a paradigm plaguing those in industry who have recently applied machine learning to their business. Industry has therefore started to consider explainable AI, a research area that works towards building interpretable machine learning solutions for complex models and pipelines.

Developing explainable machine learning models for CatsAI would facilitate the business to build trust in their predictive systems, giving practitioners more confidence in making informed next steps. We hope that this case study will provide practical guidance to support the broader adoption of explainable machine learning and AI.

3 Exploring Explainable AI

We include below a preliminary explanation of some research approaches in explainable AI. Extensive reviews and far more details can be found in [1, 2].

3.1 Explainable AI: some recent progress

There are many different dimensions by which to categorise the existing literature on explainable AI, including considering the type of end user of the explanation, the stage in the machine learning pipeline an explanation is being applied to or the type of explainability metric.

3.1.1 Interpretable Models

Often the easiest way to achieve explainability is through the use of inherently interpretable models whose relative simplicity (when compared to black box models) offer a direct way of relaying model characteristics to the end user. Common interpretable models include logistic and linear regression, decision trees and rule based models. There are those in the explainable AI space who argue the only way to achieve true explainability is through the sole use of interpretable models and we should stop trying to explain black box models [3]. However, this is hotly contested within the domain and the predictive performance of black box models is hard to ignore.

3.1.2 Model Agnostic Methods

Model agnostic explainability methods are perhaps the subset of explainability techniques that have attracted the most popularity. Their applicability to almost all machine learning models is desirable.

Global Model Agnostic Explanations: Global explanations offer insight into the behaviour of the overall underlying machine learning model. These include methods that learn an interpretable model alongside the black-box to be used as an explanation module [1].

Local Model Agnostic Methods: Local explanations provide

justifications for the individual predictions made by the machine learning model. These include feature importance measures such as LIME [4], SHAP [5] that question features were important for a particular prediction, visual explanations such as saliency maps as well as counterfactual explanations that reframe the question to ask, which features would have to change to change the predicted class of a model.

3.1.3 Model Specific Methods

Desite not offering the generality of model agnostic explainability, the customisation offered by model specific explanations often means they are able to avoid some of the pitfalls associated with model agnostic methods. These include specific methods for support vector machines and neural networks.

3.2 Issues with some state of the art methods

Despite the recent progress within explainable AI there are several issues that represent the open research questions within the space [2]:

- **Bad Model Generalization**: Under- or overfitting models will result in misleading interpretations regarding true feature effects and importance scores, as the model does not match the underlying data generating process well. An interpretation can only be as good as its underlying model. It is crucial to properly evaluate models using training and test splits and cross validation methods. Flexible models should be part of the model selection process so that the true data generating function is more likely to be discovered.
- **Unnecessary use of Complex Models**: A common mistake is to use an opaque, complex ML model when an interpretable model would have been sufficient. Authors recommend to start with simple, interpretable models such as (generalized) linear models, LASSO, generalized additive models, decision trees or decision rules and gradually increase complexity in a controlled, step-wise manner, where predictive performance is carefully measured and compared. Complex models should only be analysed if the additional performance gain is both significant and relevant.

- **Interpretation with Extrapolation**: When features are dependent, perturbation-based interpretable machine learning methods such as the Permutation Feature Importance (PFI) and Partial Dependence Plots (PDP) extrapolate in areas where the model was trained with little or no training data, which can cause misleading interpretations. Before applying interpretation methods, practitioners should check for dependencies between features in the data. ALE plots are preferable to the PDP when visualizing feature effects of dependent features. Furthermore, dependent features should not be interpreted separately but rather jointly. This can be achieved by visualizing, e.g., a 2dimensional ALE plot of two dependent features.
- **Correlation is confused as causation**: SelfExplanatory problem. Solutions can be a lowdimensional data can be visualized to detect dependence (e.g., scatter plots). If dependence is monotonic, Spearman's rank correlation coefficient [6] can be a simple, robust alternative to PCC. For categorical or mixed features, separate dependence measures have been proposed, such as Kendall's tau for ordinal features, or the phi coefficient and Goodman and Kruskals lambda for nominal features. several nonlinear association measures with sound statistical properties exist. Kernelbased measures such as kernel canonical correlation analysis (KCCA) [7] or the HilbertSchmidt independence criterion (HSIC) [8] are commonly used. In addition to using PCC, use at least one measure that detects non-linear dependencies (e.g. HSIC).
- **Misleading Effect due to Interactions**: Global interpretation methods such as PDP or ALE plots can produce misleading interpretations when features interact. While PDP and ALE average out interaction effects, ICE curves directly show the heterogeneity between individual predictions.
- **Ignoring Model Variance and Estimation Uncertainty**: Due to variance in the estimation process, interpretations of ML models can become misleading. Methods such as PDP and PFI use Monte Carlo sampling techniques to approximate expected values. By repeatedly computing PDP and PFI with a given model, but with different permutations/bootstrap samples, the uncertainty of the

estimate can be quantified, for example in the form of confidence intervals.

- **Unjustified Causal Interpretation**: Practitioners are often interested in causal insights into the underlying data generating mechanisms, which ML methods in general do not provide. Consequently, the question whether a variable is relevant to a predictive model does not directly indicate whether a variable is a cause, an effect or does not stand in any causal relation to the target variable. The PDP between a feature and the target can be interpreted as the respective average causal effect if the model performs well and the set of remaining variables is a valid adjustment set. Designated tools and approaches are available for causal discovery and inference [2].

3.3 Beyond correlation based explanations towards counterfactuals

Many of the above issues with state of the art explainability metrics were noted by Russell et al. [9], who argue that counterfactuals should be the standard explainability mechanism for individual predictions. Counterfactuals are explanations in the form of “By what amount do I need to change feature X to change label Y” e.g. “By how many degrees does the temperature need to change to order 100 more croissants?”.

Counterfactuals should provide actionable insights rather than unchangeable reasoning leading to certain predictions which is common with usual explainability algorithms. For eg: a person’s loan being rejected due to poor credit history but doesn’t clarify what steps can be taken to improve this. Counterfactual would say ”if you earned 10,000$ more you would have received the loan”. Single counterfactual explanation may not be useful as sets of conterfactuals, depending on personal circumstances.

It is not possible to identify individual-level causal effects from the observational data. However, if the conditions of *positivity*, *consistency* and *exchangeability* are satisfied, average causal effects are shown to be identifiable [10].

4 Data overview

4.1 Dataset Summary

The data provided was for one client (a large wholesaler) catering to several sites across a single country for 2018 and 2019. Multiple datasets covered characteristics such as location, product details and weather variables. All the datasets were merged prior to use. Each data point pertained to sales, product and weather data for a single site on a single day of the corresponding year. We include a detailed glossary of features and their definitions in Appendix 9.1.

4.2 Data Description

The data falls into the following broad categories.

Location: The location feature was broken down into several levels of detail starting from "*Level_1*" being the country/state to "*site*" which was individual bakery sites. There were 5118 sites in total across the two years of data provided. There were seven unique values in the "*Level_2*" feature pertaining to counties/districts, which were focused on during model building.

Product: There were 45 unique products each belonging to one of the seven Family and Category features, explained in Appendix 9.1. Another additional feature Units per order included the number of items per box.

Weather: There were 41 weather variables provided which included several indicators such as temperature, pressure and wind gust which were then analysed for consistency.

As discussed in the next section, there were inconsistencies found in different feature variables that were independently tackled.

4.3 Data Quality Issues

We appraised the quality of the data. Here we describe the approach taken to identify and address inconsistencies in the data.

4.3.1 Missing Data

Figure 1 shows that there were several missing in the various features. Of particular interest was that 83% of values of '*sales*' (our target variable) were missing. We consulted with the challenge owner who confirmed that a missing value meant there were no sales for that product on that day for the specified site. Therefore, this missing data proved instructive for our prediction task. In the rest of the report we will refer to missing sales data as "zero sales" to differentiate this from months where data is absent (i.e. Jan-Feb '18 and Nov-Dec '19), as described in Section 4.3.2 below.

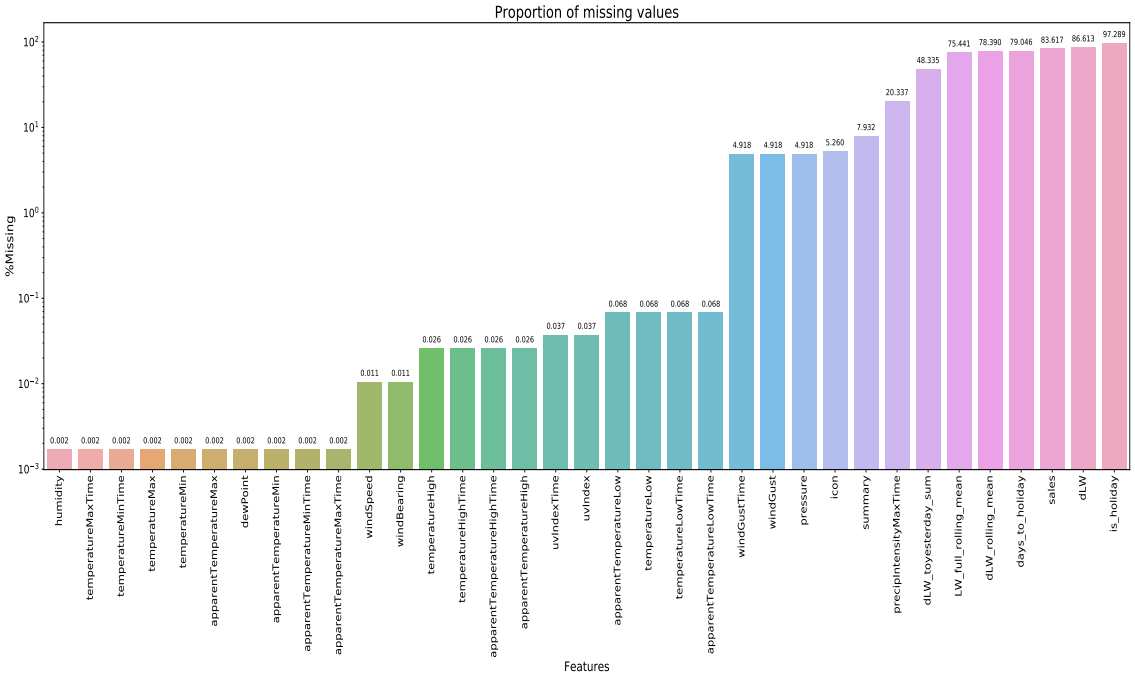

Figure 1: Missing Values as a percentage of all entries

On investigating the missing sales data further, Figure 2 shows January and February 2018 as well as November and December 2019 did not have any data for this characteristic. This was an important finding that influenced the model building and conclusions sections.

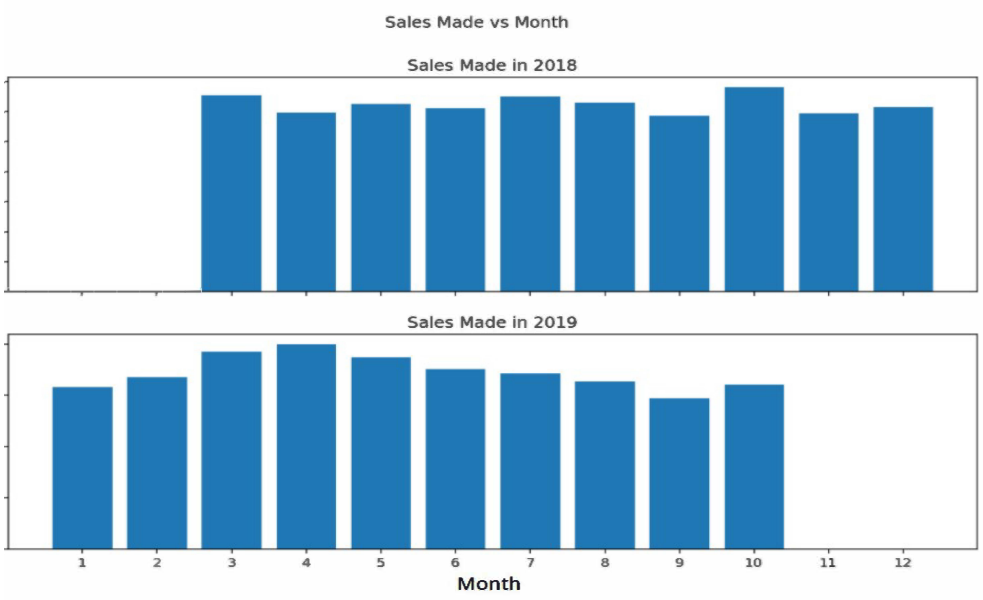

Figure 2: Sales for each month (2018-2019)

4.3.2 Predictive Limitations of Missing Data

We faced challenges due to the limited period for which the data was available. To better capture the seasonal variation of sales with respect to weather, we would have benefited from data collected over a longer period with more consistent weather descriptions. We found contradicting data in the weather variables such as *icon* (Eg: rain, clear-day), *Summary* (Eg: Rainy morning) and *cloud cover*. Since the weather data was largely a daily average, it did not necessarily remain so across the day. Based on the exploratory data analysis, unanimous decisions were made on which characteristics would benefit the predictive models.

