BMAN73701
SALES FORECASTING

Group Number 20:
9967004, 11356880, 10748942, 10803746, 11462664,10749285

Aims

- ABC prediction of sales
- Overstocking – cost of removal
- Understocking – lost revenue
- ML prediction

Source: Image Creator from Microsoft Designer

Key Topics

Section I: Key Themes from EDA

Section II: Feature Engineering

Section III: Model Selection

Section IV: Model Performance

Section V: Results

Section VI: Next Steps/ Impact

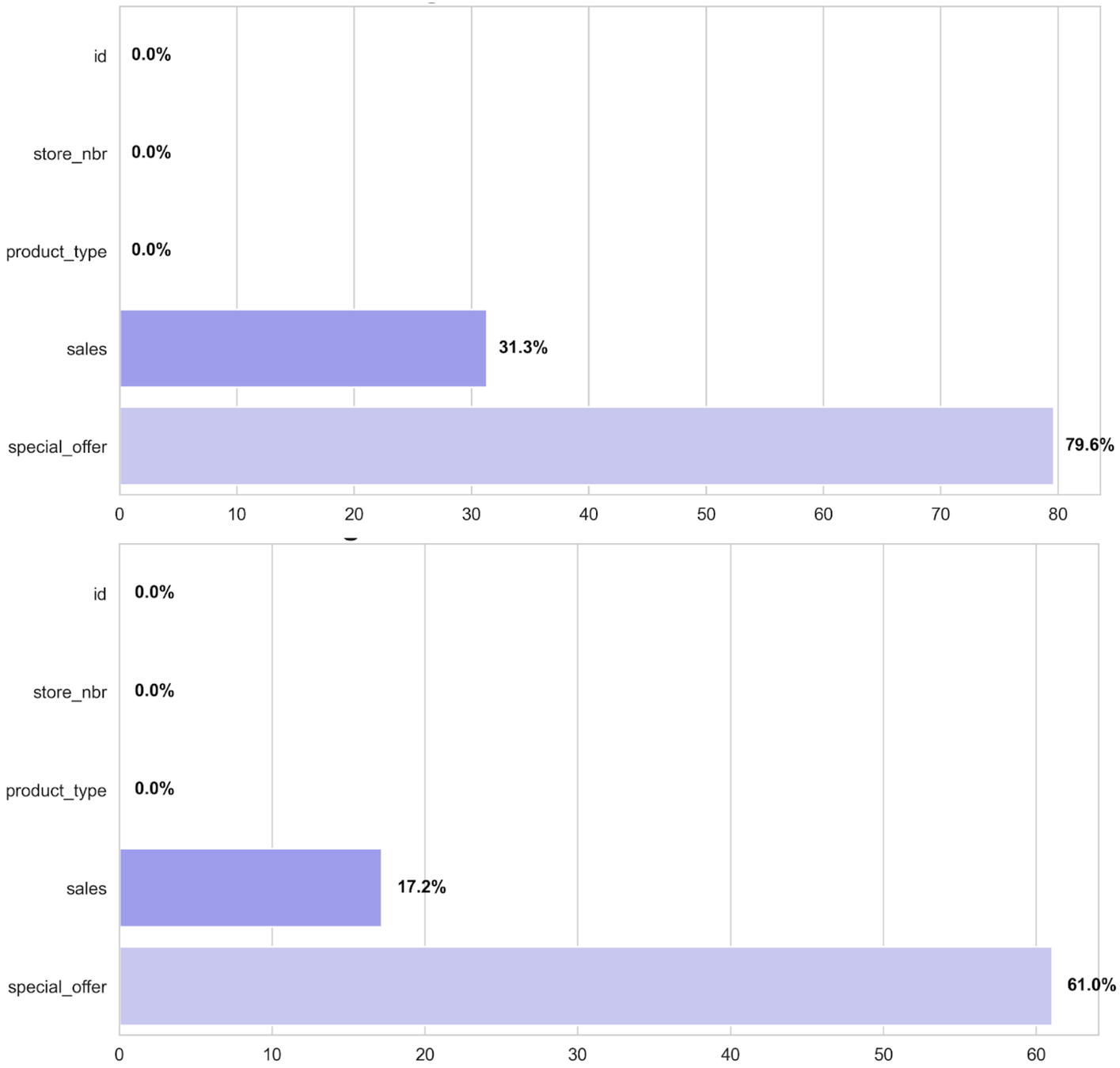
I. Key Themes from EDA

High proportion of 0 values in sales

Were some of the stores closed?

