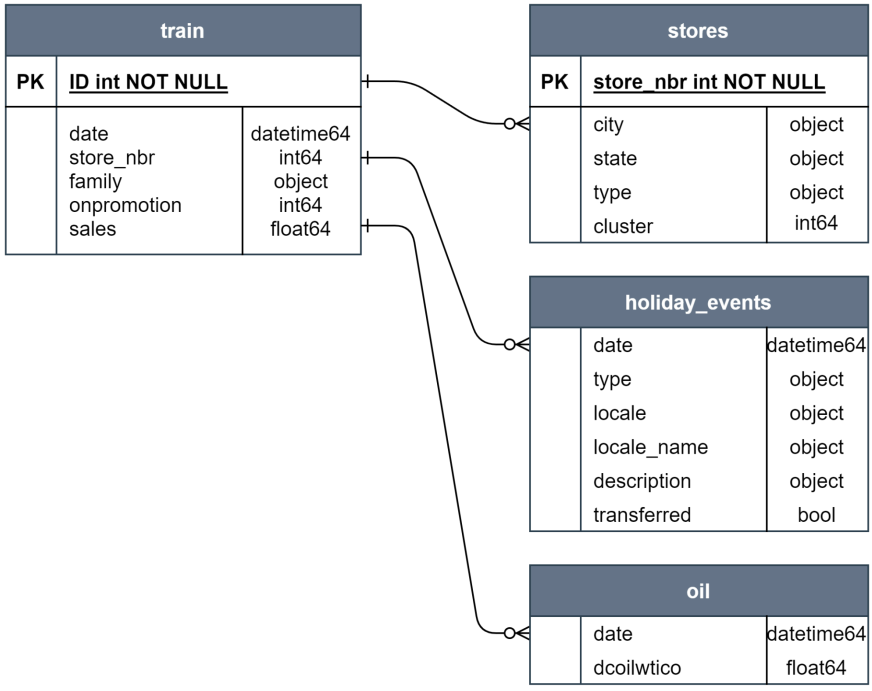
288R Project Progress Report - “Store Sales - Time Series Forecasting”
Project Group #6
Authors: Cory LeRoy, Alec Bothwell, Will Earley

Background:
This project aims to investigate the efficacy of time-series forecasting ML models in mitigating food waste within grocery stores. The imperative of curbing food waste, while ensuring adequate sustenance for individuals, underscores its profound implications for environmental sustainability.

Dataset:
Alexis Cook, DanB, inversion, Ryan Holbrook. (2021). Store Sales - Time Series Forecasting. Kaggle. https://kaggle.com/competitions/store-sales-time-series-forecasting

Data Pipeline and Feature Extraction:

Below is a diagram of the tables used and the columns provided in their raw form before transformation.



Below is the final table used. It includes features used from the original dataset tables and features created with feature engineering in green.
Initial data merging was used to get all fields into 1 table.

final_df		
PK	ID	int
	date	datetime64
	store_nbr	int64
	family	object
	oil_price	float64
	cluster	int64
	isPromoted	boolean
	day_of_week	int64
	month	int64
	year	int64
x21	holiday_dummy	boolean
	rolling_sales_mean	float64
	sales_lag	float64
x4	family_clusters	boolean

X next to the dummy variables indicates how many dummy variables were created.

Existing works were leveraged for some of the extracted features. 'Store Sales TS Forecasting - A Comprehensive Guide'[2], created day_of_week, month, year, and the rolling_sales_mean. The team added the holiday dummies which we shortened using OLS (description in progress report section) and family_clusters. A description of the features is below and more detail on the extracted features is included in the *feature extraction* section later in the paper:

- ID: sales Id as given by train
- Date: date of sales as given by train
- Store_nbr: store id number as given by train
- Family: product category being sold as given by train
- Oil_price: price of oil that day as given by oil
- Cluster: store cluster. Stores with similar sales across their families are clustered together
- isPromoted: will be a boolean simply indicating whether or not there was a promotion on that day for that family of items
- day_of_week, month, and year: will help with leveraging the date as a feature
- Holiday_dummy will indicate whether there was a holiday on that day
- Rolling_sales_mean provides a 7 day rolling mean for the data. As seen later in this report, we implement a lag_1 for the sales_lag to use the previous day to help predict each new day
- The original Family feature is represented by 4 clusters instead of 33 families

Progress Report Details:

Exploratory Data Analysis

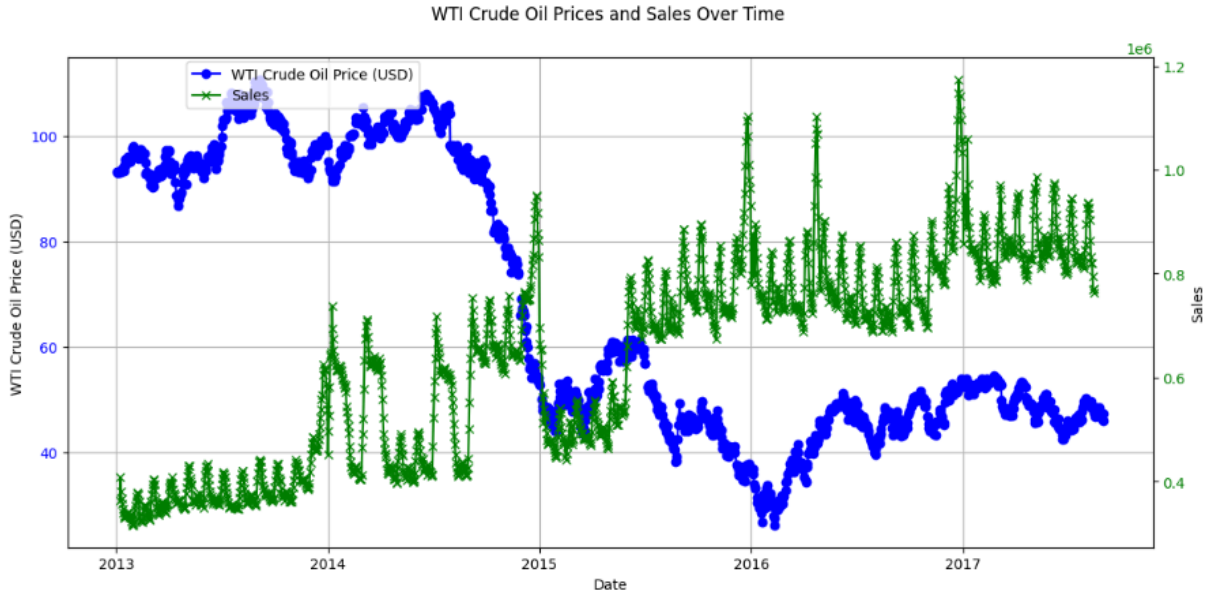
Dataset Collection

Full train dataset, along with support files (oil, holidays, stores, etc.), have been collected and are ready for use in feature extraction as mentioned in *Data Pipeline and Feature Extraction*