5 Exploratory Data Analysis

5.1 Data Wrangling

Multiple datasets had to be combined coherently for analysis. The data was available in monthly frequency and were combined to form a single

dataset for 2018 and 2019. There were different datasets for product, site and weather data which were united using common keys like *productId* and *siteId*.

We used the dataset which already had *weather_local* information in it, but augmented it with other data we found in *weather_city*. *weather_local* is used by catsAI API. *weather_local* has information about the city itself; where values were missing we used values provided in *weather_city* which covers a wider radius.

The combined dataset had 4.5 million rows and 60 columns. The dataset had 45 unique product names and 5118 unique sites.

5.2 Data Analysis

5.2.1 Product Sales Analysis

We first analyse the sales of products across different categories over time. The motivation behind this step was to see whether any particular products were being ordered more or less in a particular month or season. We noticed 47 unique entries for *productId* but 45 unique product names. On further inquiry, we found out that two products have two IDs, namely **Large Salt Bread** and **Muffins**. Hence, we decided to use *Product Names* for this analysis which helped since we could associate products with sales rather than referring back to the *product_info* data.

Figure 3 illustrates that **Donuts, Croissants** and **Swirl** are the most ordered items in general in our dataset (regardless of precipitation, rain or snow). **Extra Large Baguette** tends to be ordered when the weather is nice. **Savory Ready to Proof** are always popular (especially during Spring and Winter). Between March and October there were more orders (this is because some months were excluded because of holidays). **Savoury read to proof** is popular in all seasons. **Pastry fully baked** is very popular in spring compared to other seasons.

5.2.2 Monthly and Seasonal Analysis

The monthly and seasonal analysis was conducted to identify the '*sales*' distribution among months for each year and among the different

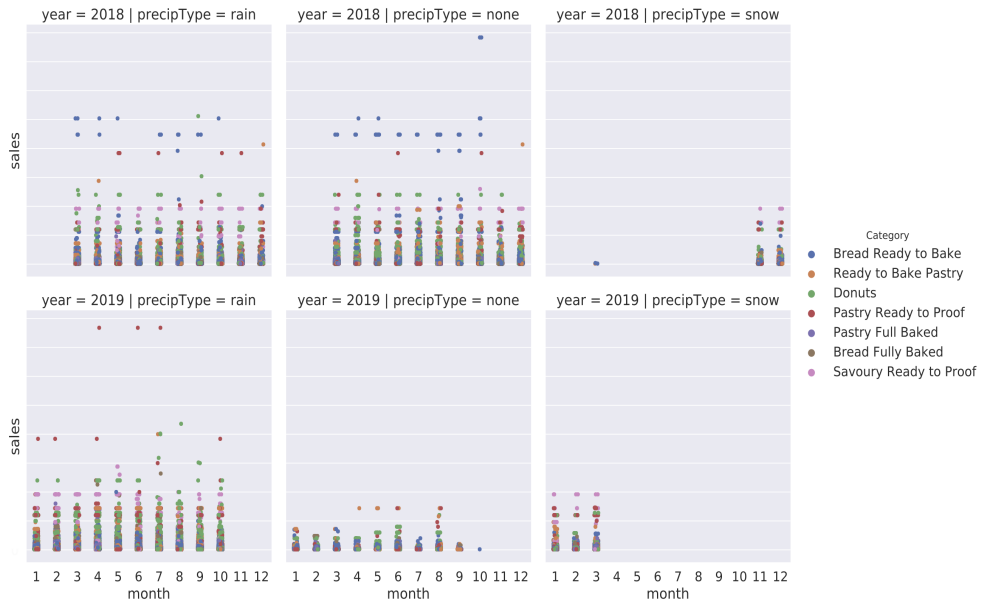

Figure 3: Sales over months/years across different categories differentiated over Precipitation Type

seasons. The month and year variables were extracted from the date column. We could see that data is absent for January and February 2018 and November and December 2019. Some months were preferred over others; however, these spikes in preference were not consistent. In 2018, October, July and March took the lead, with October having substantially higher sales than the other months. While in 2019, March to May saw huge sales spikes.

For a seasonal analysis, we aggregated the sales data and grouped them by each season. The seasons were divided as follows:

- Spring: March, April and May
- Summer: June, July and August
- Autumn: September, October and November
- Winter: December, January and February

The seasonal data showed a clear bias towards Summer and Spring; however, there were a few missing months making seasonal analysis

unreliable. When Autumn had complete data, it was at the same level as Summer and Spring while Winter was absent from the dataset in both the years; hence, it displayed a perpetually underwhelming performance when it came to sales. However, as seen in the monthly data, the spring of 2019 was a high sales period, with all three months being the top 3 sales periods.

While monthly data and seasonal data did not show the seasonality we were expecting, we see a huge bump in sales during the summer and spring months. In 2018, Autumn is comparable to the other two months only because of the huge bump in October Sales. While a period of approximately two months is absent from the dataset, even if the sales of Summer and Spring values are divided by three, they would still be more than the winter sales. This indicates that lower temperatures discourage sales and a pleasant temperature with light showers might be favourable towards bakery sales. We will turn to weather data for further analysis.

5.2.3 Comparing prevalence of different weather condition on a monthly basis

To understand the prevalence of different weather conditions monthly, we calculated the ratio of different weather conditions, as indicated by the *icon* feature in the dataset, on the left axis and plotted the total monthly sales on the right axis (Figure 4). The figure shows that the month with the most sales happened between March 2019 to May 2019. However, those are the months with well-mixed weather conditions, and hence we could not identify a strong link between *icon* and *total sales*. Another finding was the shift in weather pattern between the two years provided to us; weather in 2018 seems to favour more clear days than 2019.

We took the ratio of every weather condition under feature *icon*, plotted it on the left axis, then plotted the sale for every category of pastry on the right axis. We find some products have seasonal preferences; for example, ready to bake items are less popular in winter than other times of the year; those months have the highest mixture of weather conditions. The months with increased rain shows a higher purchase of **Ready to Proof** bakery items.

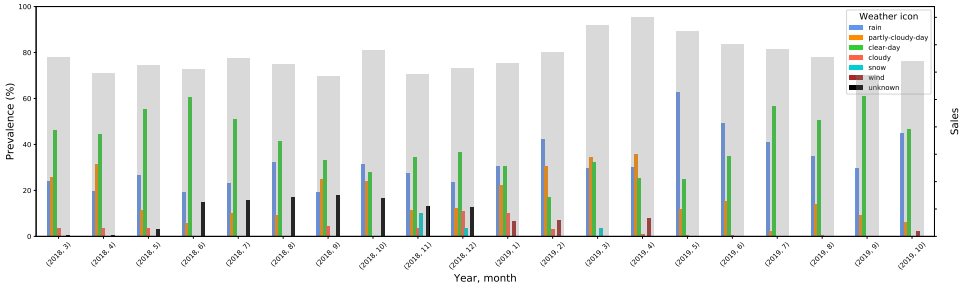

Figure 4: Prevalence of weather type on a monthly basis.

Between March and October, there were more orders (this is because some months were excluded because of holidays).

5.2.4 Sales on Weekends and Holidays

We found out very early in this analysis that Sunday has no sales whatsoever. Consequently, we added *is_sunday* to the dataset and updated the one shared with everyone. This allows us to provide a shortcut in prediction models to predict zero for Sundays. There was no variation between Saturdays and other days of the week.

Next, we looked at bank holidays, which sometimes actually have orders on that day. We looked at four days before and up to four days after bank holidays and plotted a double plot that highlighted these days. Bank holiday has no immediate impact on sales.

5.2.5 Sales vs UV index for each category

Figure 5 confirms that there's a clear tendency of placing more orders when the weather is nicer (clear sky, higher uv index, low to none precipitations, as shown above) compared to when it's rainy or cold.

Figure 5: UV INDEX

5.2.6 Sales behaviour for each product category and weather variables

We focused on the interaction between the sales of a particular product category and its relationship with a particular weather variable. We plotted scatter plots for numerical weather variables (Figure 6). The scatter plots help identify how the sales are dependent on numeric weather variables for each data point. These are presented for precipitation intensity in Figure 6 and Visibility in Figure 7. There is a clear preference for 'clear' conditions: We can see that despite monthly variation in sales, shops tend to order more products from the warehouse when it is not raining.

5.2.7 Location Analysis

To understand how the location of the sites contributed to the sales data, we explored the contribution from each *Level_2* district in terms of log of total sales (Figure 8). It is evident from the figure that most of the sales are contributed from two districts, *Level2_2* and *Level2_3*, which means that any trained model will inevitably be biased towards the customer behaviour

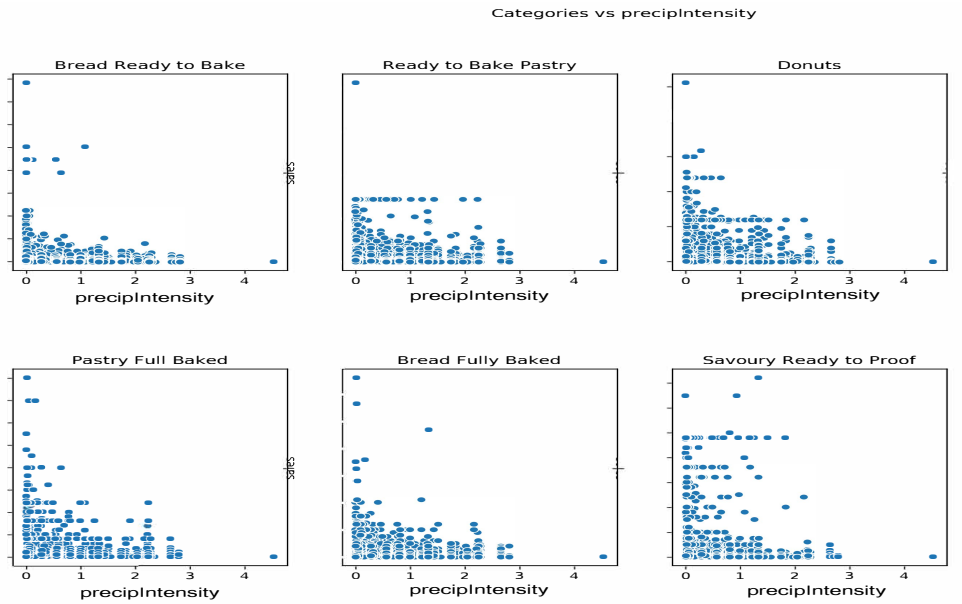

Figure 6: Scatter Plots for each category vs the Precipitation Intensity (2018-2019)

Figure 7: Scatter Plots for each category vs the Visibility (2018-2019)

in the first two districts and have an impact on the ability of the model to generalise to other districts.

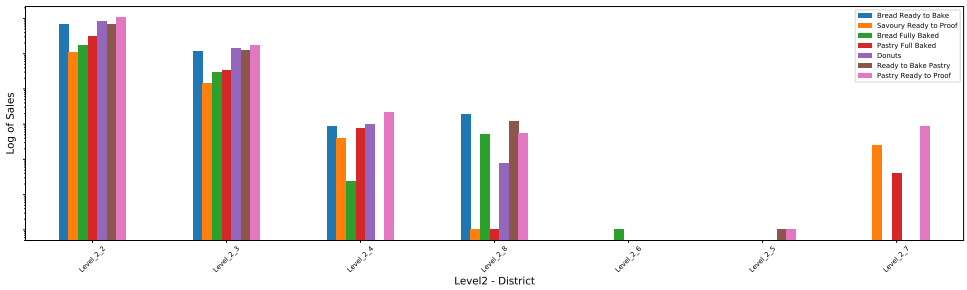

Figure 8: Total Sales in different level 2 districts

5.2.8 Sales trend analysis

Daily Sales Pattern. To understand if there are any obvious trends (e.g. seasonality, outliers), we plotted the daily total sales from every site (blue line)(Figure 9). The orange line is the rolling average of the daily sales with a 30 days window. The date of public holiday is also plotted here (red dotted line). There is a frequent drop to zero which mostly corresponds to Sunday, when the warehouse closed down and did not receive any orders from bakeries. By inspecting the daily sales variation, there is an indication that sales are increasing from Spring 2019 to late Summer 2019. This increase is more evident if we look at the 30-days rolling mean. This increase in sales is unseen in 2018, which is mostly flat compared to 2019. The long duration of increased sales is also unexplained by seasonality nor public holidays. The difference between the two years flags the issue of inhomogeneity. This prompted us to focus on instead on 2018 rather than the entire datasets.