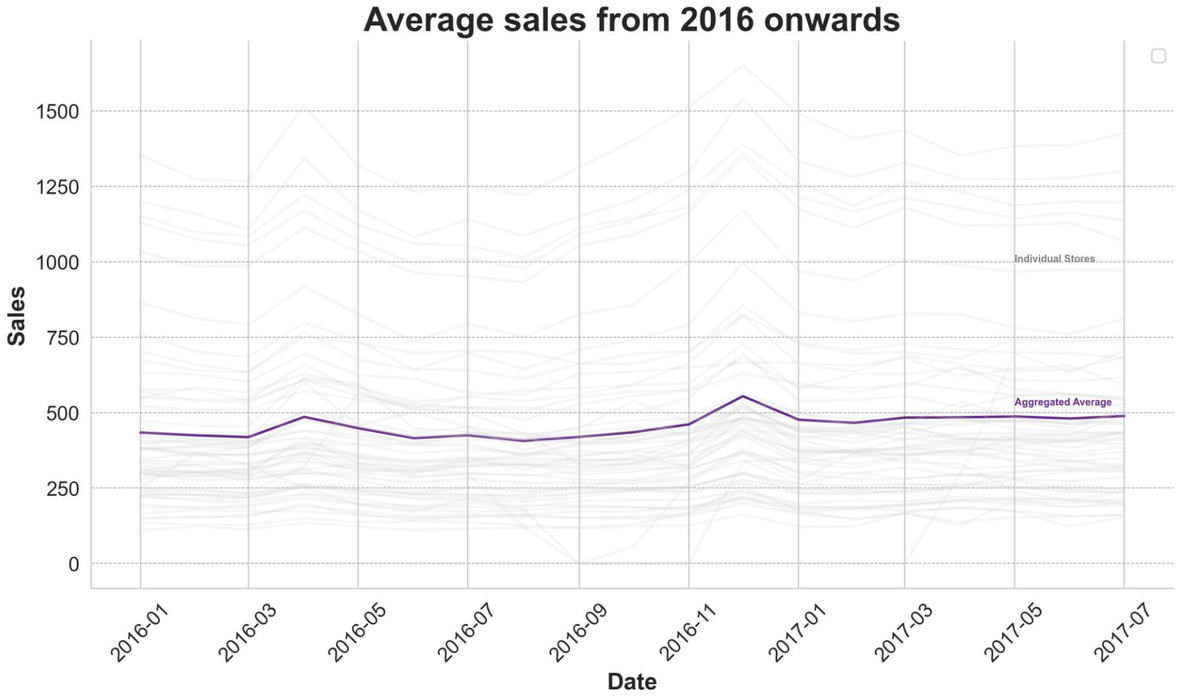
Data from 2016 onwards

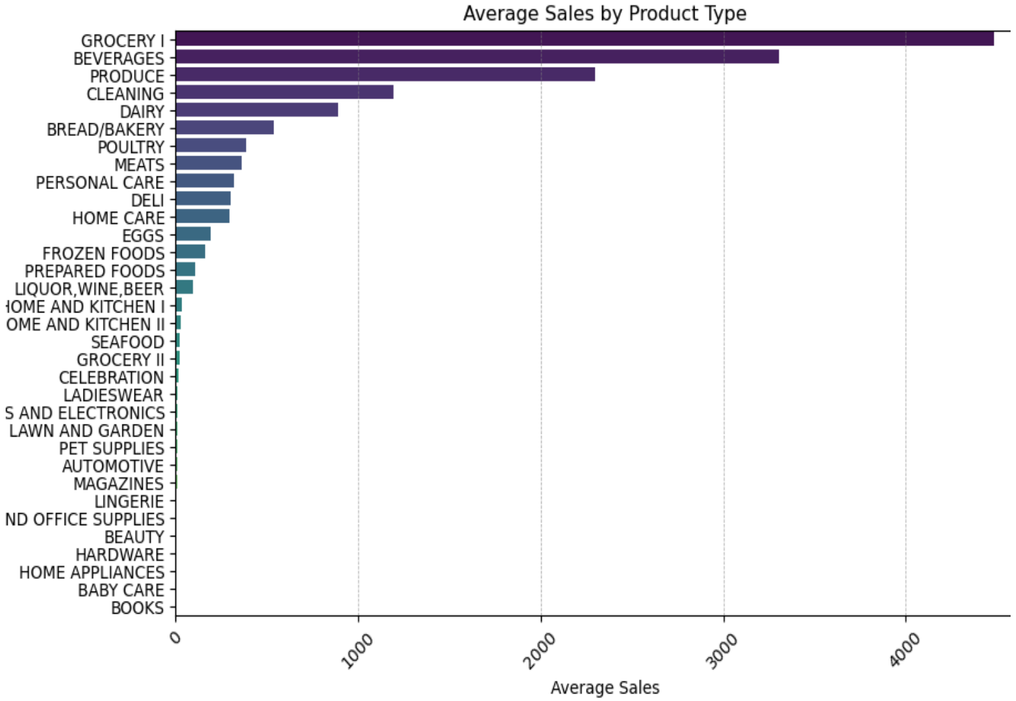
- Most up-to-date
- Reduces 0 values
- 1054944 observations