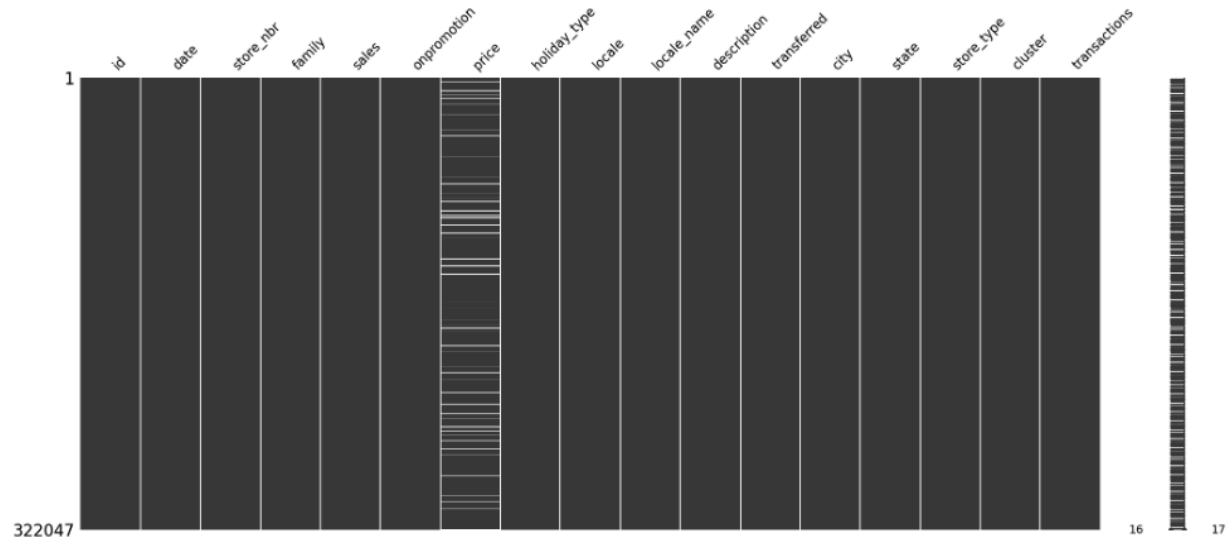
Summary of the steps for features:

Oil:

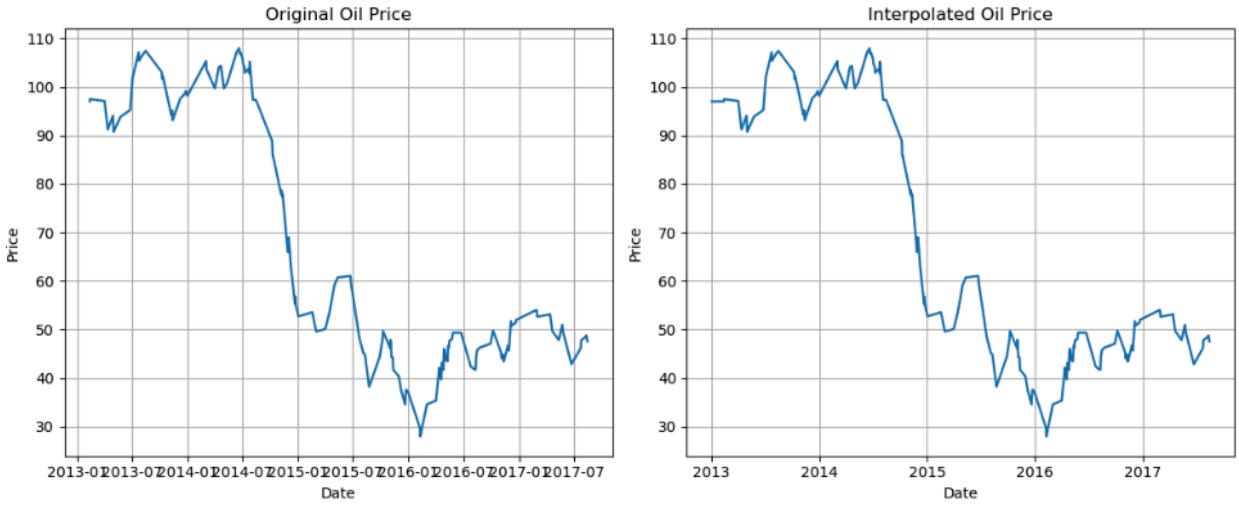
The following graph seems to indicate a possible inverse correlation between oil prices and total sales across all product families. For this reason, we decided to include the oil price as a feature, with the possibility of using it as a feature in our predictive model.



We created a matrix to visualize any missing values, shown below. From this visual as well as other analysis, we knew the ‘price’ column, which represents the price of oil, was missing many values, 22,044 missing values to be exact.



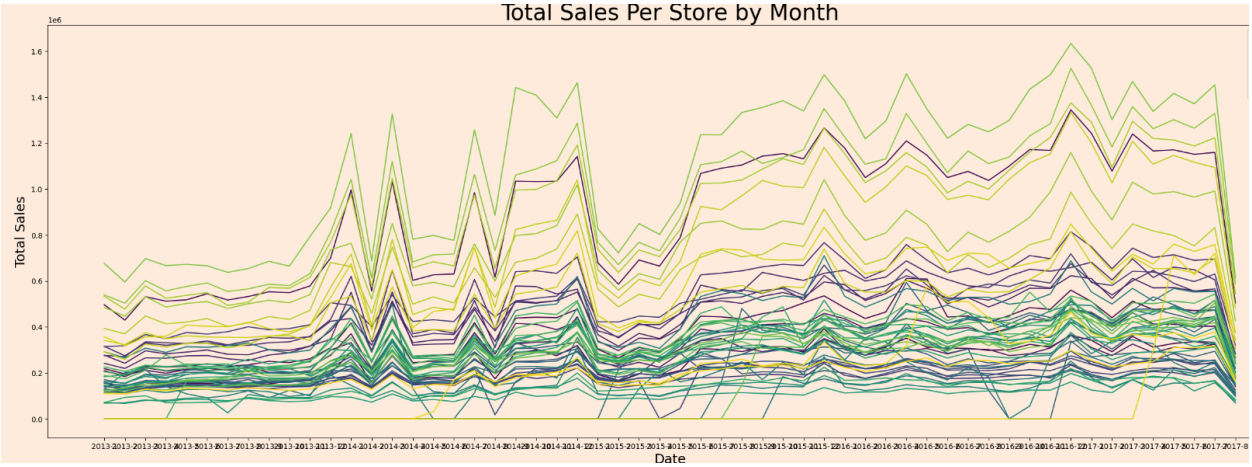
We will be handling these missing values through a process called interpolation. Through interpolation we can fill in these missing values based on known values to create a smoother representation of the data. The original oil price over time graph is shown below next to the graph with interpolation introduced.



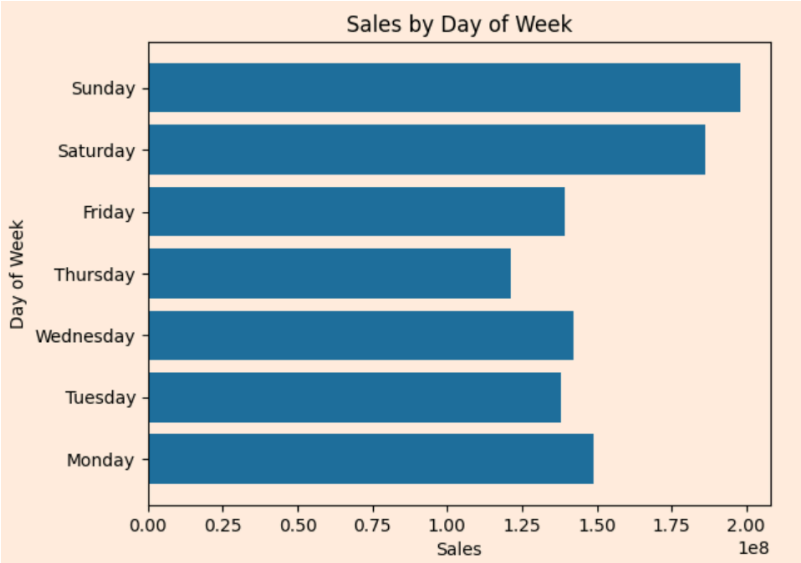
Date[2]:

The date columns in train, test, and holidays were all converted to datetime[64]. The date range of the train data is from 2013-01-01 to 2017-08-15. Every store and family in train has an entry for every date except Christmas, 12-25, of every year. However, not all stores were open the entire date range but rows existed in the dataset which leads to potentially years worth of 0 values. This is shown in the below image which is a plot of each store's daily sales over time. These rows were removed to prevent them from bringing down the sales predictions.