5.2.9 Week on week sales pattern

While looking at the data and our part of our understanding of the problem we formed a hypothesis that if a bakery under/over-orders in a certain week, the next week they will over-/under-order to compensate, and after that, they should stabilise without a sudden increase in that order. To investigate the hypothesis, we plotted sales on a particular day against

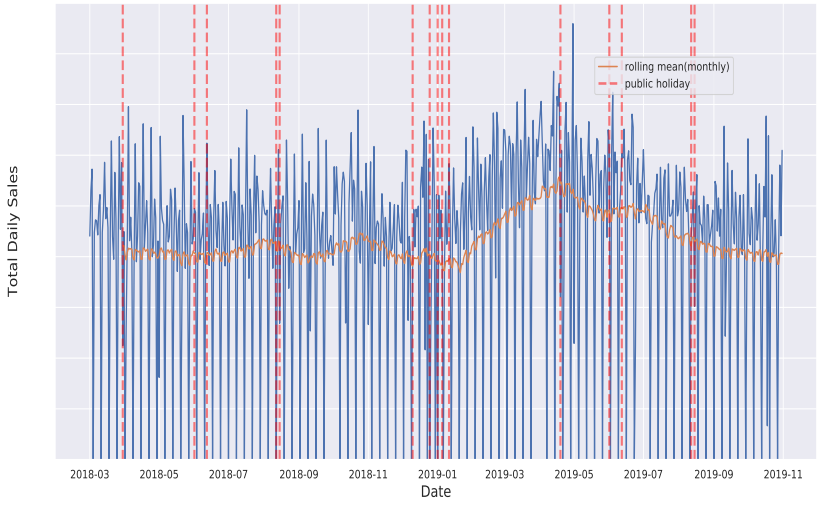

Figure 9: Total Daily Sales

the sales of the previous week and aggregate the results. The heatmap 10 shows the pattern that validates the hypothesis; lighter colour shows a substantial increase in orders placed from the bakery, often followed by a darker colour, meaning the following week they under-ordered, but soon after it stabilises and averages around 0 difference with +-200 variance between a week on week.

This property shows that knowing the placed order of the last week can be influential in a model that aims to learn a pattern in a time series, as highlighted in Figure 10.

5.2.10 Daily Sales pattern in each category.

We extended the same approach from the previous section and applied it to different categories. We realised that different categories exhibit very different patterns in terms of sales. For instance, sales in the **Pastry Fully Baked** category seems to have a recurring peak around the April period. While other categories like **Pastry Ready to Proof** and **Donut** seem to follow the global sales trend outlined in the previous section.

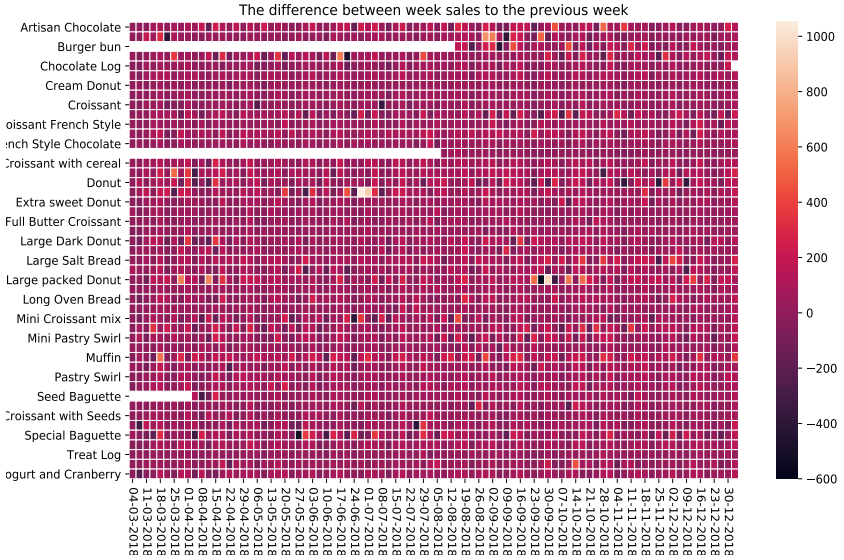

Figure 10: Heatmap showing the difference between a sale on a specific day compared to the sale of that product the week before, allowing a visualisation of any possible patterns of week on week purchase patterns.

Each category's different sales patterns hinted at the difficulty in deriving a general model for every product and every site (Figure 12).

Figure 11: Total Daily Sales in each category

5.2.11 Sales seasonality pattern analysis.

It appeared that there is some inherent seasonality to it: certain days will be more popular with bakeries than others. We guessed Monday after a weekend and Thursday closer to the weekend are days when bakeries place their orders, and to explore this hypothesis we produced plot 12. In the plot, the x-axis shows the date and plotted against the total daily sales of all products and sites. While the aggregation is too granular, the main goal of the plot is to find if there are any seasonality patterns. To detect the seasonality we used a Gaussian Filter [11] and used in [12] for seasonality analysis. We used a sigma of 1 to capture daily behaviour, and sigma of 3.5 to capture the monthly trend.

Plot 12 captures the seasonality trends. The green shaded labels show a sine-wave like pattern week-on-week for the total daily sales indicating that we can reasonably assume days are indeed a factor in the total order. The month-on-month (yellow in the plot) shows some pattern, but it is not a seasonality pattern. Further to this plot, we noticed that sales dip on Sundays, and we confirmed with the stakeholder afterwards that no sales happen on Sunday.

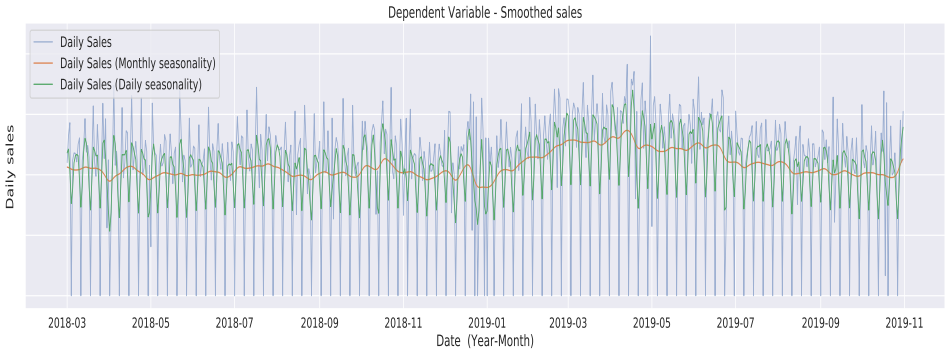

Figure 12: Daily and monthly seasonality trend for total sales in every location.

	Random Forest	Gradient Boosting Machine	XGBoost	MLP	Time Structured Prediction
Replaced with mean	Overall mean: pressure	Overall mean: temperatureHigh, temperatureLow, humidity, windSpeed, temperatureMin, temperatureMax	grouped mean: temperatureHigh, temperatureLow, pressure, windSpeed, windGust, windBearing, uvIndex		KNN mean: Temperature features
Categorical encoding	label encoding	label encoding	label encoding	one hot encoding	label encoding
Removed features	Product name, family, category, level 1, level 3, moon_phase, days to holiday, summary	Removed features with >5% missing values	Product name, family, category, level 1, level 3, days to holiday, summary		product name, family, category, level 1, summary
Date transformation	Yes				No
Standardisation	No	No	No	Yes	Log of sales

6 Experiments

6.1 Pre-processing Steps

We performed several preprocessing steps before we fed the data into various black-box/white box models. As preprocessing steps are tailored to different methods, Appendix 9.2 summarises the different approaches we pursued in the process. For all approaches we have performed the following two steps:

1. Removed observed weather time-related features. Features that specified the observed weather time.
2. Added additional features: *is_sunday* and *days_to_holiday*.
3. Replace missing values in sales-related features with zero. This step does not affect the interpretation of the data as no sales is treated as missing in the first place

6.2 Performance Metrics

We use the R^2 metric to evaluate all predictive models' performance since it is a standard metric used for evaluating regression problems[]. R^2 was also the metric used by CatsAi. Future work should explore how other metrics such as Mean Average Error (MAE) and Root Mean Squared Error (RMSE) compare with R^2 for this particular scenario.

Train Data	Test Data	Model Performance
All of 2018	All of 2019	0.22
Mar 2018 - Oct 2018	Nov 2018 - Dec 2018	0.23
Mar 2018 - Aug 2019 (Top 5 features)	Sept 2019 - Oct 2019 (Top 5 features)	0.25
Mar 2018 - Aug 2019	Sept 2019 - Oct 2019	0.26
Mar 2018 - Oct 2018 (filtered by category - Bread ready to bake)	Nov 2018 - Dec 2018	0.28
All of 2018 (filtered by category - Donuts)	All of 2019 (filtered by category - Donuts)	0.29

6.3 Predictive Models

We experimented with various black-box and white-box models for generating sales predictions.

6.3.1 Random Forest Regressor

The first tree-based model that we implemented to predict the orders that a bakery places on each day was the Random Forest Regressor, an ensemble method that uses the concept of bootstrap aggregating or bagging. It is a powerful tree-based algorithm that considers the bias and variance within the dataset by random sampling and replacement. It is also an explainable model which shows us where the tree was split and on what conditions. For this model, we trained the RandomForest on multiple data splits and achieved a variety of results. The first split was 2018 data for training and 2019 data for testing. We achieved an R^2 score of 0.22 on this split. On changing the testing data to September-October 2019 and training data to March 2018 to August 2019, we achieved an R^2 score of 0.25 which was the best performing RF on the complete data. Furthermore, when the data was filtered by category, the R^2 scores improved, leading to the conclusion that category-specific models will be more effective. Below, we have laid out the combinations of train-test split used to build the models. We include the full list of features in Appendix 9.3. Once the model was created, we also performed feature importance to identify the features which contributed the most to the models. Figure 13 shows that the 10 most important features.

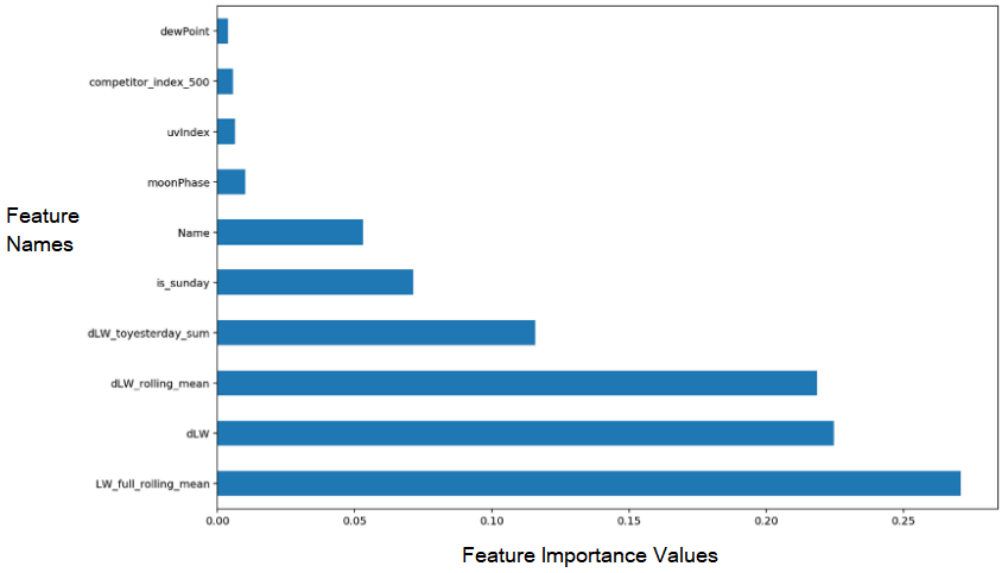

Figure 13: Random Forest - Feature Importance plot

Gradient Boosting Regressor			
Model Type	Train Data	Test Data	Model Performance
Model with all the features [1]	Mar 2018 to Aug 2019	Sept and Oct 2019	0.2483
Model with only the 5 most important features [2]	Mar 2018 to Aug 2019	Sept and Oct 2019	0.2468
Model with all the features	all of 2018	all of 2019	0.2351
GBM hyperparameter tuning via grid search	Subset of all data stratified across sales for 2018 and 2019		N/A
Model with 22 features [3]	80% of all data stratified across sales for 2018 and 2019	20% of all data stratified across sales for 2018 and 2019	0.2873

6.3.2 Gradient Boosting Regressor

The second tree-based model that we implemented to predict sales was Gradient Boosting Machine (GBM), a supervised machine learning technique for regression and classification problems. GBM combines a group of weak learners in order to enhance the predictive performance of the model. In Python, these models were built after removing outliers. They were trained on 2018 and 2019 except September 2019 and October 2019, which were used to assess the models' performance. Different gradient boosting regressor models were built; a brief overview of the variables used to train and test data, including the model performance is provided in the table below.