I. Key Themes from EDA

Strong variability in sales across stores and product types

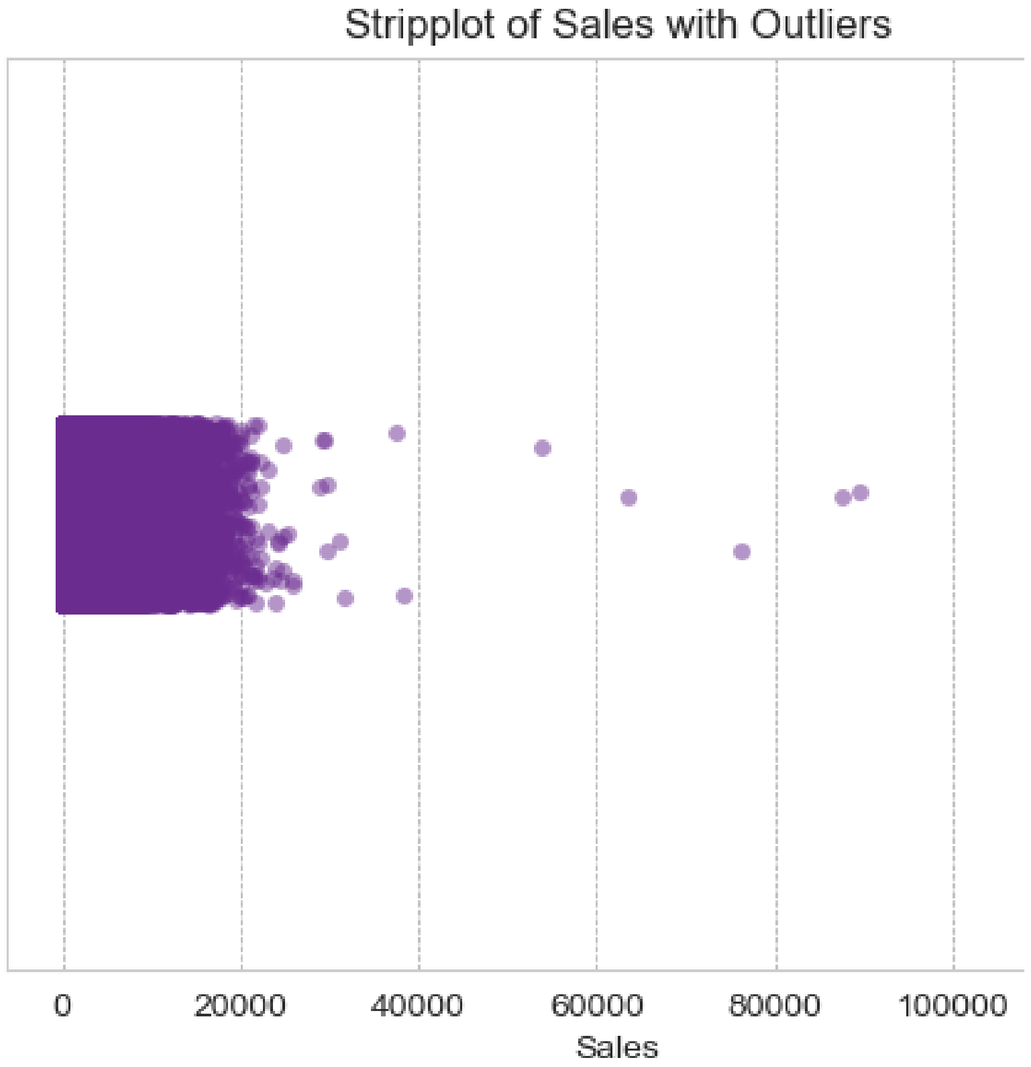




I. Key Themes from EDA

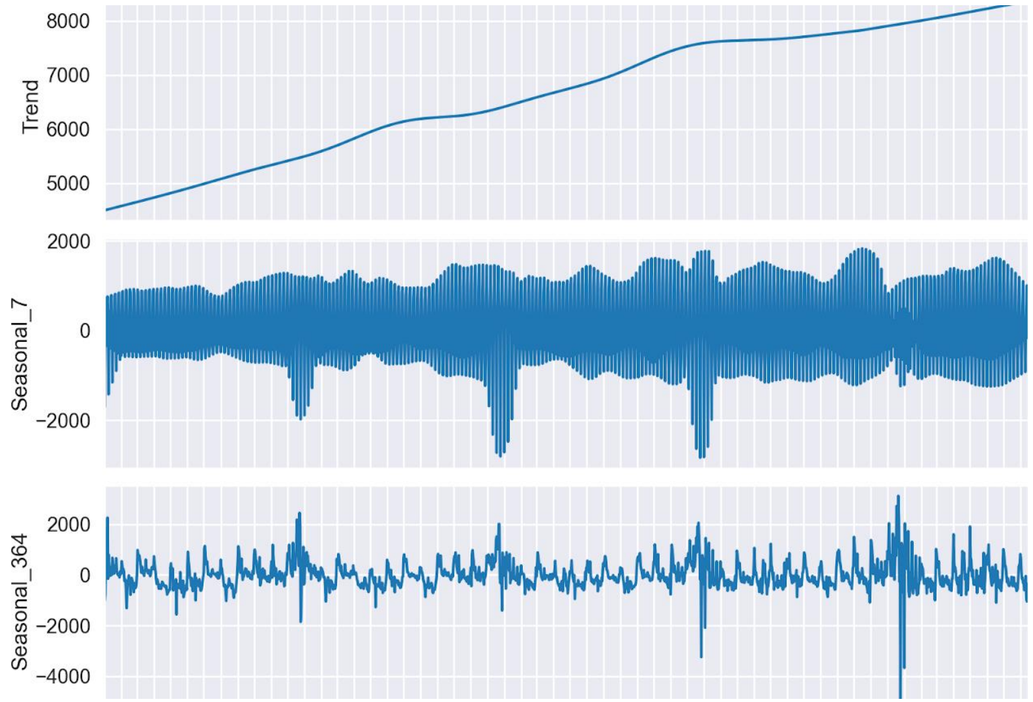
- Significant variance in the data means traditional outlier identification methods are infeasible

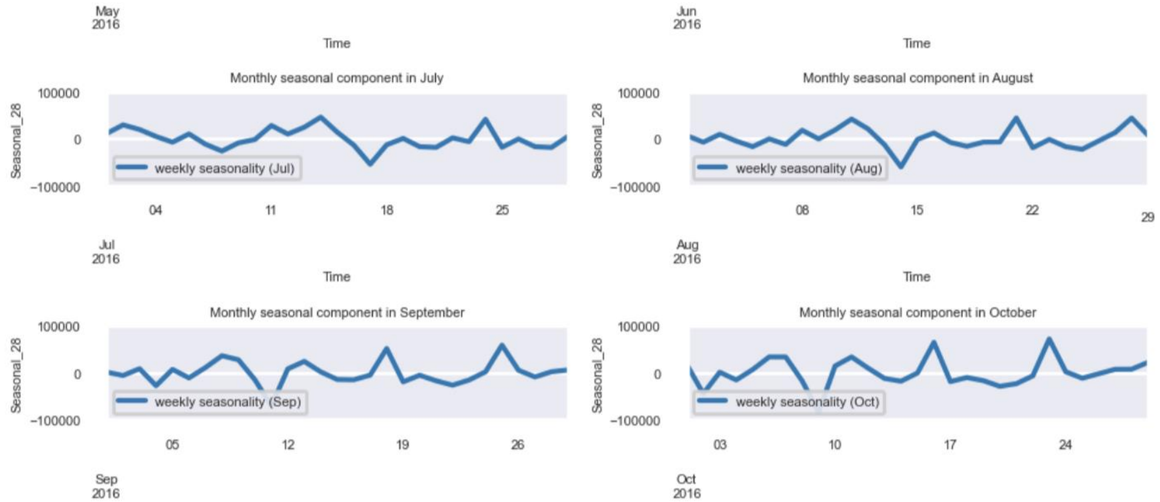
- Only extreme outliers above 40k were removed to keep the variation



I. Key Themes from EDA - Seasonal Decomposition

- Strong growth in sales seen by the positive trend
- Element of weekly seasonality with sales varying on the day and increasing through the week (stronger sales on weekends)
- Drill-down into the monthly seasonal component for every month in 2016 shows a clear seasonal pattern amongst pair of months