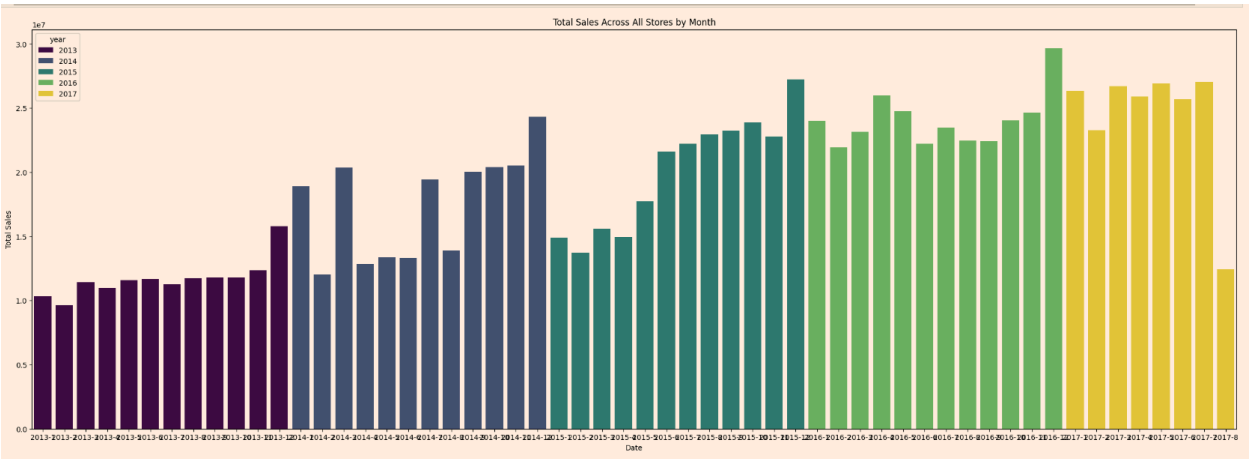
Additionally, there are open stores that never sell certain products. These products should not be removed but should be considered when modeling. It might make sense to always assume a 0 sales prediction for these products.



Seasonality:
The dataset was found to have strong day of week and monthly seasonality. The below image of the total sales per day of week shows that sales bottoms out in the middle of the week and is highest on the weekends.

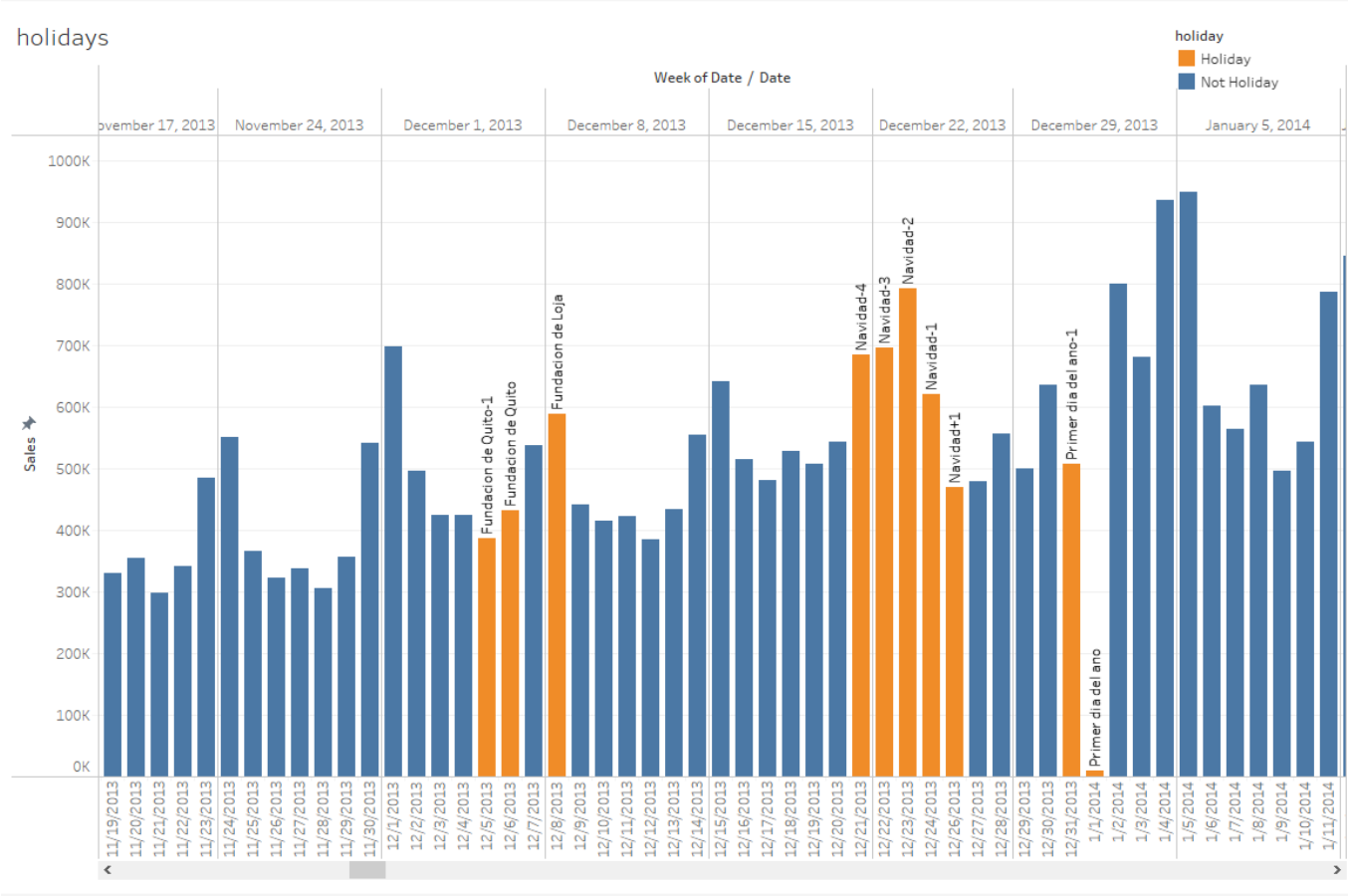


The image below shows the total sales per month. Each year is a different color. From this chart we can see that the sales generally get higher later in the year with a consistent maximum amount of sales in December.



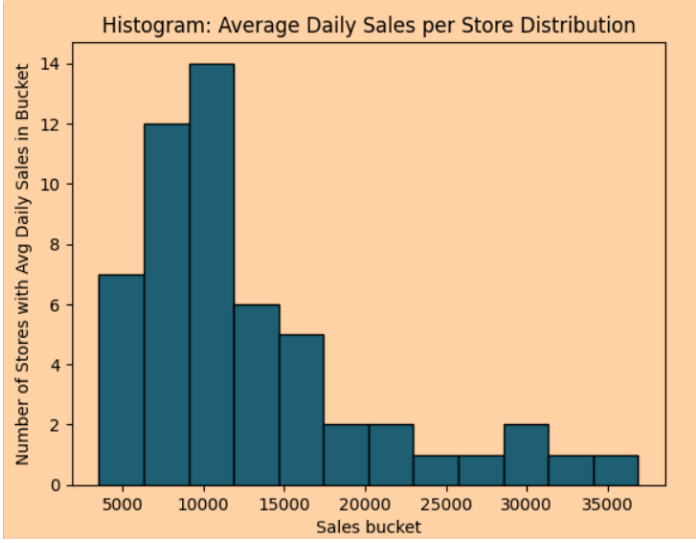
The seasonality found will be an important feature to feed into the models being used. SARIMAX and fbProphet have parameters for seasonality built in.

Holidays:
There are huge spikes and drops in daily sales. Some of these can be described by holidays. For example, Jan 1 of every year always has extremely low sales. On the other hand, certain holidays don't seem to impact the daily sales at all. Below is an example of a time window showing this. Holidays (orange bar) can both have a huge impact on sales or barely impact it at all. This led to the team using OLS (as described in feature extraction section below) to only keep statistically significant holidays. Full chart can be found here.