6.3.3 Extreme Gradient Boosting (XGB)

One of the tree-based models we implemented to predict sales was Extreme Gradient Boosting (XGB). XGB is a fast implementation of Gradient Boosting and allows users to obtain good performance and training speed in most real-world problems. Because of its flexibility, performance and intrinsic feature selection, it is often considered state of the art in many applications. The model works by iteratively building a set of high-bias weak learners (models that perform slightly better than chance) and finally combining them into strong learners.

We trained an XGBRegressor on 2018 and most of 2019 data which was tested on the remaining 2019 data. Before training the Regressor, we removed outliers and encoded the categorical features present in the dataset. After grid-search, the best performance we obtained was using the following hyperparameters; "eta" of 0.1, a "max_depth" of 8, "subsample" of 0.8 and "gbtree" as booster.

6.3.4 Feed-forward Neural Network or Multi-Layer Perceptron (MLP)

Since deep learning neural networks have a reputation for being the most opaque machine learning models, we thought it would be interesting to explore the interpretability of this approach. A simple multi-layer perceptron was selected because it requires significantly less processing power than a Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN) and because it was already successfully employed by the CatsAi team. The use of a Long Short-Term Memory (LSTM) [13] network was also considered due to this algorithm's inherent ability to process sequences and text well. As such, it would be valuable to consider this approach in the future since MLP's have no memory and are probably not the best-performing or state-of-art technique for this problem.

It seemed valuable to include all features at the onset since the standard deep learning approach largely excludes manual feature engineering in pursuit of abstract representations learnt by the algorithm of choice. However, since a key metric of this work is explainability to a lay-person, it was necessary to select a subset of important features that could perhaps predict sales.

In terms of pre-processing for the neural network, our initial approach was to create a data.matrix() from the data.frame() containing both variable types - categorical and numerical. However, the neural network did not seem to respond well to this structure. We then switched to treating each variable type differently. First, we combined all categorical variables and one hot encoded them and then we combined all numerical variables and normalised them within one standard deviation of the mean. The target variable, 'sales', was simply extracted from the full dataset for 2018 and transformed into an array using the as.array() function. In order to train a model, it was decided to use 2019 data for the held-out test set since the temporal order of this data was an important consideration for the problem statement at hand. The test data was pre-processed in the same way the training data had been.

The model architecture can be described as a basic fully-connected network. CatsAI made use of a model with one hidden layer, but we opted to use two hidden layers of 10 units each and a 'relu' activation. The output layer contained only 1 unit for the sales target and no activation function. An 'adam' optimizer was used with an 'mse' loss function. At first, we wanted to train the network for 100 epochs but subsequently scaled it down to 50 with a matching batch_size of 50. Other hyperparameters that were used included a dropout of 0.1 in each hidden layer and an L1 regularisation of 0.001.

The best practice for validation for this kind of approach would be to use k-fold cross validation. This was attempted initially using a k value of 5, but due to a slow CoCalc environment and other issues, this choice of validation was discarded in an attempt to get a working model. Also, we tried tuning the parameters (i.e. L1 regularisation, n_units, and dropout) of the neural network using random samples of 10% of the whole training dataset. Natural next steps for this work would be to consider a greater proportion of the training set for parameter tuning. During the Data Study Group, we did not have time to perform hyperparameter tuning using the complete training set. The final model performance with 2018 as training data and 2019 as test data was 0.26.

6.3.5 Structured Time Series Prediction

Model summary. We developed a white-box model to forecast sales. The model is a Bayesian structured time series (STS) model that captures the seasonality of the features and explains it by inspecting the model, the model additionally allows answering questions of the form: ”what is the likelihood of sales being under 700 but over 400.”. The model achieved an R^2 of 0.68 when tested against December 2018 after training on the earlier months.

Model background. An alternative approach to explainability is to use a white-box explainable model in the first place. The term white-box refers to the idea that each component of the model is something a scientist can look at in isolation and interpret its results. Often these are much harder to develop as the scientist requires specific domain knowledge and deep understanding of their data, but once those are captured the model can learn faster and outperform other more ”complex” methods. The main motivation behind these models is that we humans already have developed years and decades of expertise in a certain problem area, we can feed this expertise to models and let them focus on capturing the things we do not know rather than the things we already know.

Bayesian structural time series models [14] are techniques for incorporating domain expertise over the analysis of time-series data. These models are an easily interpretable and common technique to reason with time-series data, for example, Brodersen et al. [15] used STS models also used to infer causality and reason with the market response in the field of economics and marketing. These models are an appropriate choice for the Bakery sales prediction problem; as we observed from the exploratory data analysis we performed and discussed earlier, we recognise many of the features are directly impacted by the time of the year (with shifting pattern from a year to year due to climate change phenomena that can be captured easily as well).

These models provide several advantages over traditional ML models:

- They capture the seasonality trend over various time-steps.
- They capture linear locality of features: the interaction between

features around the same time step.

- They handle anomaly and out of distribution data by incorporating priors and not be impacted significantly by data that does not follow the general distribution.
- It is easy to extend them to handle externality by fitting an autoregressive layer that attempts to learn a latent random variable that captures anything that is not defined in the feature set—allowing the scientist to know if they have enough features in their model or they need to incorporate more domain knowledge.
- They forecast the future and provide uncertainty intervals, allowing queries such as: ”What is the likelihood of selling between 300 and 400 doughnuts in the London bakery in 3 days”. The ability to enable questions with lower, upper or various statistical language makes them very powerful models that facilitate greater levels of interpretability.
- They allow a large degree of freedom of expressing the output shape and various input variable, thus allowing a mixture of features to be used: boolean indicators, a running mean for sales or category for pastry type.
- Infer causality between features and output predicted variable by marginalising every feature and comparing the weight impact on the overall sale.

Since these models incorporate domain knowledge information, they can learn from very little data and generalise well.

Tools used. Tensorflow Probability (v0.11) was used in building the Bayesian model. Tensorflow (v2.4) was used for features matrix multiplication.

Model specification. Here we describe the building blocks of the STS model and the prior assumption we had. The building blocks for these models are predefined models found in the TensorFlow probability package.

We fitted a linear trend model over the date feature via a Gaussian random walk with a mean of 0 and variance of 1 and a step-size of 7,

allowing us to capture week-on-week trends. For *days_to_holiday* feature, we created a masking layer over the days that are not within five days of a public holiday, and this layer ensured that the weights are not impacted in those days. The prior here is that holidays only affect sales when they are +-4 days. We then fitted a linear regression for this feature. For every weather feature, we standardised them over the distance from their respective mean value and then fitted a linear regression model over them. All the model components were summed over in a final layer and summarised using a mixture distribution.

We performed stochastic variational inference optimisation for training using Adam optimiser with a learning rate of 0.1 and over 1000 steps. Given more time or computational resources, NUTS or HMC sampling would have produced a better fit for the model.

Evaluation.

We trained the model on the 2018 dataset, the model was trained on the months' March to November and was tested on December's month. As a simplification, we looked at a single city (*locality_29*) and the overall sales for that day of all sites. The model achieved an R^2 of 0.68, however, worth noting Bayesian models provide uncertainty and upper and lower bounds of the prediction and the R^2 score doesn't take that into account. Figure 14 shows 100 draws from the predictive posterior distribution of the STS model. The figure shows the model was able to follow the trend of the data and the ground truth is within the expected variance of the model, on the y-axis is the daily product sale in logarithmic scale, and the x-axis is the date.

Explainability The main aspect of the challenge is understanding the impact of the features on the model, this subsection will go through two features and what the model learnt of them, we refer the reader to the appendix **??** for a visualisation of every feature analysis and its impact on the sale.

Temperature impact on sales Here we inspect what the STS model on both main temperatures features *temperatureHigh* and *temperatureLow*. The first plot 15 shows the weights of the linear regression layer with 100 draws from its posterior distribution. The *temperatureHigh* had a strong correlation to sales during the summer months (higher temperature), and

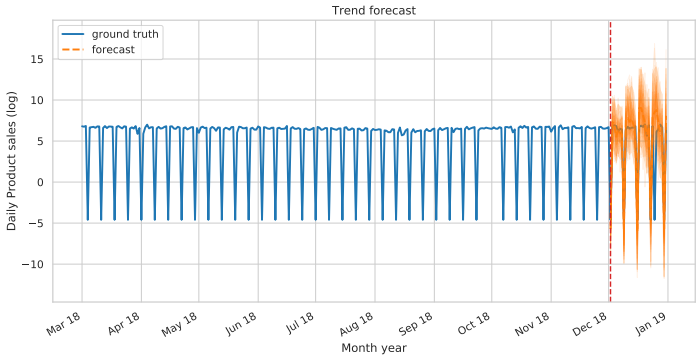

Figure 14: STS model forecasting the sales for the December month.

the opposite effect during winter and fall. The variance of this feature is very tight, i.e. the model is more confident about the impact it has on the overall predictions, and this communicates to the bakeries that the *temperatureHigh* in summer plays a big role in their orders.

A similar story is shown in the second plot 16 of the *temperatureLow*. However, the main difference is this feature has a higher variance meaning the model is not as confident. However, it still follows the same trend of the *temperatureHigh*, and this communicates that while the *temperatureLow* of the day plays a role, the impact of it changes depending on other factors, this is very obvious during winter months when *temperatureLow* has a higher variance with the p95 weight of -1.5 - meaning the temperatures in these months has a strong negative correlation with the sales.

Day of the week on sale In figure 17 we inspect a feature that follows a seasonality trend, please note we did not tell the model about the impact of Sunday on the sales nor told it about the seasonality of the days. The model learnt by performing a Gaussian Random walk of size 7 (days) over the date feature and learnt that no sales happen on Sundays, while Tuesdays and Thursdays are more popular for sales, this remained consistent throughout the year. A similar analysis was done on the *days_to_holiday* feature as well as shown in the appendix **??**.

Future extensions. The STS model is complex to set-up but simple to reason with, due to the time limitation of the DSG we made many

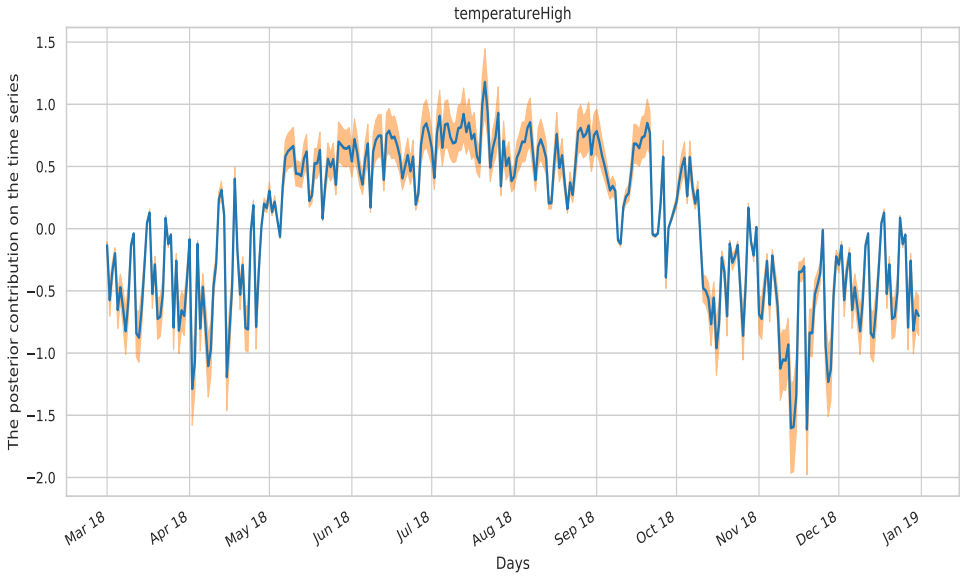

Figure 15: The posterior distribution of the weights in the linear regression layer of *temperatureHigh* feature.