II. Feature Engineering

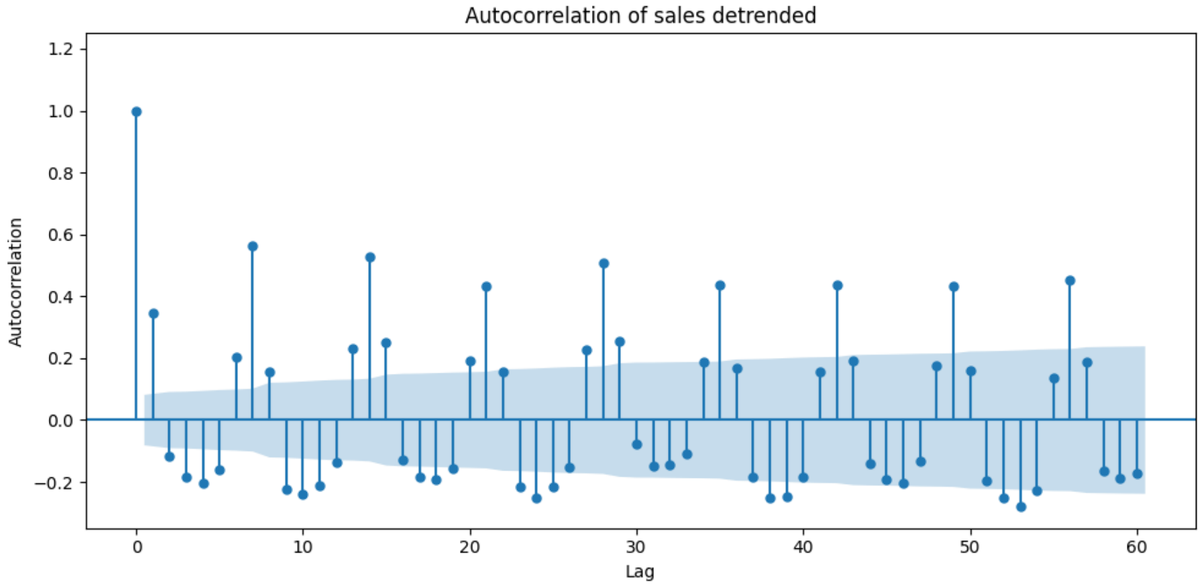
Four categories of new features were created balancing **dimensionality** in feature size and **computational** limitations, whilst capturing **temporal dependencies** in the dataset:

1. Lagged
    - ACF plot shows a statistically significant correlation between current values and past values at 1 and 7.
    - We created a feature for a lag of 7 days (1 day showed high multicollinearity with 7 days)

2. Window
    - We created a 7 day rolling windows for std. dev. and skew to capture variability / volatility over the seasonal period

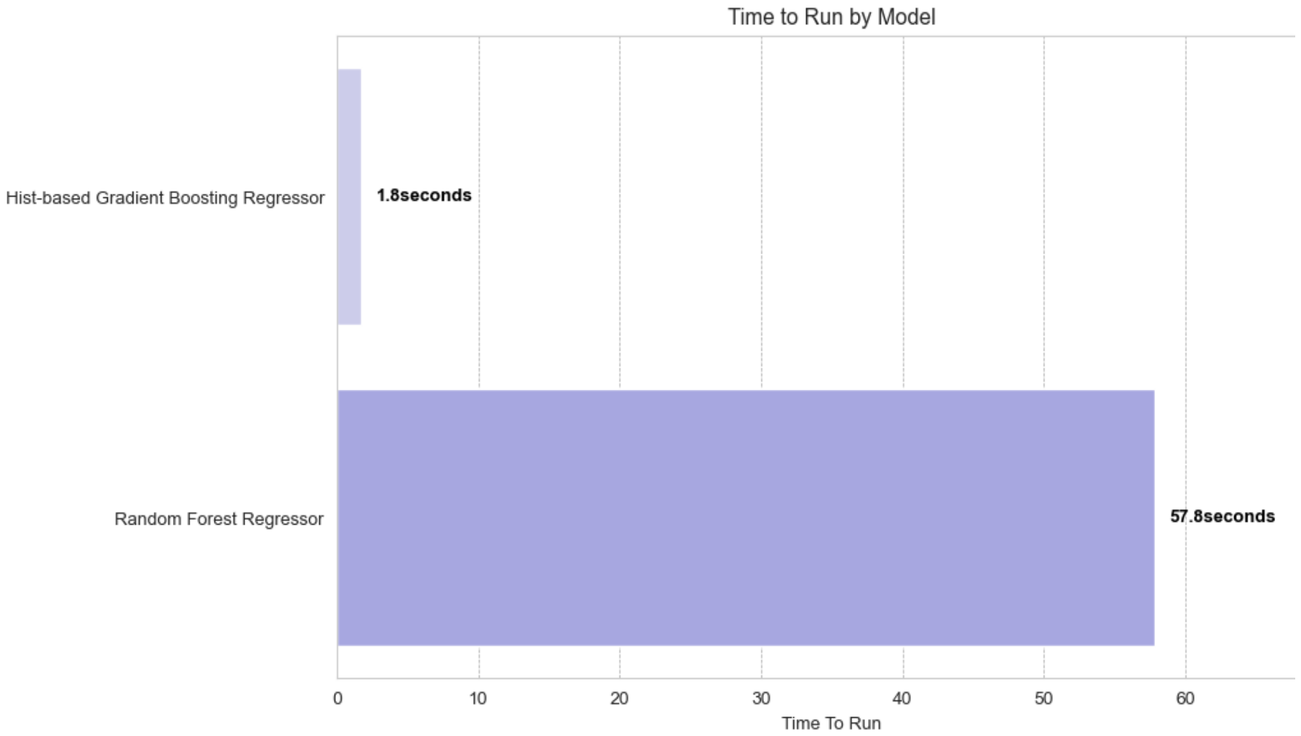
3. Time
    - We created a feature for day of the week, month, and day of the month to capture the relationship between these variables and sales

4. Categorical
    - We converted product type to a categorical variable between 0 and n_classes – 1 using label encoding
    - Store number was treated as a categorical variable by the HistGradboost