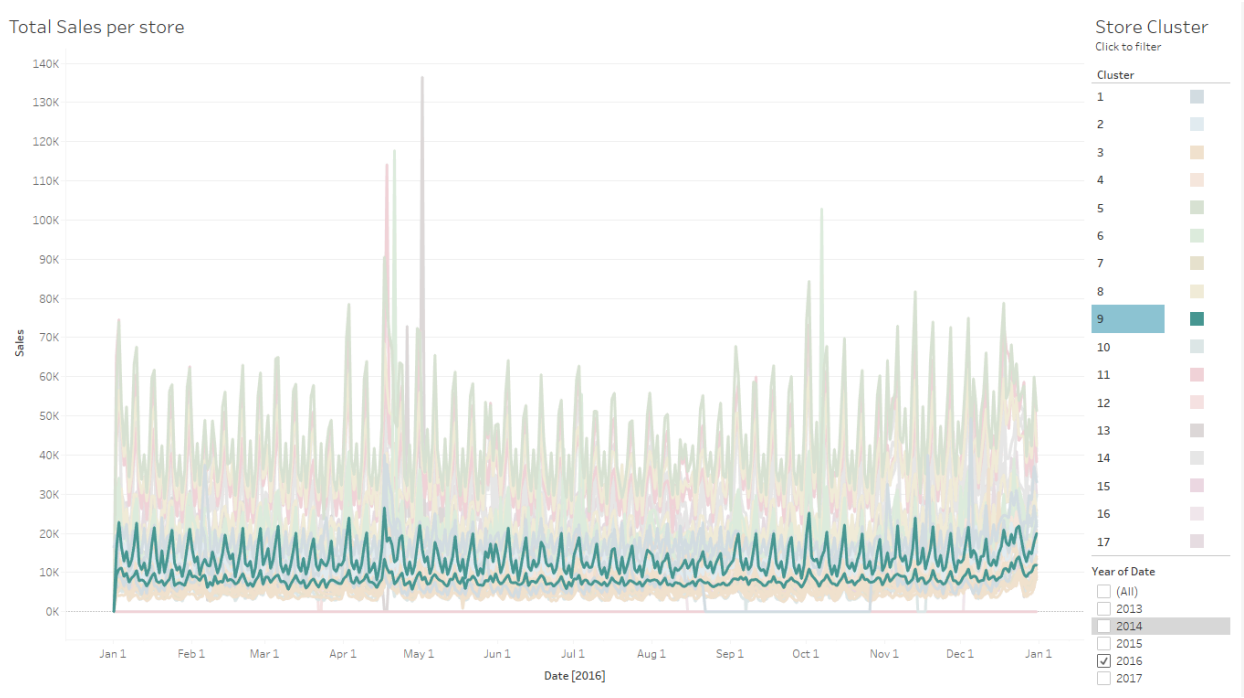
Store_nbr and Cluster:

The average daily sales per store is a right skewed distribution.

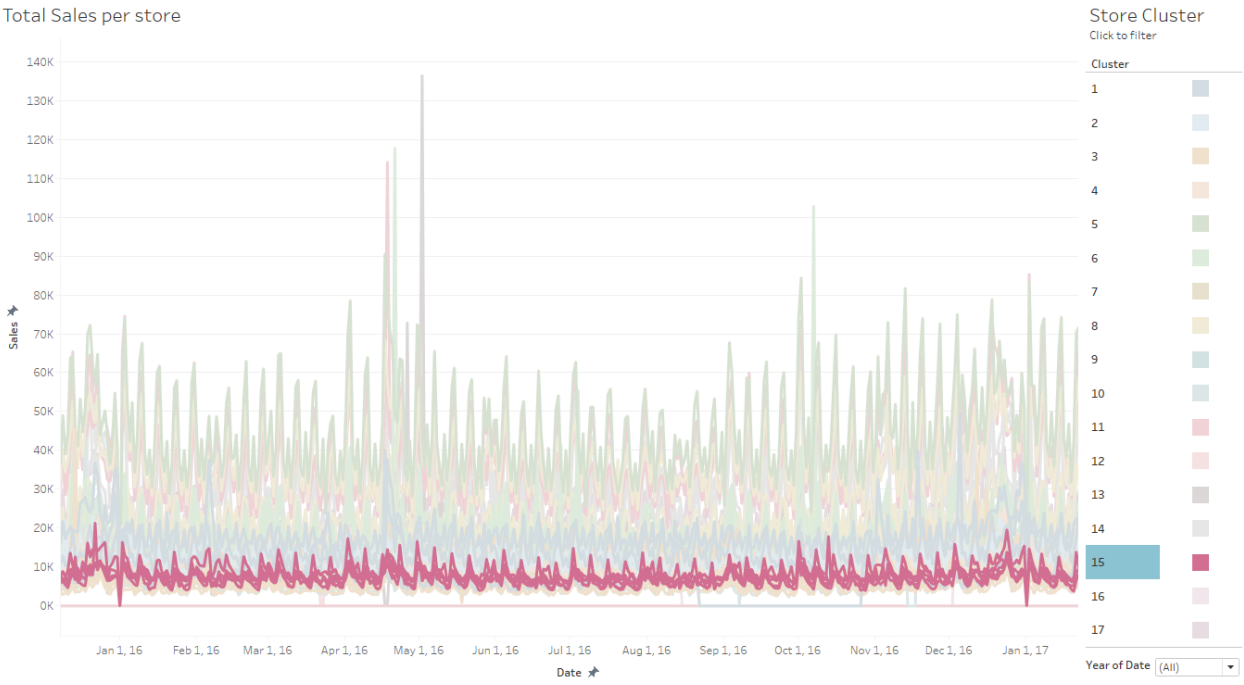


The cluster is a feature given by the store.csv. To find out what this meant, the team placed every store's daily sales together on a chart and colored the lines by cluster. Below shows how the stores in the same cluster tend to have similar sales amounts and patterns.

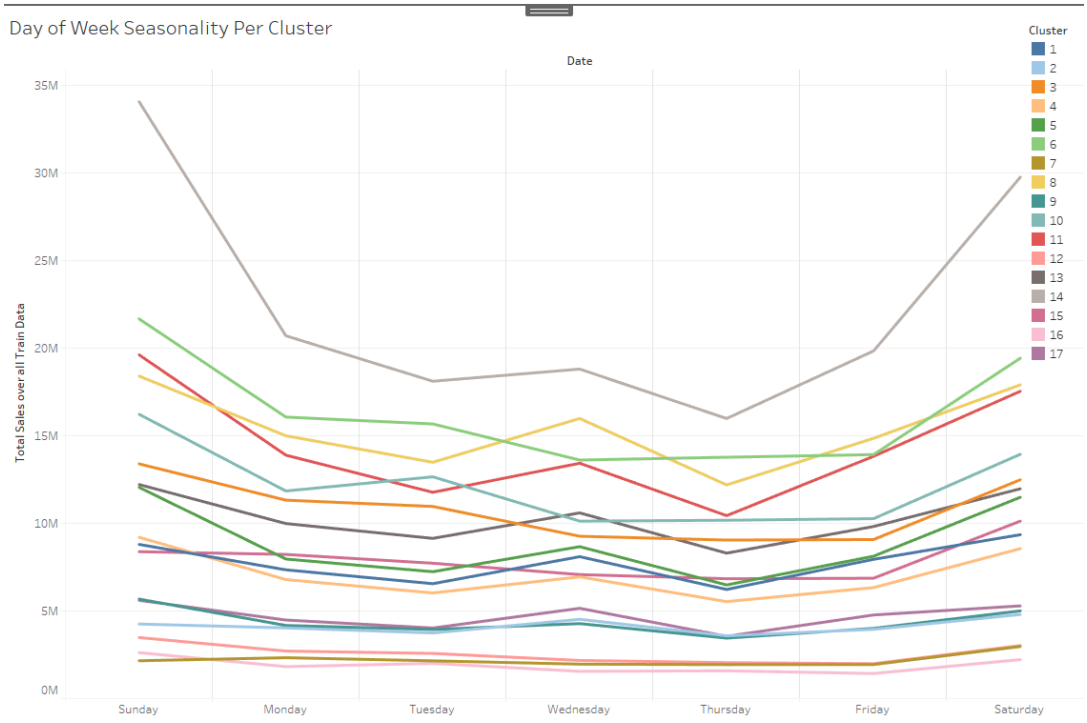
The chart below shows the stores in cluster 9 (stores 4 and 23) highlighted with the other stores' daily sales in the background. Chart can be found here.