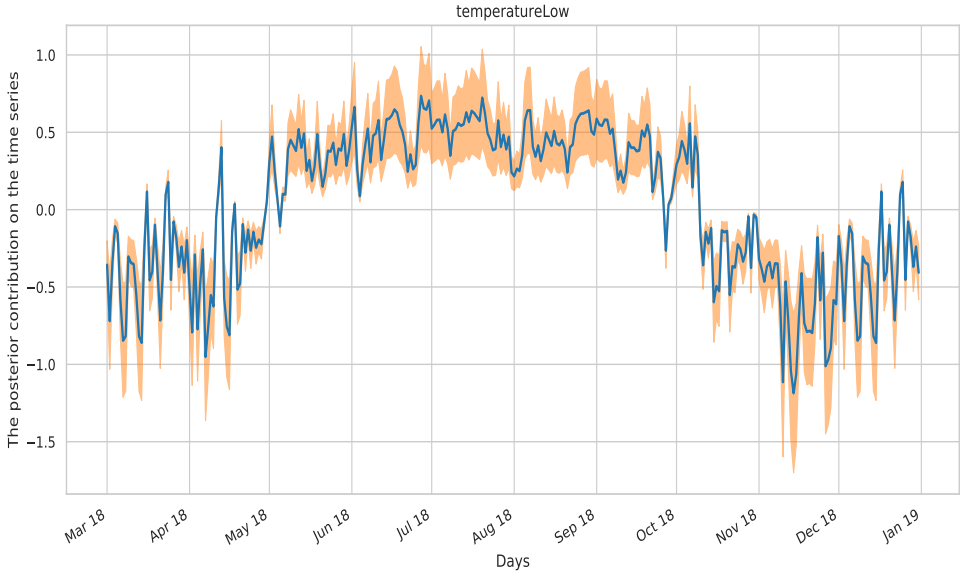

Figure 16: The posterior distribution of the weights in the linear regression layer of *temperatureLow* feature.

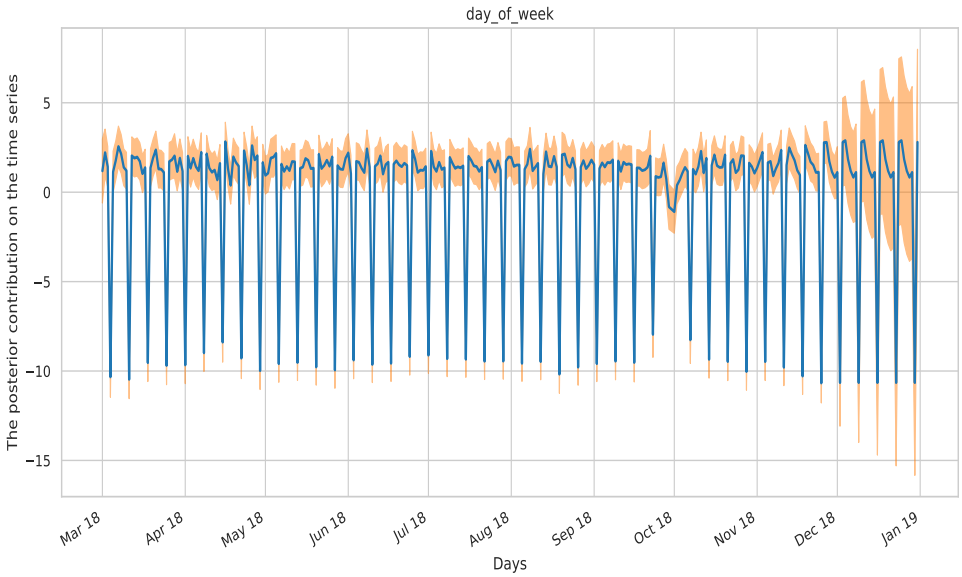

Figure 17: The posterior distribution of the weights in the linear regression layer of *date* feature.

assumptions to expedite the model development process, here we state the assumptions and the possible mitigations:

- The model operated only on locality. **Mitigation**: introduce a Categorical distribution that captures a different probability distribution for every locality within the dataset, allowing the model to be trained on the joint distribution of every site independently of each other whilst sharing knowledge between them - for example, customer behaviour around the holidays.
- The model output is log scaled: **Mitigation:** Training on non-normalised sales records but have a bijector that transforms the input and output of the model to Poisson distribution. The justification here is that sales are a form of count variable and Poisson distribution is a perfect fit for these, its output always positive and able to capture the generator process of these variables.
- The model ignored family, category, name, and site features.

Mitigation: Similar to the locality, these are a form of categorical random variables, fitting a different categorical distribution for each variable will allow the model to learn any ”same-family” ”same-category” or ”same-site” behaviours and provide even more in-depth explanations of predictions.

- The training was approximated using a variational inference method. **Mitigation**: variational inference is a fast class of training method in Bayesian landscape by approximating the result into an exponential distribution family form. Instead, sampling methods such as HMC methods provide a better fit by drawing many samples from the model until it reflects the real-world model. They are easy to configure as they are already defined within the TensorFlow (and other libraries) however they require much longer training time (a day in comparison to VI five minutes).

Alternative consideration. Gaussian processes (GPs) [16] are also a popular choice for time-series analysis [16, 17]. GPs take in a kernel choice as prior (expected behaviour of the data) and for time series a periodic kernel is often chosen. They are non-parametric methods that aim to capture the shape of the data by fitting all features in a covariance matrix and performing regression over them. They are probabilistic methods, so they also provide uncertainty over their output and are much simpler to construct than the STS model described. However, GPs have certain limitations that stop them from being applied in this DSG: GPs have a complexity of $O(N^3)$ where N is the data * features due to the Cholesky decomposition used to calculate the inverse of the covariance matrix and multiply it by the feature weights. There are several optimisations to GPs that allow them to scale, however fitting them to the large dataset we have is still challenging. GPs assumes all features converge towards a Gaussian distribution at the limit and does not handle categorical data or unexplained jumps in the data easily. While in our STS model we simplified this by taking out categorical data, if we had additional time we could encode a categorical distribution transformation that allowed the STS to work with these features, however, there there is no straightforward way of doing this in a GP. With these two limitations, we decided to focus our Whitebox modelling effort on the STS model.

6.4 Explanation Methods

Within Section 3 we motivate the importance of providing explanations for predictions made in this context. Here we explore SHapley Additive exPlanations (SHAP) in Section 6.4.1, Local Interpretable Model-Agnostic Explanations in Section 6.4.2 and Counterfactual Explanations in 6.4.3.

6.4.1 Shapley Additive Explanations

SHAP (SHapley Additive exPlanations) is a game-theoretic feature importance approach. It is better than the conventional feature importance methods provided with the machine learning models. SHAP overcomes a huge drawback associated with the conventional methods which do not provide the direction of impact associated with each feature, by illustrating the direction of impact. This is very important for explaining the output of any machine learning model. It is a method to explain individual predictions of the model, aggregate them together for each feature and quantify them by the means of optimal credit allocation with local explanations using the classic Shapley values.

This study used TreeShAP, a variant of SHAP for tree-based machine learning models such as decision trees, random forests, gradient boosted trees and XGBoost. The advantage of SHAP is its ability to compute Shapley values for all the data points and produce global model interpretations for the model predictions. The current SHAP was run on the XGBoost model, which gave an R^2 score of 0.26 (ref XGBoost table)

Summary Plot

Figure 18 is a summary plot that we have created which helps in interpreting model feature impact on the target variable aggregated over the whole data set. The summary plot combines feature importance with feature effects. Each point on the summary plot is a Shapley value for a feature and an instance. Characteristics of the summary plot:-

- The features are ranked in the decreasing order of importance.
- The graph appears to have a lot of dots for each feature. Each dot

represents a single data point for the feature.

- The colour of the point represents the value of the feature (blue - low value and red/pink - high value).
- The points which are found on the right side of 0 SHAP value point have a positive impact on the predictions and the features on the left side have a negative impact.
- The position of the point on the x-axis relates to the degree of impact of that point on the predicted value.

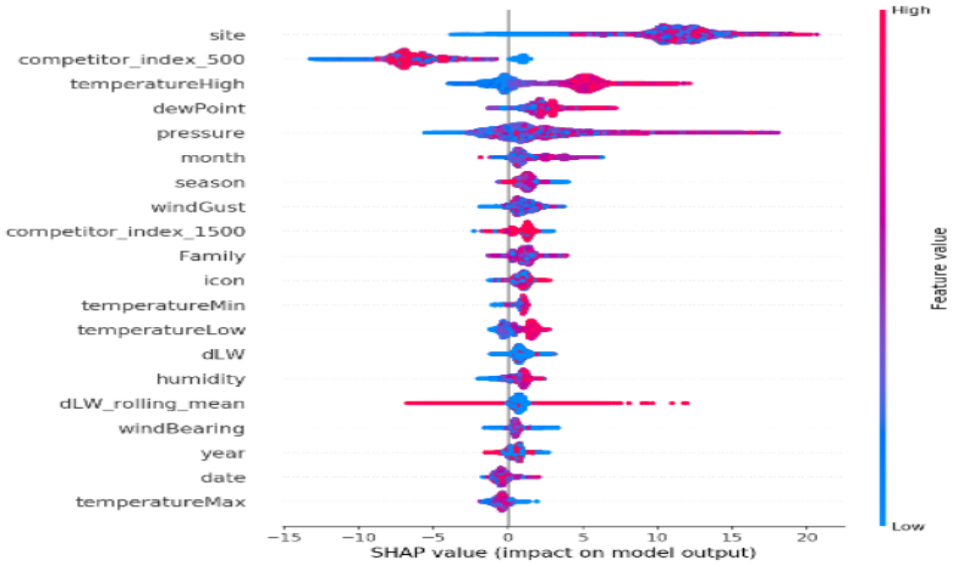

Figure 18: Shap Values for the most important features in the XGBoost Model

Inferences from the SHAP summary plot The location of the store is the most important feature in the prediction of the sales for the model. The number of competitors within the 500 m radius is also impactful but in the opposite direction. Higher number of competitors in the same area discourages sales. Among the weather variables temperature really stood out. Low temperature values (expressed in blue) tend to have a negative impact on the sales which is in line with the EDA that we performed. Low Pressure is also not preferable while moderate pressure values have a

positive effect on the sales.

Force Plot

Figure 19 is a ”force plot” showing the impact of individual features on a single prediction. In this case, we can see that high temperature and low competitor index (1500) bring the score up (above the base value - the average value across all the data points-) suggesting that they have a positive impact on sales, whereas a competitor index (500) of 20 brings the score down suggesting a negative impact.