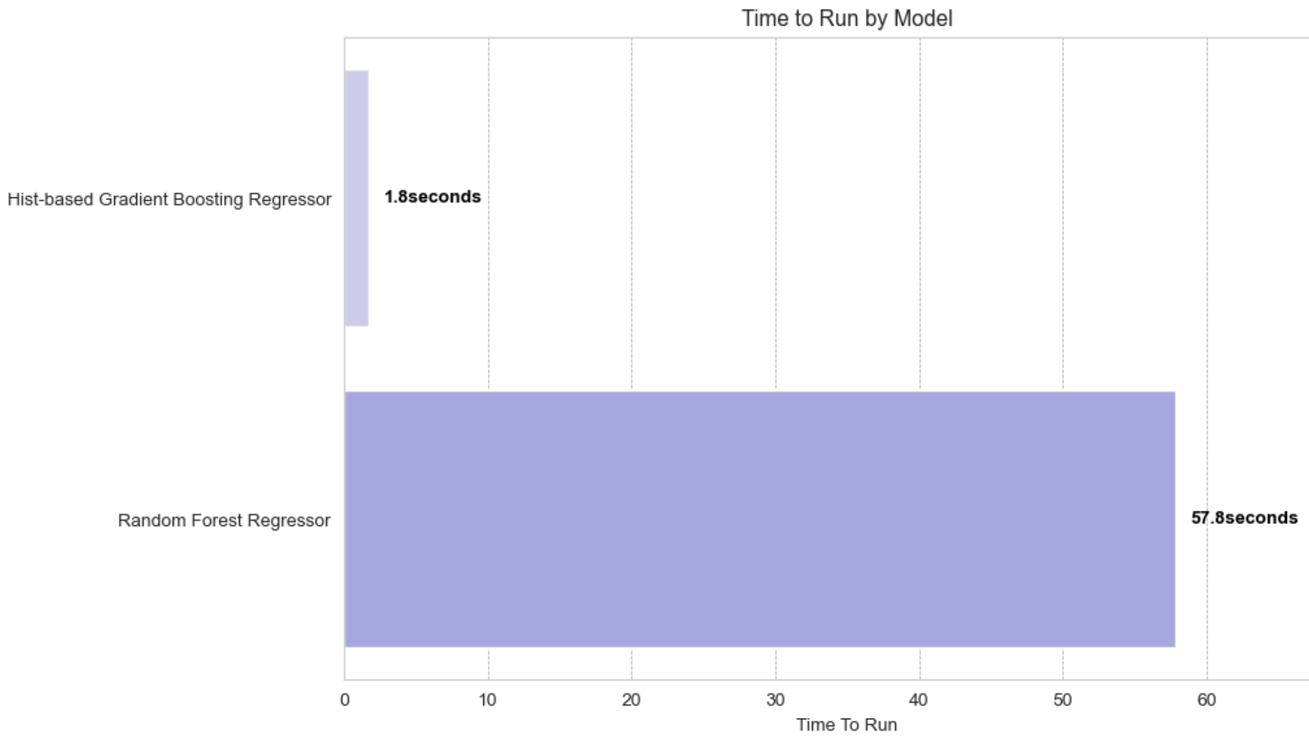
III. Model Selection

- **Linear vs Ensemble models**
  - Linear models like Linear regression can't deal with non-linear data (e.g. seasonality)
  - Ensemble models: handle non-linear data, less overfitting, less multi-collinearity
- **Random Forest Regressor (RFR)**
  - Diverse trees reduce overfitting but computationally intensive model
  - Less influenced by multicollinearity
  - Computationally expensive on large datasets



III. Model Selection

- **Histogram - Based Gradient Boosting Regressor (HGBR)**
  - Addresses shortcomings of RF and is more computationally efficient
  - Each tree is built sequentially leading to a better fit
- **HGBR over RF**
  - More suitable for larger datasets, giving us more capacity for optimisation
  - RMSE: HGBR (245), performs better than Random Forest (272)



IV. Model Evaluation: TimeSeriesSplit

What is TimeSeriesSplit:

- TimeSeriesSplit maintains the temporal order of the data, when performing cross-validation.

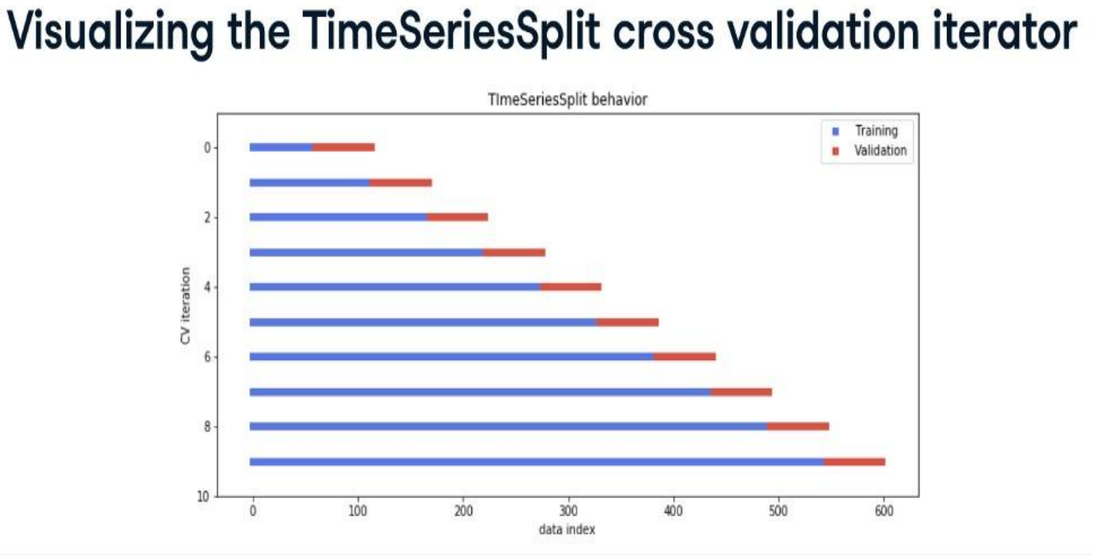
Why is it important:

- Prevents data leakage occurring through look-ahead bias, which occurs when future values are used to predict past values, leading to overly optimistic assessments of model performance.

- consistency of the performance across the folds is indicative of a model that generalises well to a range of time periods

Drawback

- it doesn't use the whole data for each fold for cross validation like train_test_split which is a necessary sacrifice.



Source: DataCamp

IV. Model Evaluation: Performance Metric

Evaluating the performance of our model using the Root Mean Squared Error (RMSE) because:

- Same unit as our target variable
- Metric focuses on minimizing larger errors, which fits the business logic of ABC as larger errors are more disruptive than many smaller errors for a grocery retailer.