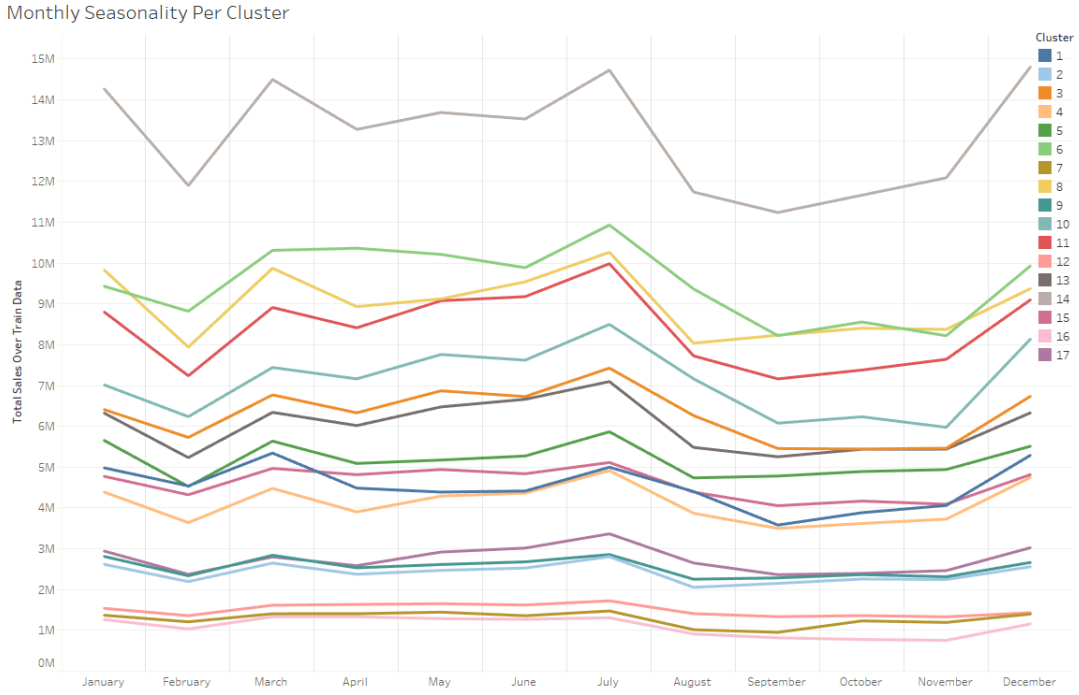


Below is another example of cluster 15 which contains stores 10, 12, 13, 15, and 19



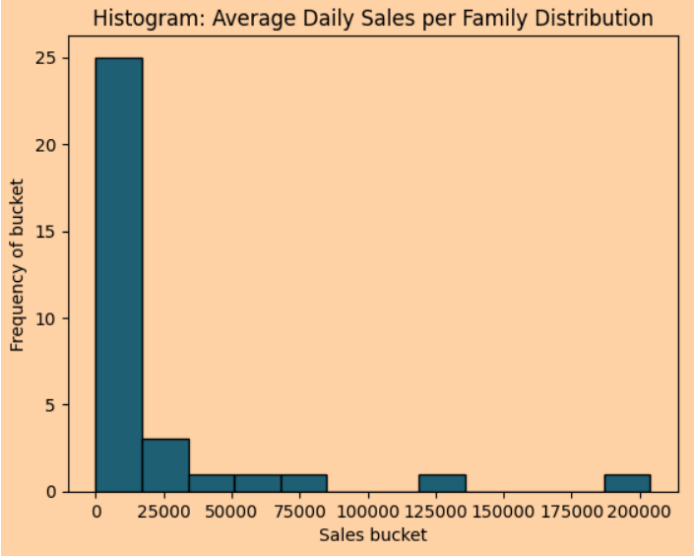
The team then looked to see if there was different seasonality per cluster in day of week and month. We can see that all clusters generally follow the same pattern so there does not look to be much to gain by looking at this level of granularity.