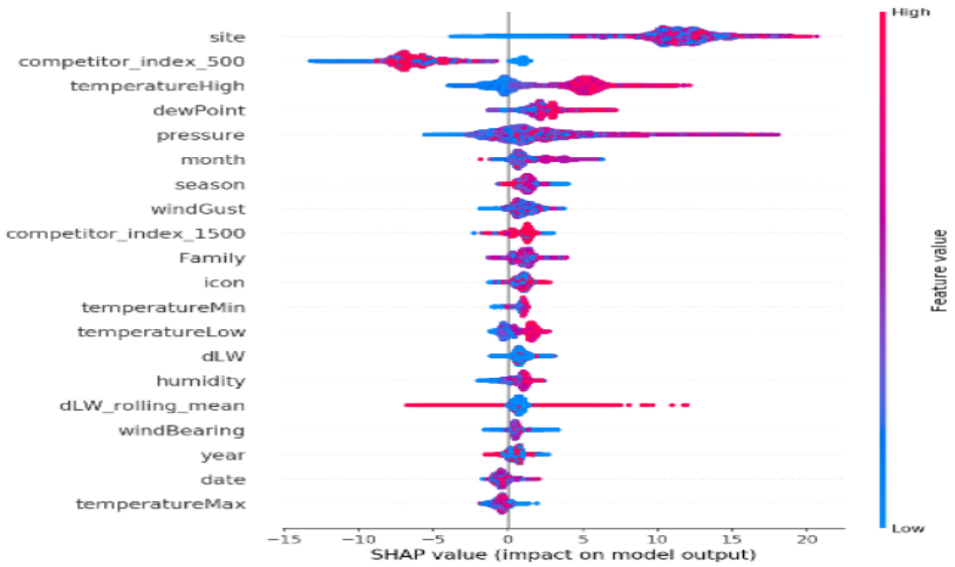

Figure 19: Force Plot for the first entry using SHAP

6.4.2 LIME

Lime (Local Interpretable Model-Agnostic Explanations) is a tool often used to explain model predictions. In this challenge we used it to interpret the XGB model’s prediction when the ‘sales’ variable is equal to 0. Below, three “Explainer Correlation Plots” show the features that are positively or negatively correlated to zero sales. Three data points corresponding to the same outcome were chosen to evaluate explanation consistency. As shown in the graphs, dlW, is_sunday and dlw_to_yesterday_sum are

negatively correlated with sales. TemperatureMax, on the other hand, seems to be positively correlated to sales.

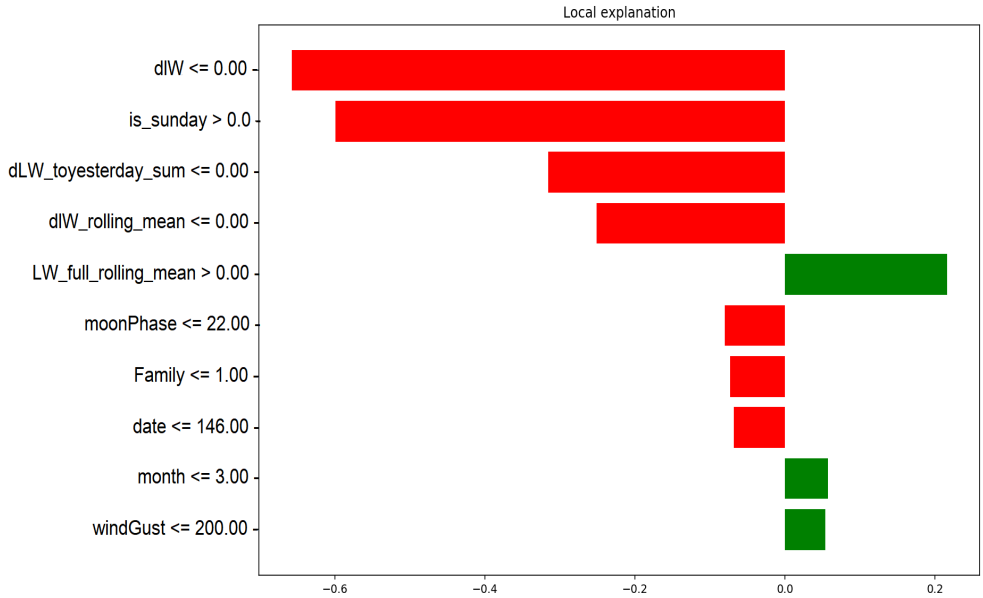

Figure 20: LIME explanatory plot for the first entry for XGB model's prediction of zero sales.

6.4.3 Counterfactual Explanations

Counterfactuals in the context of regression: an open research problem: When considering counterfactual explanations of the form "If X had not happened then Y would not have happened", it is clear that they are more aligned with classification tasks, "If X had not happened then Y would have been predicted by a different class". For regression, where the target variable is continuous, the application of counterfactual explanation is more ambiguous. Primarily, a concern is the infinite number of alternate worlds where the event Y (the predicted target) had not happened. For this reason most state-of-the art approaches to counterfactual explanations for black-box models are only applicable to classification problems [1].

From Regression to Classification: For this reason, we chose to bin our target variable *predicted sales* into two discrete classes, *no sales* and

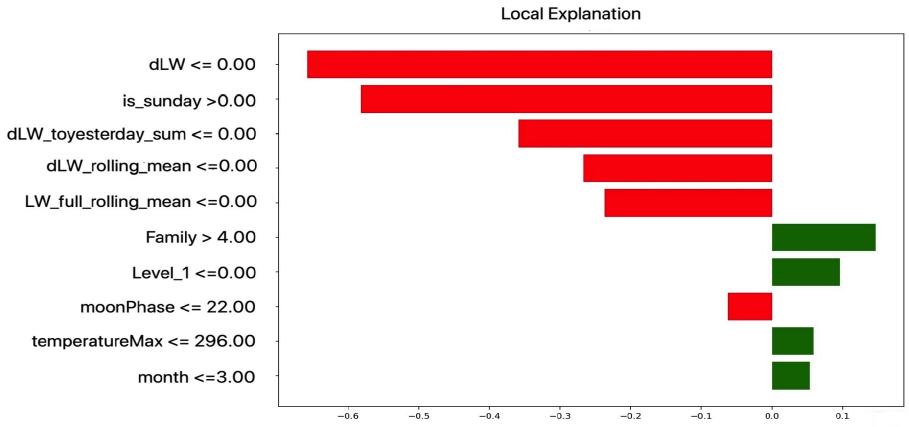

Figure 21: LIME explanatory plot for the second entry for XGB model's prediction of zero sales.

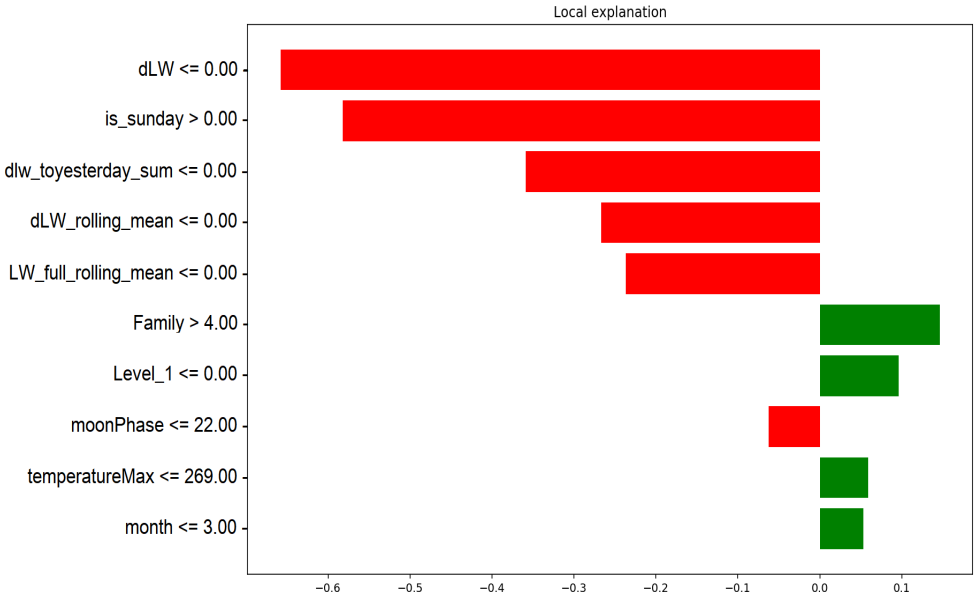

Figure 22: LIME explanatory plot for the third entry for XGB model's prediction of zero sales.

Gaussian Naive Bayes			
	Precision	Recall	F1-score
Accuracy			0.84
Macro Average	0.63	0.52	0.51
Weighted Average	0.78	0.84	0.78

Random Forest Classifier			
	Precision	Recall	F1-score
Accuracy			0.84
Macro Average	0.80	0.51	0.48
Weighted Average	0.83	0.84	0.78

sales, transforming the regression problem into a binary classification challenge. The distribution of the target, particularly the fact that 83.61% of sales were zero, meant choosing an alternative binning strategy (e.g. multiple classes or *low sales* and *high sales* to be infeasible as this would result in highly unbalanced classes.

Preprocessing the data The feature set selected and preprocessing steps are consistent with those described in section 5.1

Unbalanced dataset and classifier choice: Due to the large number of zero sales, our two classes were heavily imbalanced. This had consequences for both our choice of classification model and evaluation metric. We chose to use a Random Forest Classifier and Gaussian Bayes Classifier. Random Forests are robust to outliers and can handle missing values, however, they have a high complexity due to the creation of a large number of trees. The Gaussian Naive Bayes algorithm works well with high dimensional data and is relatively fast to compute, however it operates under the assumption that all features are independent. The models generated the performance illustrated in the above tables.

Future work would involve experimenting with alternative classifiers, particularly exploring which models work best with imbalance data, however, as the focus of this part of the project was on generating counterfactual explanations we arbitrarily selected the Gaussian Naive Bayes Classifier to be used as the black box model to be explained.

Counterfactual Explanations To generate counterfactual explanations,

we used the implementation from the fat-forensics library [18]. The algorithm finds counterfactual explanations for the prediction of an arbitrary black box model for a given instance by using brute-force grid search with a specified step size. Due to the high variance in our data we selected the step size to be 10. The algorithm works to find the nearest instances to the given example that are classified as the opposite class.

The counterfactual explanations given for a given classified instance are composed of several parts: A list of the nearest counterfactual neighbours (instances assigned a different class to the example being explained) The Euclidean distance between the example to be explained and each of its neighbours The predicted class of each nearest counterfactual neighbour The difference between the example to be explained and each of its nearest counterfactual neighbours which translates into the modifications to each feature needed to be made to the example to be explained in order for it to be classified as the opposite class.

An example counterfactual explanation for a given test point is given by:

*Explaining data point (index 0) of class *no_sales* (class index 0)*

*Counterfactual instance (of class *sales*): Distance: 11*
*feature *dLW*: *0* $\rightarrow$ *10**
*feature *is sunday*: *1* $\rightarrow$ *0**

*Counterfactual instance (of class *sales*): Distance: 21*
*feature *dLW to yesterday_sum*: *0* $\rightarrow$ *20**
*feature *is sunday*: *1* $\rightarrow$ *0**

*Counterfactual instance (of class *sales*): Distance: 40*
*feature *dLW*: *0* $\rightarrow$ *40**

The example above generates an explanation for the test point predicted as *no sales*. The generated explanation consists of the three closest nearest counterfactual nearest neighbours, at a Euclidean distance of 11, 21 and 40 respectively from the example to be explained (calculated across all features). The explanation generated by the first counterfactual

neighbour can be interpreted as: If the example to be explained had 10 sales on this day last week instead of 0 and it wasn't a Sunday, it would have been classified as *sales*. Discussions with domain experts suggest such an insight is instructive in communicating the factors impacting on this prediction.

Evaluating Counterfactuals The counterfactuals above provide an intuitive explanation for a particular example. In order to provide some insight into the validity of our counterfactuals we chose to evaluate by comparing the explanations generated by the algorithm for two similar test points where we would hope the counterfactuals generated would also be similar. We chose two test points that were identical in all but three feature values *Name*, *UnitsPerOrder* and *Family* with a Euclidean distance of 11. The top 10 nearest counterfactual neighbours generated for each instance were identical. This is reassuring as an explanation mechanism that generates similar explanations for similar test points implies robustness.

7 Future Work and Research Avenues

We consider several avenues for future work. These include the expansion of techniques experimented with in this project, as well as implementing new methods.

Data. Due to the seasonal nature of sales, data over longer periods would be highly beneficial for the machine learning models to learn any inherent trends. A dataset encompassing similar seasons/weather conditions over multiple years will provide better insights. Additional features pertaining to customer behaviour, such as end-user sales data or location of stores, would be valuable in future data collection exercises. Similarly, knowledge of characteristics related to competitors within the explored competitor index, such as information if the competitor is an independent bakery or a chain, may influence sales in the site.

Modeling Sales. It would be interesting to apply reinforcement learning or agent-based modelling to replicate human behaviour given certain weather conditions. We could also frame the problem as a time series task which would consider shorter spans of time to account for future

sales. Apart from this, we suggest the inclusion of causal inference approaches, like Directed Acyclic Graphs (DAGs). DAGs are a subset of graphical causal models that are used to illustrate the causal relationships between an exposure, outcome and confounders. All arrows of a DAG are unidirectional (thus, “directed”) and the variables cannot cause themselves (thus, “acyclic”) [19]. In a few words, a DAG is a graph that allows researchers to verify the causal assumptions of the data generating process.