IV. Model Evaluation: Hyperparameter Tuning

For our initial HGBRM there were several hyperparameters that we believed could leverage the findings made in our EDA and model selection. The two most important were:

- **Max_depth :**

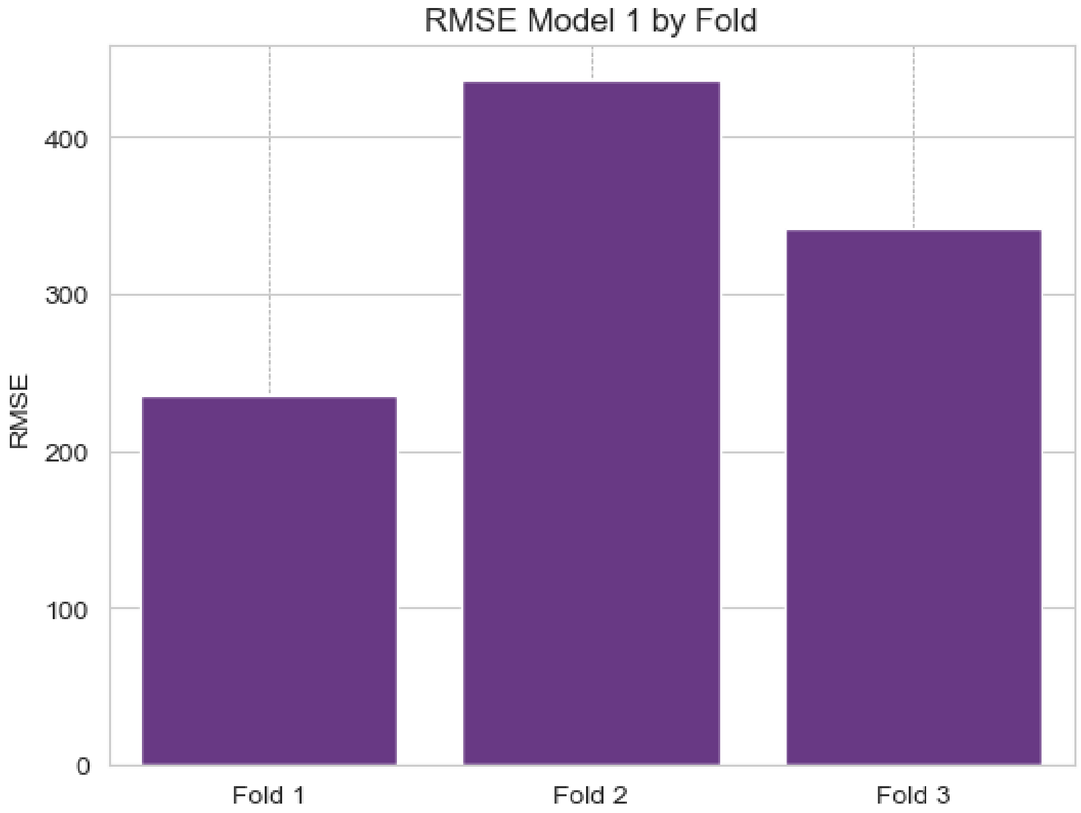
We chose low values for this hyperparameter because as Time Series data tends to be noisy, deep trees are more prone to capture it.

- **Learning_rate**:

To leverage our model's high computational efficiency to find more optimal parameters, we chose a very low range of values.

IV. Model Evaluation: Model 1

- As can be seen in the graph, the performance across the folds is highly variable/less stable
- The runtime of the optimisation is low, it gives us more headroom to add more hyperparameters with greater range of values in our subsequent tuning.




```
Randomized Search took 11.20 seconds.
```

IV. Model Evaluation:
Introducing New Hyperparameters

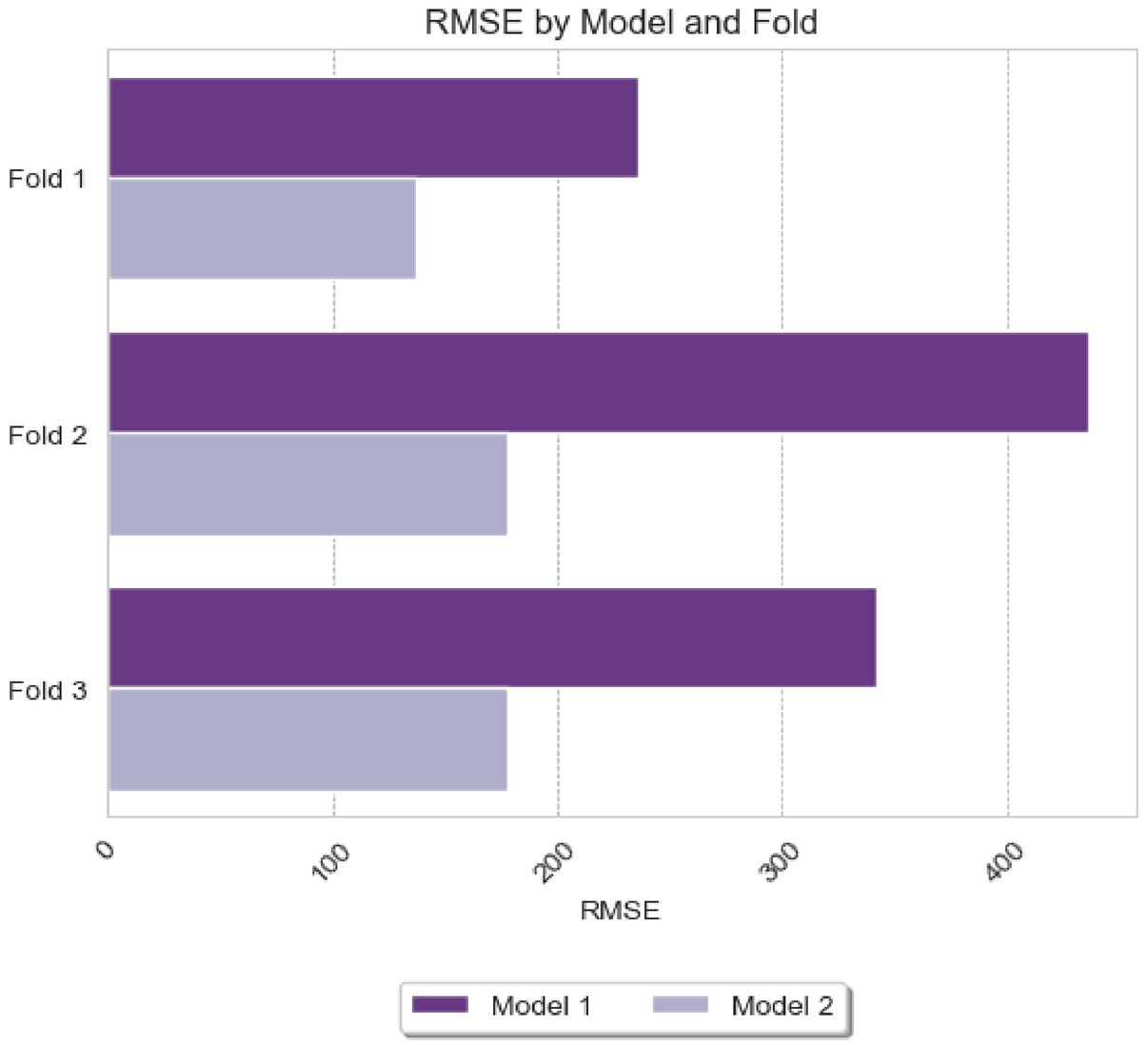
- Improve model's performance across the different folds, but balance between not overfitting and getting stable results.
- Low runtime -> we can increase range of hyperparameter values
- New hyperparameters:
  - **Max_leaf_nodes:** controlling number of leaf nodes in reasonable range enables the model to better segment the time series data and avoid overfitting.
  - **Max_iter:** As model is more resilient to overfitting (l2 reg and less leaf nodes), we increased the number of trees - better at identifying more intricate patterns