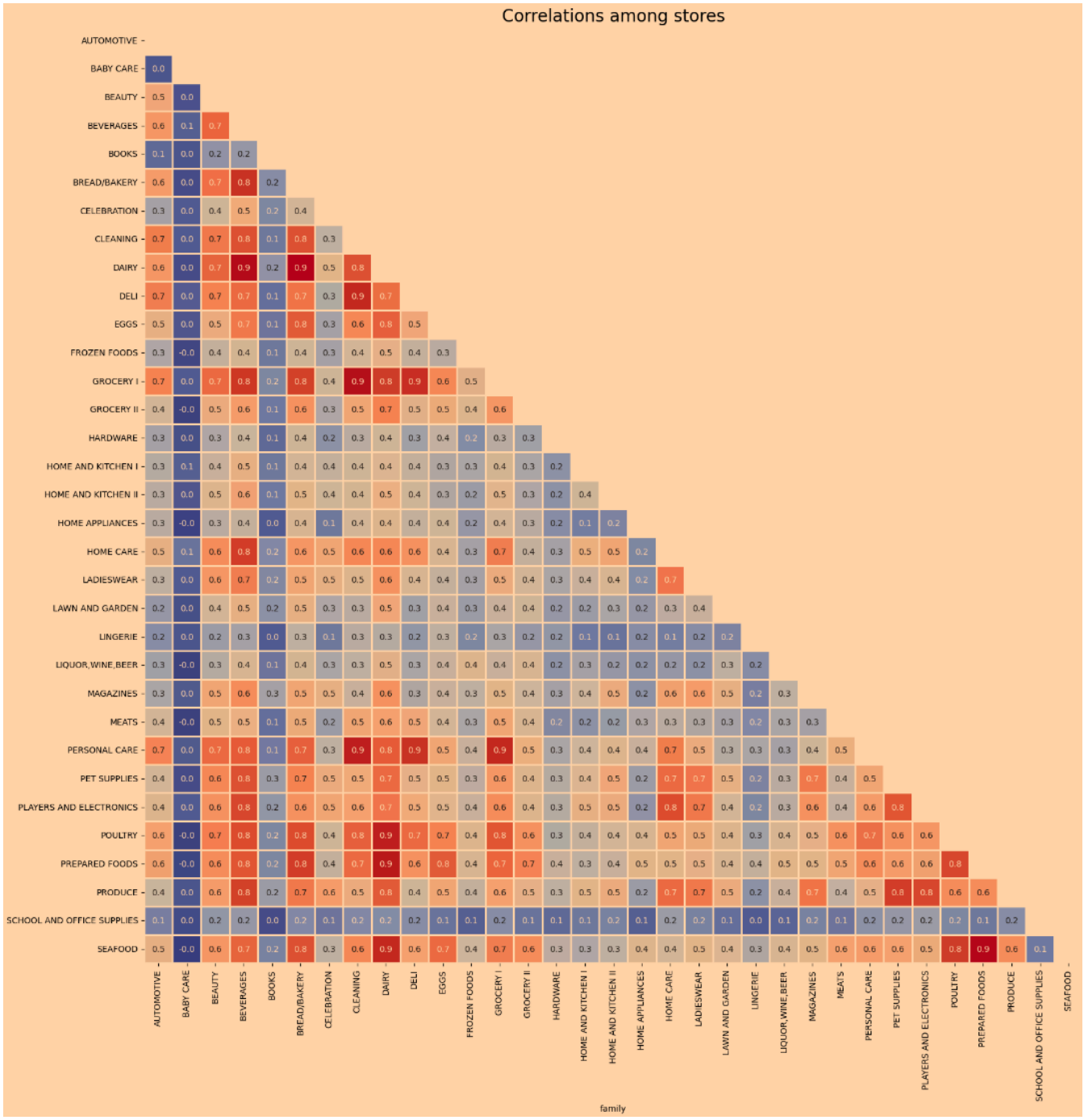


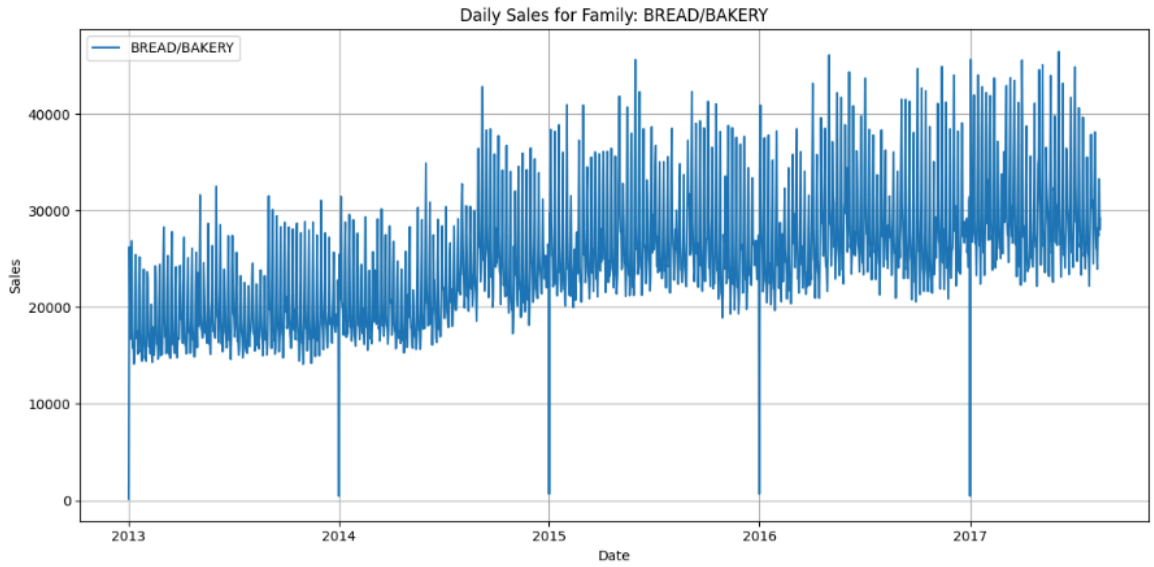
Family:

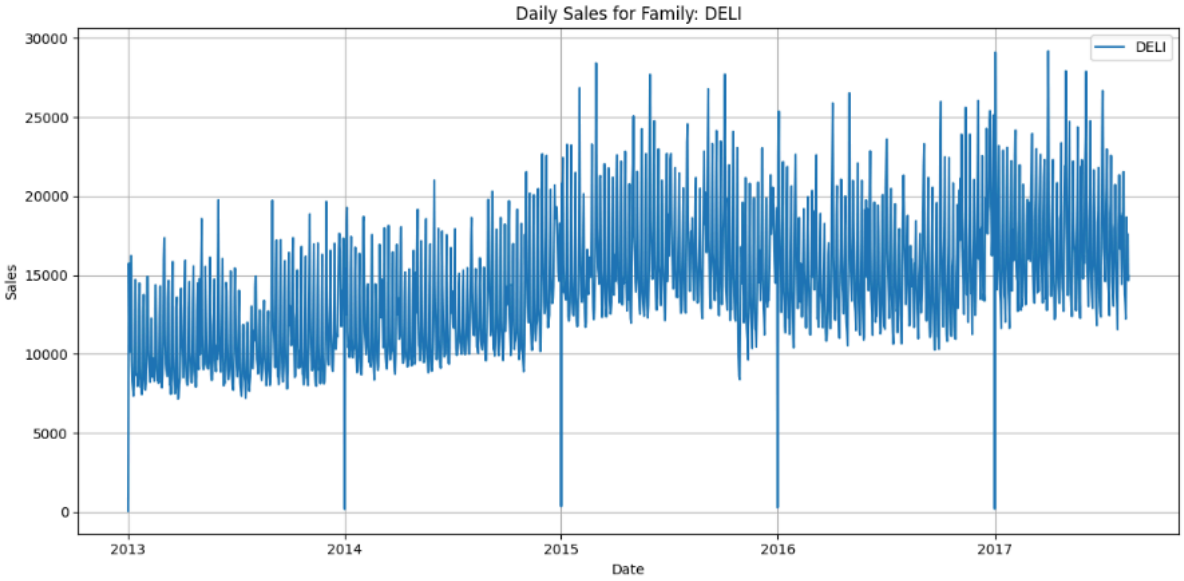
The sales distribution of families is a heavily right skewed distribution. Most product families do not sell very much per day. There are two outliers which are beverages and grocery I which are the two buckets on the far right of the chart below.



There are 33 distinct product families. This would be a lot to include in modeling as each would be its own feature when one hot encoded. The team made a correlation map[2] to see the relationships between the sales of different families. We can see that certain products like baby care have minimal correlation with any other family. Many other products are strongly correlated like dairy with seafood, prepared foods, poultry, grocery I, beverages, and bread/bakery. This tells us customers buy these products together. This is a good use case for PCA which the team used to combine families into a smaller group of clusters (described below in feature extraction)


Correlations among stores
AUTOMOTIVE
BABY CARE
BEAUTY
BEVERAGES
BOOKS
BREAD/BAKERY
CELEBRATION
CLEANING
DAIRY
DELI
EGGS
FROZEN FOODS
GROCERY I
GROCERY II
HARDWARE
HOME AND KITCHEN I
HOME AND KITCHEN II
HOME APPLIANCES
HOME CARE
LADIESWEAR
LAWN AND GARDEN
LINGERIE
LIQUOR,WINE,BEER
MAGAZINES
MEATS
PERSONAL CARE
PET SUPPLIES
PLAYERS AND ELECTRONICS
POULTRY
PREPARED FOODS
PRODUCE
SCHOOL AND OFFICE SUPPLIES
SEAFOOD
family





