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.
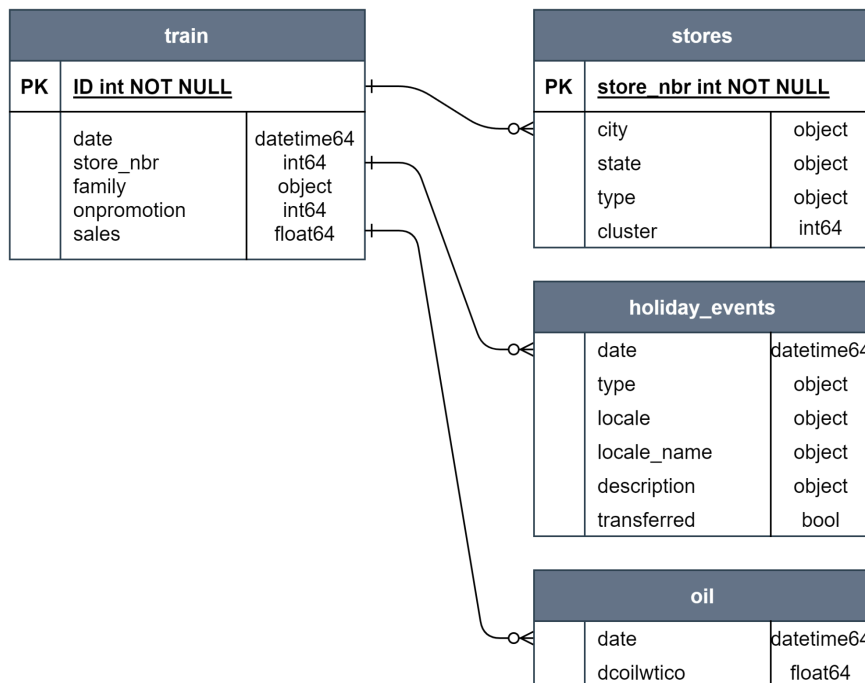
Below is the final table used. It includes features used from the original dataset tables and features created with feature engineering in green.

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
x32	family_dummies — boolean

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

We were able to leverage existing work 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. We added the holiday dummies which we shortened using OLS (description in progress report section) and family dummies. A description of the features is below:

- 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
- Dummy variables are set up for each family of items in family_dummies

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: $RMSE = \sqrt{\dfrac{\sum\limits_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$

Equation 2: $RMSLE = \sqrt{\dfrac{\sum\limits_{i=1}^{n}(log(1+y_i) - log(1+\hat{y}_i))^2}{n}}$
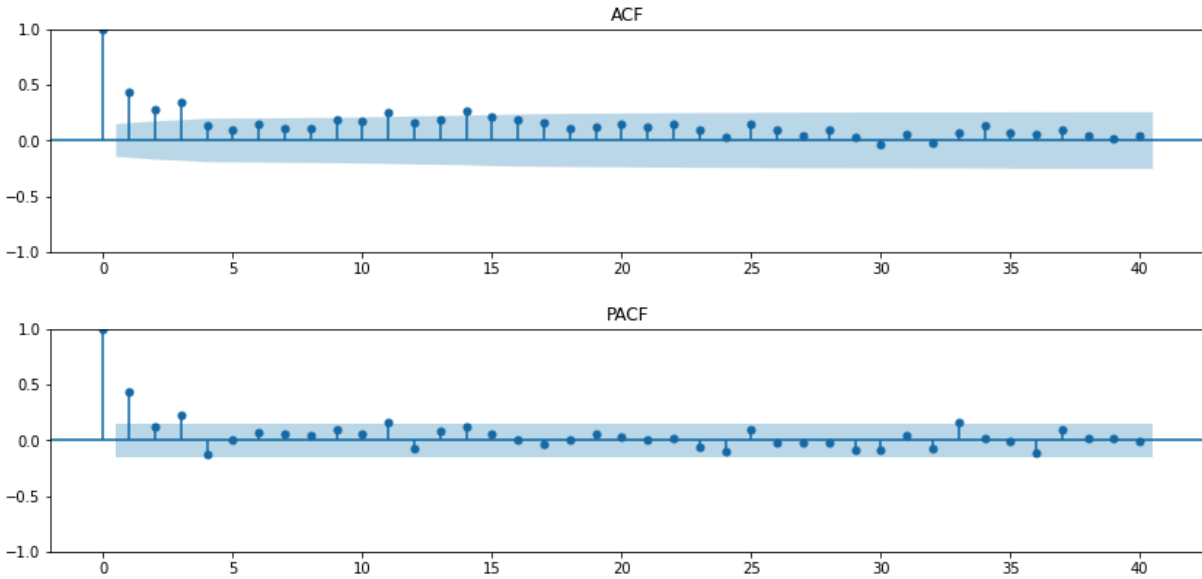
SARIMAX:

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.
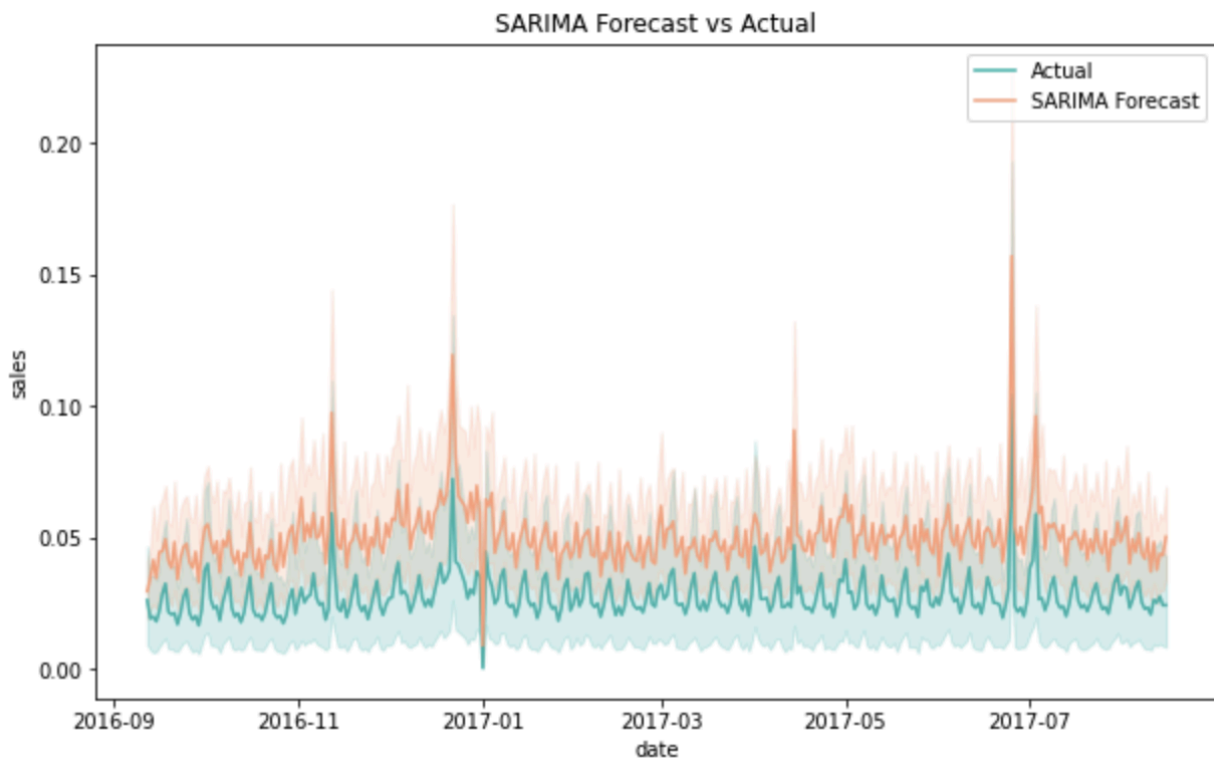
We also see the possibility of gradient boosting being effective here. The gradient boosting framework XGBoost, employed as an ensemble learning technique, will be another source of exploration moving forward. XGBoost's ability to handle missing data (some of which we have, particularly in the oil price 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).

To help evaluate the model, the R^2 value 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.

Progress Report Details:

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.
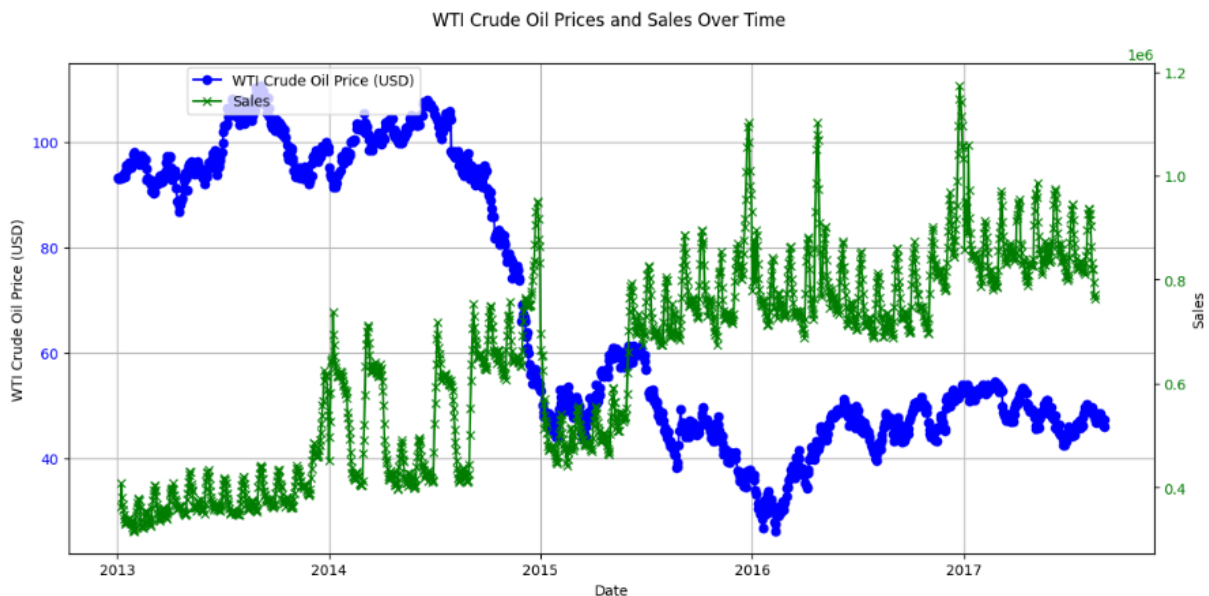
Date [2]:

The date range of the train data is from 2013-01-01 to 2017-08-15. Not all stores were open the entire date range but rows existed in the dataset. The values for sales in these rows would be 0 until a store would open. These rows were removed to prevent them from bringing down the sales predictions.

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. We are shifting the sales variable one day backward in time to explore autocorrelation and to use as an additional input feature to our ML models.
8. Family_dummies: family category is one hot encoded

Feature Scaling: We will be scaling our features using MinMaxScaler to introduce normalization, improve gradient descent convergence, and create consistent model performance.

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.

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