IV. Model Evaluation:
Model 2 (Final Model)

- Consistent scores across the folds showing the model has become more generalisable
- Very low runtime as compared to other types of models on this same dataset.

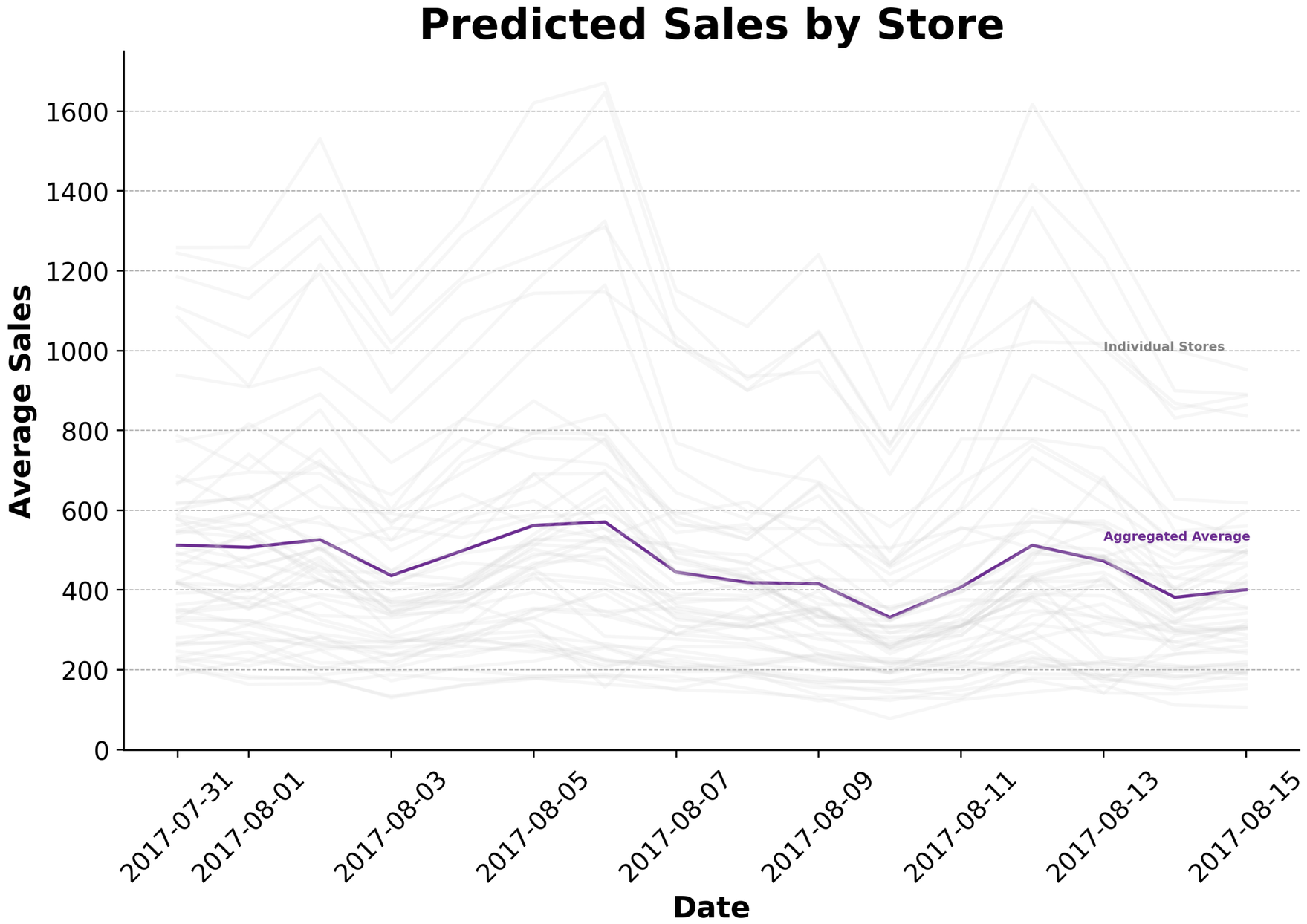
Limitations:

- As we see a pronounced difference in performance between validation and test set, it can be indicative of potential overfitting.