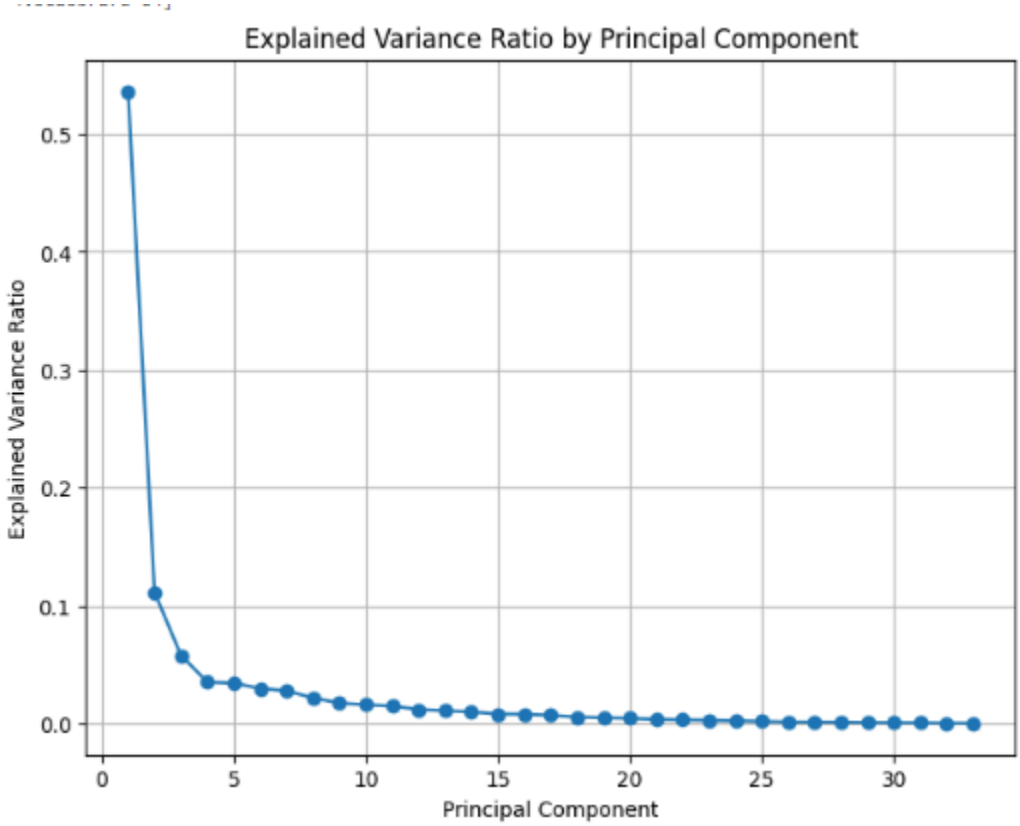
Observations in the heatmap above, pointed to similar trends within specific product families. Digging a bit deeper, we plotted the sales across different product families in separate line plots. Similarities, like the trends between bread/bakery and deli pointed to the possibility of clustering to create product categories that product families may roll up to.

Feature Extraction

Summary of the steps for each created feature:

1. isPromoted: True if item is promoted False if not
2. Day_of_week: numerical day of week taken from date
3. Month: numerical month taken from date
4. Year: numerical year taken from date

5. Holiday: 83 of the original 103 holidays were dropped. The effect of each holiday was measured by creating a dummy variable for holidays then running OLS on the holidays, onpromotion, dates, cities, and sales. Holiday's that had a p value below .05 were considered not statistically significant and were dropped. Holidays happen at different levels. A local holiday only impacts a city, regional holiday only impacts a state, and a national holiday is for all points in the dataset. Merges must be made to reflect this.
6. Rolling_sales_mean[2]: Rolling mean for the past 7 days, used to smooth trends, reduce noise, and help in identifying patterns.
7. Sales_lag: Lag feature for the sales column, 1 day lag. The sales variable is shifted one day backward in time to explore autocorrelation and to use as an additional input feature to our ML models.
8. Family_cluster: We reshaped the train table so that each row was a date, each column was a product family, and each cell represented the number of sales on a specific date for a specific product family. We determined that dimensionality reduction using PCA could be a viable option for representing the data, based on the similarities seen , and came up with a graph of explained variance (shown below) that easily lets us apply the elbow method to choose a good representation. Four principal components seemed to be a good representation, and shows that there's 4 product categories that the families roll up to, displaying similar trends in sales movement.



Feature Scaling: We will be scaling our features using MinMaxScaler to introduce normalization, improve gradient descent convergence, and create consistent model performance. The features we are scaling are “onpromotion”, ‘day_of_week’, ‘lag_1’, ‘rolling_mean’, ‘sales’, and all the family columns. By scaling these features to make them each between 0 and 1 we can optimize our performance later on and generalize any models we develop.

Tests

We have run RMSE and RMSLE on our ARIMA and SARIMA models so far. The SARIMA model has performed better with a RMSE of 0.0472 and a RMSLE of 0.0844, we will continue to pursue this model, OLS, and fbProphet moving forward.

Modeling:

SARIMAX[7], fbProphet[8], and Ordinary Least Squares (OLS)[9], and will be used to model and make predictions on sales for each ID in the test set, in that order of importance from highest to lowest. SARIMAX and fbProphet were designed specifically for time series data so they should have an advantage over OLS. OLS will be used as a baseline. All 3 models will be evaluated against each other using root mean squared error (RMSE) (equation 1) and root mean squared logarithmic error (RMSLE) (equation 2)

Equation 1: $$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

Equation 2: $$RMSLE = \sqrt{\frac{\sum_{i=1}^{n}(log(1+y_i) - log(1+\hat{y}_i))^2}{n}}$$

SARIMAX:

As we complete extensive exploratory data analysis, we see a few key trends and features in our dataset. In general, we have an upward trend in sales as time goes on. The general nature of a time series dataset lends itself well to the ARIMA model. ARIMA is made up of 3 parts [6].

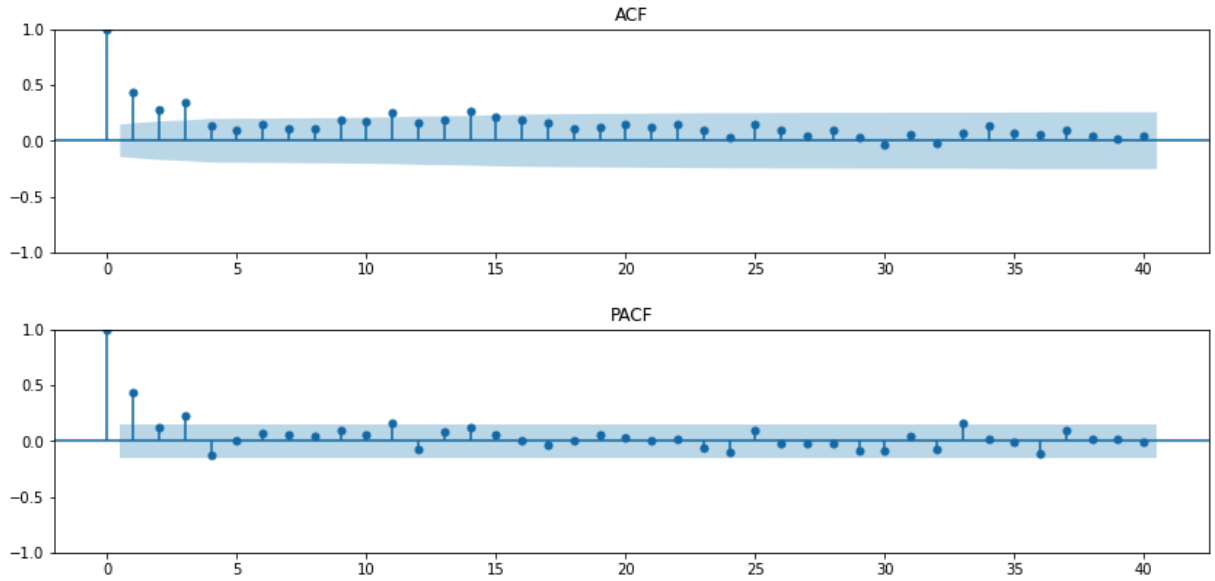
- AR - autoregressive: how the past values affect the current values
- MA - moving average: the relationship between current values and the previous values white noise
- I - differencing: The differencing step makes the data stationary by taking the difference of the target variable between time steps. Stationary data has mean, variance, and autocorrelation that are the same over time.

However, there is strong seasonality, which given the grocery context, makes sense and is something we can exploit to create an accurate model. Due to the seasonality, ARIMA will not suffice and we will utilize the SARIMAX model.