Explaining Models. To improve the explainability aspects of this project, we suggest gathering a better understanding of the end-user requirement. We think the feasibility of the explanation plays a critical role since not all features are under human control. For example, a change in weather condition cannot be controlled; however, a suggestion to consider sales on a Sunday may be more feasible for the client to incorporate. There is also scope to apply counterfactual software such as DiCE (Diverse Counterfactual Explanations), which may improve the recommendations obtained.

Evaluating explanations. Our analysis puts forth explanations from several interpretability methods and our initial experiments confirm their validity. Beyond our work, the success and efficacy of explanations are best understood through the lens of the various stakeholders. This includes the model developers and the clients who hope to benefit from model predictions. While the former would find explanations useful in confirming their hypotheses about their models, the latter might use them to assess the reliability of the models’ predictions. To that end, explanations generated for mode predictions, both whitebox and blackbox, should be evaluated based on their effectiveness to developers and clients. Future work can look at two avenues for evaluation: quantitative and qualitative.

1. **Quantitative evaluation - Intermethod agreement.** Future work can extend our existing comparison of explanations for similar data points. Further experiments with explanation on randomly stratified data can be used to gauge whether the same and different methods are consistent. Using interrater reliability metrics like Cohen’s kappa can shed light on the relative performance of different explanation methods.

2. **Qualitative Evaluation - Agreement with human explanations.** Previous work has compared explanations from humans with machine-generated explanations, finding that the two often do not converge. Future work can explore which feature humans find most informative and compare that to machine-generated explanations. Finally, one can also assess the efficacy of explanations based on how helpful they are to clients. Two types of user studies can be undertaken:

   Before/after test. Users would be asked about their user experience before and after having access to explanations.

   Treatment-control test. Similar to a randomized control trial, clients can be randomly divided into treatment and control groups such that both groups have a similar distribution of characteristics. The treatment group would have access to explanations while the control group would not. Statistical tests can then be conducted to

Impact of COVID-19. The predictive models developed in this report by The Alan Turing Data Study Group were based on data for 2018 and 2019 before the outbreak of a novel coronavirus that brought cities to a halt in 2020. After lockdowns in countries worldwide, businesses, such as bakeries and coffee shops, were forced to close and move their operations online or await the end of these forced closures. After many long months, most businesses have been able to resume their operations. However, the nature of their immediate environments have changed, people are less drawn to city centres as they spend their days working from home, avoiding crowded areas, while others have resorted to spending less as they recover from the economic knock of job losses or their businesses closing. Students worldwide have travelled back to their home countries and have not returned to university campuses. The world has changed, and the data we used before 2020 may no longer be applicable. What are the new drivers of sales amidst a global pandemic? Does weather still play a role, and in what sense? Are bakeries likely to see an increase in business again as people start to work wherever they might find a seat, a strong wifi connection and some fresh coffee? Will bakeries be able to measure their sales in new and advanced ways as they sell their products online or via home delivery? Is the necessity of a social media page to market your bakery now more critical than ever? Will

all these factors influence sales or will life in 2022 mirror the sales seen in 2018 or 2019 once again? Only time (and data science) will tell.

8 Team Members

Divya Balasubramanian is an MSc Data Science graduate working as a Data Scientist at NHS England with several years of experience in Data Analytics across different industry segments. She is passionate about application of data science in the healthcare. She completed her dissertation on using one class classification to aid research in administration of anaesthesia for patients with neuro-motor diseases treated using deep brain stimulation. Outside of work, she enjoys mentoring women who are looking to launch their career in technology and finds it is rewarding to see success stories from her mentees.

Kai Hou Yip is a third-year PhD student at the University College London. His research focuses on applying Machine Learning/ Deep Learning techniques to further our understanding of planets outside our solar system.

Indira Sen is a doctoral candidate at GESIS, Leibniz Institute for the Social Sciences. Her research entails understanding and improving measurement of social phenomena from sociotechnical platforms. Through interdisciplinary work spanning mixed method natural language processing and measurement theory, her work focuses on how researchers and policymakers can draw more valid, reliable, transparent, and ethical insights from digital trace data originating in sociotechnical platforms.

Matthew Forshaw is National Skills Lead at The Alan Turing Institute and Senior Lecturer in Data Science at Newcastle University.

Nikita Vala has just completed her MSc. in Big Data & Digital Futures at the University of Warwick. Having left the field of chemical engineering in pursuit of tech, she is now rebranding herself as a data scientist and wants to explore the many applications of deep learning.

Prakhar Rathi is a senior undergraduate student at Shiv Nadar University, India. He is currently pursuing a B.Tech in Computer Science and Engineering with a minor in Economics. He is also a Data Science For Social Good Fellow at the University of Warwick. His research interests include natural language processing, time series forecasting and their intersection with Finance and Business.

Ridda Ali is a second-year (first-generation) PhD student at the Leeds Institute for Data Analytics (LIDA). Ridda graduated with a First-Class Honours degree in Computer Science in 2019 before starting a 4-year ESRC funded integrated PhD and MSc in Data Analytics and Society (Distinction) at the University of Leeds. Her PhD focus is a mix of methodological and applied advances surrounding the understanding of prediction and causal explanation, and their distinct differences in obesity research, with emphasis on accounting for data complexity arising from clustering effects.

Sami Alabed is a first-year PhD student at the Alan Turing Institute and Cambridge university in the system research group looking at interpretable probabilistic models for distributed systems optimisation. He lives to explore all the world food cuisines and is particularly fond of anything that says cheese.

Samuel Edet is a doctoral candidate at KU Leuven, Belgium and is a Research Assistant at IMT School for Advanced Studies Lucca, where he works with the Laboratory for the Analysis of CompleX Economic Systems (AXES).

Sara Masarone is a second-year PhD student at the Alan Turing Institute and Queen Mary University of London. She's interested in applying machine learning to medicine (immunology) to solve real-world problems.

Stephen Kinns is the CEO and co-founder of CatsAi. An explorer in the exciting frontier of predictive analytics and ML, his dream is to make complex ML so affordable that every baker on the high-street can use it to make better decisions.

Tatiana Alvares-Sanches is a postdoctoral research fellow in the GeoData Institute at the University of Southampton. Her main interests are in the application of Machine Learning and AI to urban analytics, geospatial analysis and public health.

Torty Sivill is a first-year PhD student at the Alan Turing Institute and the University of Bristol. Her research is exploring building explainable AI for multimodal data in health with a particular interest in fusing symbolic AI with machine learning. In her spare time, you can find her in the pub.

Nikita Vala has just completed her MScin Big Data & Digital Futures at the University of Warwick. Having left the field of chemical engineering in pursuit of tech, she is now rebranding herself as a data scientist and wants to explore the many applications of deep learning.

Gordon Yip is a third-year PhD student at the University College London. His research focus on applying Machine Learning/Deep Learning techniques to further our understanding of planets outside the solar system.

9 Appendix

9.1 Glossary

Here are the feature names and definitions of the features in our dataset.

site: site key

date: date (d)

productId: product key

sales: actually 'orders'. Those orders taken by the wholesaler from the bakery themselves, not the sales at the bakery to consumers.

dLW: sales this time last week (d-7).

dLW_rolling: average sales for the last two weeks (d-7 + d-14) of the current day of the week.

dLW_toyesterday: total sales for the last week (d-7 to d-1).

Level_1: administrative_area_level_1 indicates a first-order civil entity below the country level. Within the United States, these administrative levels are states. Not all nations exhibit these administrative levels. In

most cases, administrative_area_level_1 short names will closely match ISO 3166-2 subdivisions and other widely circulated lists; however this is not guaranteed.

Level_2: administrative_area_level_2 indicates a second-order civil entity below the country level. Within the United States, these administrative levels are counties. Not all nations exhibit these administrative levels.

Level_3: locality indicates an incorporated city or town political entity.

Name: product name

UnitsPerOrder: the number of units (X) in an order (i.e 1 order of croissants = 1 box of X croissants)

Family: highest level category of the product.

Category: category of the product (i.e. sub-category to Family). competitor_index_500: a count (max 20) of local competitors up to a radius of 500m.

competitor_index_1500: a count (max 20) of local competitors up to a radius of 1500m.

apparentTemperatureHigh: daytime high apparent temperature.

apparentTemperatureHighTime: UNIX time representing when the daytime high apparent temperature occurs.

apparentTemperatureLow: overnight low apparent temperature.

apparentTemperatureLowTime: UNIX time representing when the overnight low apparent temperature occurs.

apparentTemperatureMax: maximum apparent temperature during a given date.

apparentTemperatureMaxTime: UNIX time representing when the maximum apparent temperature during a given date occurs.

apparentTemperatureMin: minimum apparent temperature during a given date.

apparentTemperatureMinTime: UNIX time representing when the minimum apparent temperature during a given date occurs.

cloudCover: percentage of sky occluded by clouds, between 0 and 1, inclusive.

dewPoint: dew point in degrees Fahrenheit.

humidity: relative humidity, between 0 and 1, inclusive. icon: a machine-readable text summary of this data point, suitable for selecting an icon for display. If defined, this property will have one of the following values: clear-day, clear-night, rain, snow, sleet, wind, fog, cloudy, partly-cloudy-day, or partly-cloudy-night.

moonPhase: the fractional part of the lunation number during the given day: a value of 0 corresponds to a new moon, 0.25 to a first quarter moon, 0.5 to a full moon, and 0.75 to a last quarter moon. (The ranges in between these represent waxing crescent, waxing gibbous, waning gibbous, and waning crescent moons, respectively.)

ozone: columnar density of total atmospheric ozone at the given time in Dobson units.

precipAccumulation: the amount of snowfall accumulation expected to occur (over the hour or day, respectively), in inches. (If no snowfall is expected, this property will not be defined.)

precipIntensity: the intensity (in inches of liquid water per hour) of precipitation occurring at the given time. This value is conditional on probability (that is, assuming any precipitation occurs at all).

precipIntensityError: the standard deviation of the distribution of precipIntensity.

precipIntensityMax: the maximum value of precipIntensity during a given day.

precipIntensityMaxTime: UNIX time of when precipIntensityMax occurs during a given day.

precipProbability: probability of precipitation occurring, between 0 and 1, inclusive.

precipType: type of precipitation occurring at the given time. If defined, this property will have one of the following values: ”rain”, ”snow”, or ”sleet”. (If precipIntensity is zero, then this property will not be defined.)

pressure: sea-level air pressure in millibars.

summary: a human-readable text summary of this data point.

sunriseTime: UNIX time of when the sun will rise during a given day.

sunsetTime: UNIX time of when the sun will set during a given day.

temperatureHigh: daytime high temperature.

temperatureHighTime: UNIX time representing when the daytime high temperature occurs.

temperatureLow: overnight low temperature.

temperatureLowTime: UNIX time representing when the overnight low temperature occurs.

temperatureMax: maximum temperature during a given date.

temperatureMaxTime: UNIX time representing when the maximum temperature during a given date occurs.

temperatureMin: minimum temperature during a given date.

temperatureMinTime: UNIX time representing when the minimum temperature during a given date occurs. time: UNIX time at which a data point begins.

uvIndex: UV index

uvIndexTime: UNIX time of when the maximum uvIndex occurs during a given day.

visibility: average visibility in miles, capped at 10 miles.

windBearing: direction that the wind is coming from in degrees, with true north at 0 degrees and progressing clockwise. (If windSpeed is zero, then this value will not be defined.)

windGust: wind gust speed in miles per hour.

windGustTime: time at which the maximum wind gust speed occurs during the day.

windSpeed: wind speed in miles per hour.

9.2 Detailed Descriptions for pre-processing steps

- **Replace with mean values**: Missing values in these columns are treated in three ways:
    1. overall mean: take the mean value of the entire column
    2. grouped mean: take the mean value of the column using only the entries within the same month
    3. K-NN mean: replace the missing value using the mean of the neighbouring 5 days.
- **Date transformation**: Date variable is transformed to ordinal numerical values, using proleptic Gregorian ordinal.
- **Removed time related features**: Any time related features such as TemperatureMinTime is removed as the our research question looks at the sales on a particular day for a particular site, the temporal element of a certain event may not have a big impact on the decision making process.