V. Results

Prediction of aggregate sales by store



V. Results

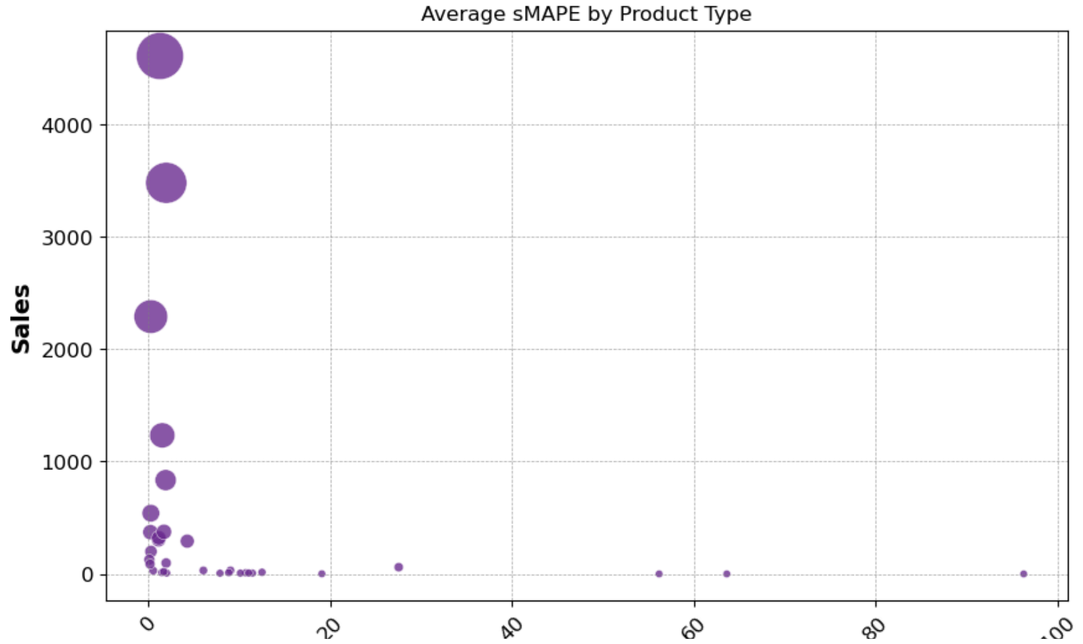
Higher accuracy for best-selling products and stores

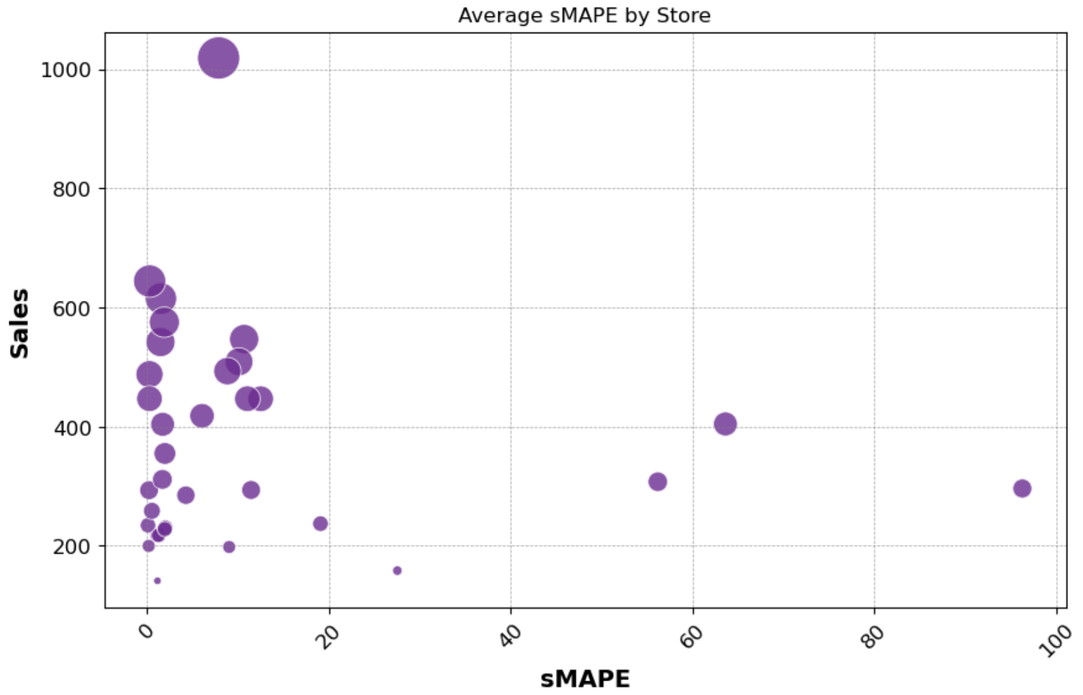
SMAPE as metric

- Benefits ABC as it aids in managing over or understocking and minimising revenue loss

Lower accuracy for low sales

Reduces chance of perishable goods overstock (environmental concerns)





We recommend improving the predictive model through the following:

Challenge		Next Steps
	**Risk of Data Leakage** • Feature engineering required historical data from the training set as features in train and test sets. Therefore, we did not split the data into train and test before engineering. • Concern increased when considering more complex features, such as rolling and expanding windows and transformations such as de-trending the data	**Build a pipeline to** capture pre-processing and feature engineering steps and apply to a separate test and train set
	**Computational Constraints** • Incorporating more than two year's of data would be ideal • Tightly controlling feature engineering, the number of iterations in hyperparameter tuning, and other computationally expensive processes limited our comprehensiveness	**Leverage a cloud environment** to host data and increase computational resources

	**Experimental Constraints**
	• Constants on time and data limited our ability to experiment with the following complex improvements:
	  ○ Explore robust pre-processing techniques to reduce the large variance in the data (i.e., Box Cot, Log transformation)
	  ○ Tree-based models struggle to extrapolate data, and given the positive trend in sales, we can consider detrending sales in pre-processing and add the trend back to the predictions
	  ○ Further explore features that capture the monthly pair-based seasonality and that consider other data points (location of stores, holidays, demand/economic growth indicators)
		Incorporate high-impact changes that consider known patterns in the data and additional data