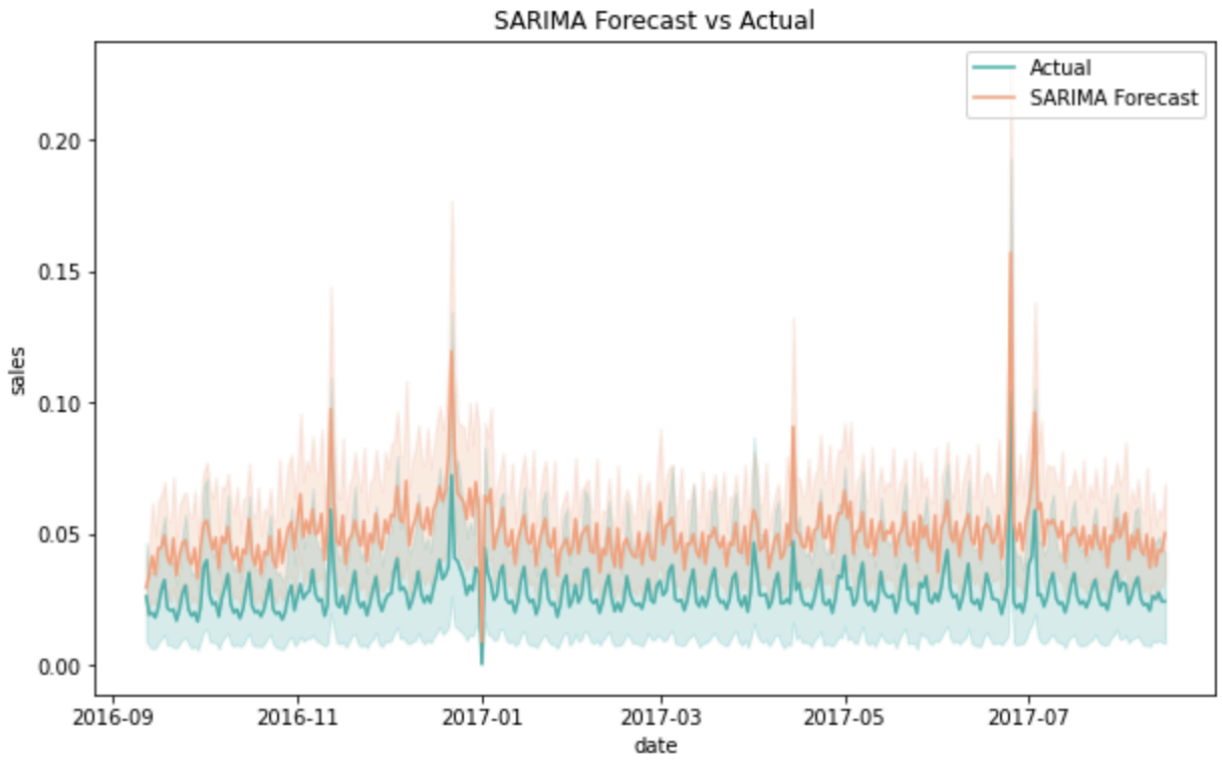
- S - Seasonality: parameter to specify seasonal cycles in the time series

- X - exogenous variables. This allows for other features that could impact the forecast. In our case that is features like holidays and family.

To quantify the autoregressive component, we computed the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) on our data, both shown below with 40 lags. The ACF and PACF in our case show the direct and indirect effect of past day's sales on the sales from our current day. Through this analysis we determined our data lends itself well to an autoregressive model. The charts below show that the previous day's sales largely impacts the current days. The blue shading represents the error associated with the ACF and PACF model respectively; anything outside of the blue shading can be considered a significant impact. There is a big drop after 1 day and then another big drop after lagging 3 days. This implies we should work with a lag_1 in our ARIMA model and add it as a feature moving forward. We will reevaluate in hyperparameter tuning whether using lag_2 or lag_3 will increase the accuracy of our model.



Additionally, we see strong signs of a moving average in our data, the ARMA family of models will be the family we work with moving forward. Due to the clear seasonality, SARIMA is our selection for our model. The SARIMA has yielded strong results so far, as shown in the graph, boasting a MSE of 0.0022 and a RMSLE of 0.0844. We will continue to pursue hyperparameter tuning to get the results as accurate as possible.



We also see the possibility of gradient boosting being effective here. The gradient boosting framework XGBoost, employed as an ensemble learning technique, will be another source of exploration moving forward. XGBoost's ability to handle missing data (some of which we have, particularly in the oil price column) and cross-validation ability could make it a strong model for this application.

Ordinary Least Squares (OLS): OLS[4] is linear regression to find a line that minimizes the squared difference of the errors between actual and predicted data. OLS will be used as a baseline model. It is a good baseline because it is interpretable and simple. OLS tells us how much each independent variable changes the dependent variable through the coefficients. It also tells us how significant each independent variable is through p value[3]. However, OLS is not as flexible. It does not handle seasonality well and does not understand large jumps in the dataset. The Ecuadorian sales data has both seasonality (through day of week and months) and large jumps (like after an earthquake in April 2016 which is modeled as a holiday).

To help evaluate the model, the R^2value will be used which is a value 0 through 1 which measures the amount of variance explained by the independent variables. 0 being none of the variance and 1 being all of the variance is explained by the model. Additionally, a F-statistic will also be used to measure the statistical significance of the model as a whole. Typically, a F-statistic <.05 indicates that the model as a whole is statistically significant. This can also be done with individual variables to help decide if those variables are impactful in the model. This is used in the feature engineering step to reduce the number of holidays the model will consider.

The model will be implemented using statsmodels.api library. Date time data will be modeled as integers. Holidays and product families will be modeled with dummy variables.

Facebook Prophet

To forecast future sales and analyze the impact of events Prophet will be used. Prophet was developed by Facebook to predict business times series sales data. This model will be a perfect fit here as it is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. This are the exact circumstances we have here and thus will be of great benefit to us. Prophet breaks the data down into three components[5]:

1. Overall trend of the times series data using piecewise linear regression.
2. Seasonality component using Fourier series
3. Noise which is random and cannot be explained by the previous two points

Prophet is advantageous because it has seasonality built into it and can understand jumps in sales due to events or holidays.

Team Member Contribution:

Alec - Data Exploration and Feature Engineering (On Promotion trends, oil and holiday trends, sales across product families)

Cory - Data exploration on seasonality, store groupings, and missing values. Deep dive into holidays data to extract relevant holidays. Build data pipeline to transform train and test data into model-ready dataframes

Will - Model exploration based on insights surfaced during data exploration. Implementing feature engineering and model evaluation.

Risks:

Have not taken a deep dive into fbProphet. May require reworking features in order to fit model needs. Apart from that we do not see any risks at this time.

References:

1. Alexis Cook, DanB, inversion, Ryan Holbrook. (2021). Store Sales - Time Series Forecasting. Kaggle. https://kaggle.com/competitions/store-sales-time-series-forecasting
2. Ekrem Bayar. (2022). Store Sales TS Forecasting - A Comprehensive Guide https://www.kaggle.com/code/ekrembayar/store-sales-ts-forecasting-a-comprehensive-guide
3. Kartushov Danil. (2022). Econometrics is all you need https://www.kaggle.com/code/kartushovdanil/econometrics-is-all-you-need

4. Jiahui Wang. (2020). How To Model Time Series Data With Linear Regression https://towardsdatascience.com/how-to-model-time-series-data-with-linear-regression-cd94d1d901c0
5. Khare, P. (2023) Understanding FB Prophet: A Time Series Forecasting Algorithm, medium.com. https://medium.com/illumination/understanding-fb-prophet-a-time-series-forecasting-algorithm-c998bc52ca10
6. Matt Sosna (2021) A Deep Dive on ARIMA Models https://towardsdatascience.com/a-deep-dive-on-arima-models-8900c199ccf
7. Bajaj, A. (2021, May 11). *ARIMA & SARIMA: Real-World Time Series Forecasting*. Neptune.ai. https://neptune.ai/blog/arima-sarima-real-world-time-series-forecasting-guide
8. Facebook. (2019). *Prophet*. Prophet. https://facebook.github.io/prophet/
9. XLSTAT. (2017). *Ordinary Least Squares regression (OLS)*. Xlstat, Your Data Analysis Solution. https://www.xlstat.com/en/solutions/features/ordinary-least-squares-regression-ols