9.3 Features for building Random Forest Models

The features include: *productId, sales, dLW, dLW_rolling_mean, dLW_toyesterday_sum, LW_full_rolling_mean, Level_2, competitor_index_500, competitor_index_1500, UnitsPerOrder, month, season, icon, precipIntensity, precipProbability, temperatureHigh, temperatureLow, apparentTemperatureHigh,apparentTemperatureLow, dewPoint, humidity, pressure, windSpeed, windGust, windBearing, cloudCover, uvIndex, visibility, ozone, temperatureMin, temperatureMax, apparentTemperatureMin, apparentTemperatureMax, is_sunday, is_holiday*

9.4 Features for Gradient Boosting Regressor (one-hot encoded)

date, dLW, dLW_rolling_mean, dLW_to, yesterday_sum, LW_full_rolling_mean, Level_1, Level_2,competitor_index_500, competitor_index_1500, Name, UnitsPerOrder, Family, year, month,

season, icon, moonPhase, precipIntensity, precipIntensityMax, precipProbability, precipType, temperatureHigh, temperatureLow, dewPoint, humidity, pressure, windSpeed, windGust, windBearing, cloudCover, uvIndex, visibility, ozone, temperatureMin, temperatureMax, is_sunday

9.5 Features for Gradient Boosting Regressor (label encoded)

LW_full_rolling_mean,dLW, dLW_rolling_mean, dLW_toyesterday_sum, is_sunday

9.6 Features used for Gradient Boosting Model in R

date,sales, dLW,dLW_rolling_mean, dLW_toyesterday_sum, LW_full_rolling_mean, competitor_index_500, competitor_index_1500, Name, UnitsPerOrder, Category, year, month, summary, precipIntensityMax, temperatureHigh, temperatureLow, humidity, windSpeed, visibility, temperatureMin, temperatureMax

9.7 Features for MLP

dLW, dLW_rolling_mean, dLW_toyesterday_sum, LW_full_rolling_mean, Level_1, Level_2, competitor_index_500, competitor_index_1500, Name, UnitsPerOrder, Family, year, month, season, icon, moonPhase, precipIntensity, precipIntensityMax, precipProbability, precipType, temperatureHigh, temperatureLow, dewPoint, humidity, pressure, windSpeed,windGust, windBearing, cloudCover, uvIndex, visibility,ozone, temperatureMin, temperatureMax, is_sunday.

9.8 Structured timeseries model components weights

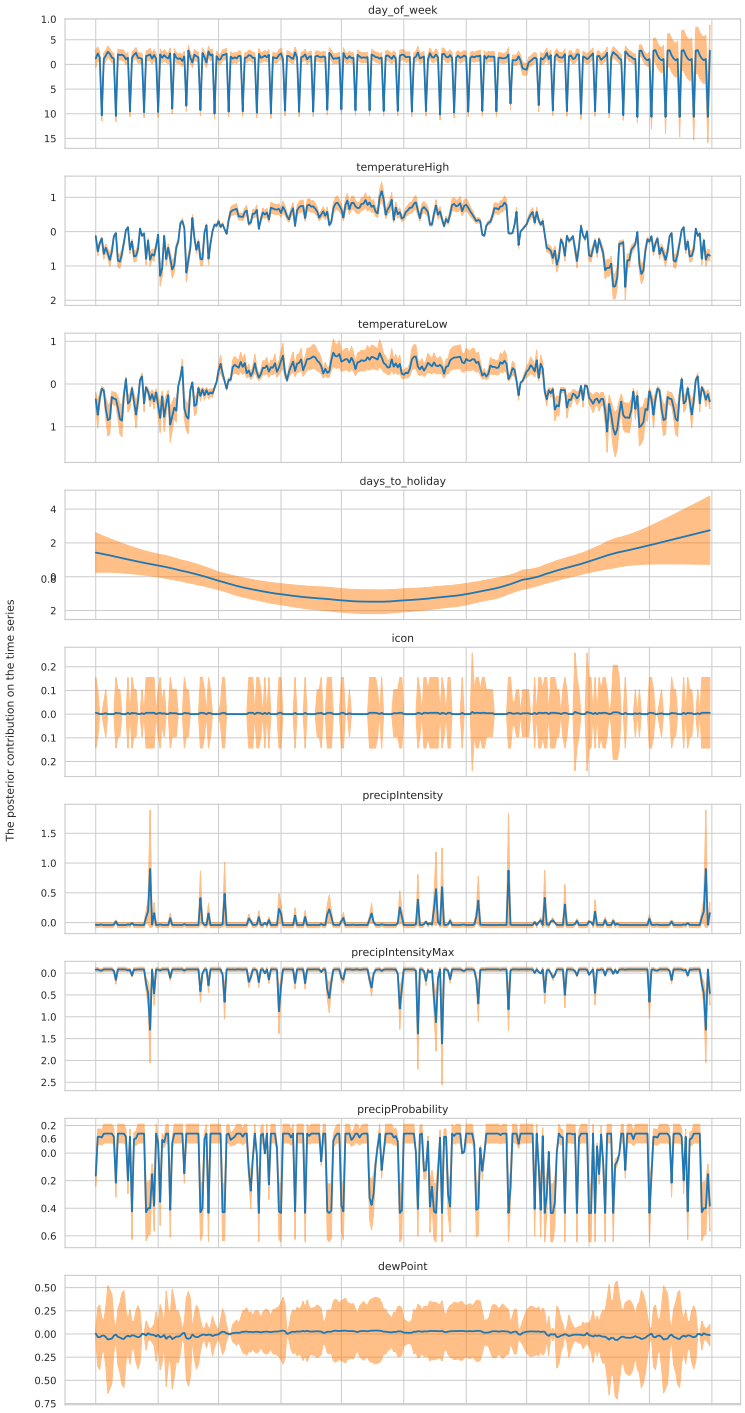

Figure 23: Part A of The posterior distribution of the linear regression weights for every component in the structured time series model. Showing the impact of the feature on the overall prediction.

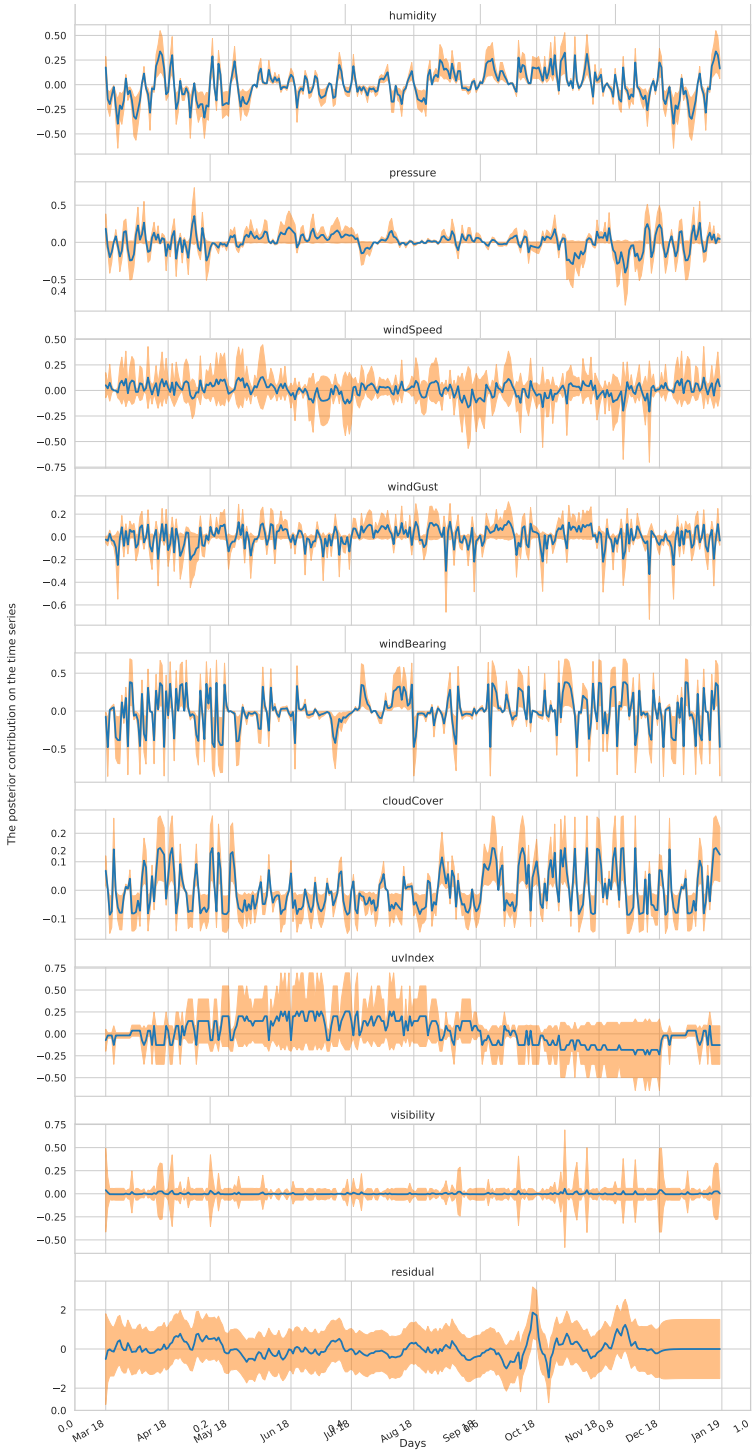

Figure 24: Part B of The posterior distribution of the linear regression weights for every component in the structured time series model. Showing the impact of the feature on the overall prediction.

References

[1] C. Molnar, “Interpretable Machine Learning.” https://christophm.github.io/interpretable-ml-book/, 2020.

[2] C. Molnar, G. König, J. Herbinger, T. Freiesleben, S. Dandl, C. A. Scholbeck, G. Casalicchio, M. Grosse-Wentrup, and B. Bischl, “Pitfalls to avoid when interpreting machine learning models,” *arXiv preprint arXiv:2007.04131*, 2020.

[3] C. Rudin and J. Radin, “Why are we using black box models in AI when we don’t need to? A lesson from an explainable AI competition,” *Harvard Data Science Review*, vol. 1, no. 2, 2019.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, “Model-agnostic interpretability of machine learning,” *arXiv preprint arXiv:1606.05386*, 2016.

[5] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in Neural Information Processing Systems*, pp. 4765–4774, 2017.

[6] A. M. Liebetrau, *Measures of association*, vol. 32. Sage, 1983.

[7] F. R. Bach and M. I. Jordan, “Kernel independent component analysis,” *Journal of machine learning research*, vol. 3, no. Jul, pp. 1–48, 2002.

[8] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, “Measuring statistical dependence with hilbert-schmidt norms,” in *International conference on algorithmic learning theory*, pp. 63–77, Springer, 2005.

[9] S. Wachter, B. Mittelstadt, and C. Russell, “Counterfactual explanations without opening the black box: Automated decisions and the gdpr,” *Harv. JL & Tech.*, vol. 31, p. 841, 2017.

[10] M. A. Hernan and J. M. Robins, *Causal Inference: What If*. CRC Press, 2018.

[11] K. Ito and K. Xiong, “Gaussian filters for nonlinear filtering problems,” *IEEE transactions on automatic control*, vol. 45, no. 5, pp. 910–927, 2000.

[12] P. K. Janert, *Gnuplot in action: understanding data with graphs*. Manning, 2010.

[13] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Advances in neural information processing systems*, pp. 473–479, 1997.

[14] S. L. Scott and H. R. Varian, "Predicting the present with bayesian structural time series," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 5, no. 1-2, pp. 4–23, 2014.

[15] K. H. Brodersen, F. Gallusser, J. Koehler, N. Remy, S. L. Scott, *et al.*, "Inferring causal impact using bayesian structural time-series models," *The Annals of Applied Statistics*, vol. 9, no. 1, pp. 247–274, 2015.

[16] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[17] S. Brahim-Belhouari and A. Bermak, "Gaussian process for nonstationary time series prediction," *Computational Statistics & Data Analysis*, vol. 47, no. 4, pp. 705–712, 2004.

[18] K. Sokol, A. Hepburn, R. Poyiadzi, M. Clifford, R. Santos-Rodriguez, and P. Flach, "FAT Forensics: A Python Toolbox for Implementing and Deploying Fairness, Accountability and Transparency Algorithms in Predictive Systems," *Journal of Open Source Software*, vol. 5, no. 49, p. 1904, 2020.

[19] K. F. Arnold, W. J. Harrison, A. J. Heppenstall, and M. S. Gilthorpe, "Dag-informed regression modelling, agent-based modelling and microsimulation modelling: a critical comparison of methods for causal inference," *International Journal of Epidemiology*, vol. 48, pp. 243–253, 2018.

turing.ac.uk
@turinginst