M5 Forecasting system Report

Prateek Bagora-40156671

Saman Soltani-40093581

Professor:

Tristan Glatard

Concordia University

Winter 2021

Table of Content:

Abstract:

Today, forecasting and predicting is one of the hottest trends in the industry. One of the industries that produce a huge amount of data every day is retailing. This data could be exploited to gain strategic insights for setting up appropriate inventory, achieving benchmark service levels, and so on. It is a challenge to handle and analyze this magnificent amount of data produced by the market giants. These companies invest a lot of their money and time to improve their forecasting solutions for achieving accurate predictions and estimating the levels of uncertainty in these predictions to avoid costly mistakes and maximize their revenue. So, we decided to work on this hot trend by choosing one of the biggest companies in the world, Walmart, where 95% of US shoppers spent their money.

Introduction:

• Context:

In retail stores like Walmart with a rapid transaction rate, maintaining a balance between customer demand and inventory is a crucial task. If we could achieve optimal forecasts of product sales for the stores spread across various regions, it would enable the decision-makers to make the best use of limited inventory space and maximize the profits. Following on these lines, we will analyze the unit sales of products across the Walmart stores in the United States and try to make optimal predictions that could aid this decision making.

• Objective and Problem to Solve:

In this project, we will analyze the unit sales of products across the Walmart stores in the United States to first explore the factors that have a significant effect on the sales. Exploiting these factors, along with the corpus of historical sales data, we will implement a prediction system to estimate the unit sales of products at various stores for the next 28 days. In the process, we will analyze and exploit the explanatory variables like weekly prices, promotions, day of the week, special events (such as Super Bowl, Valentine's Day, and Orthodox Easter), and so on.

Predicting unit sales can be as challenging as forecasting weather. For instance, a wrong prediction for the weather may result in us wearing a warm jacket or carrying around an umbrella on a sunny day. Similarly, an inaccurate prediction in a business scenario may result in financial losses or loss of opportunities. So, we will do an exploratory data analysis of the dataset incorporating various perspectives. We will further implement various machine learning algorithms and compare their results to determine the method that provides the most accurate predictions, also stating the uncertainties associated with it.

• Related Work:

1. The M5 Accuracy competition: Results, findings and conclusions: https://www.researchgate.net/publication/344487258_The_M5_Accuracy_competition_Results_findings_and_conclusions
2. Time Series Forecasting-EDA, FE & Modelling: https://www.kaggle.com/anshuls235/time-series-forecasting-eda-fe-modelling/

Contrary to most of the previous work, we will attempt to produce a more scalable solution to this problem with the application of Dask and PySpark.

Material and Methods:

Technologies:

In this project, we have primarily used Dask and PySpark, apart from the common python libraries. In the first phase of the project, that is, the exploratory data analysis, we have used Dask for data processing and Plotly for visualization. While in the second phase, that is, feature engineering and forecasting with machine learning, we have used PySpark. Although Dask sufficed for the exploratory data analysis, which mostly involved summarizing the data to generate a small-sized output, we saw an explosion of data size in the feature engineering phase. When we melted the time series data to obtain a long format, merged it with other details, and tried to add features, the data size exploded. While Dask was unable to handle it, PySpark performed satisfactorily. At a later stage, when we tried to further improve the performance of the predictions, we sought refuge in LightGBM.

Dataset:

The M5 dataset provides the unit sales of various products sold by Walmart stores in the United States over 1941 days, organized in the form of grouped time series. This dataset consists of 3049 products, classified into 3 categories (hobbies, foods, and households) and 7 product departments. These products are sold across 10 stores in 3 states (California, Texas, and Wisconsin). Besides the historical time series data, this dataset also includes further information like weekly price changes, SNAP days, and special events (such as, Super Bowl, Valentine's Day, Orthodox Easter, and so on), which will also allow us to analyze how these factors affect sales. The results obtained from the analysis of this dataset will be scalable as the same methods could be applied to gain insights for other products and stores.

The three files that we used from the dataset are:

1. calendar.csv - Contains information about the dates on which the products are sold. [1]
2. sales_train_evaluation.csv - Contains the historical daily unit sales data per product and store (d_1 - d_1941). [1]
3. sell_prices.csv - Contains information about the price of the products sold per store and date. The price is provided per week (average across seven days). If not available, this means that the product was not sold during the examined week. Note

that although prices are constant on a weekly basis, they may change through time (both training and test set). [1]

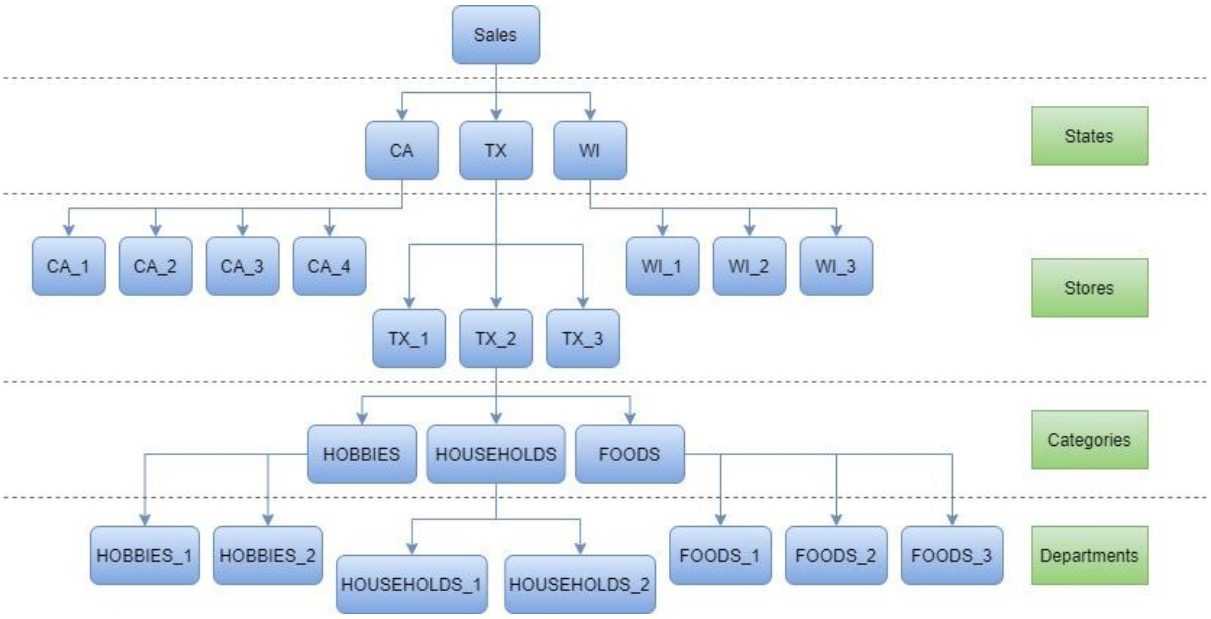

Figure 1. Structure of the sales data

Exploratory Data Analysis:

It is vital to visualize and understand the data to develop intuition about the significance of various features in the dataset. The dataset that we used had great potential for exploration, and we found a lot of interesting patterns which equipped us with the knowledge required for feature selection and engineering. We have described some visuals below:

- Sales by stores:

Figure 2 summarizes the sales by stores from 2012 to 2016. It can be observed that the sales drop to almost 0 on a particular day each year, that is, Christmas. Also, we get an intuition that the sales go up at the beginning of each year and come down towards the end of the year.

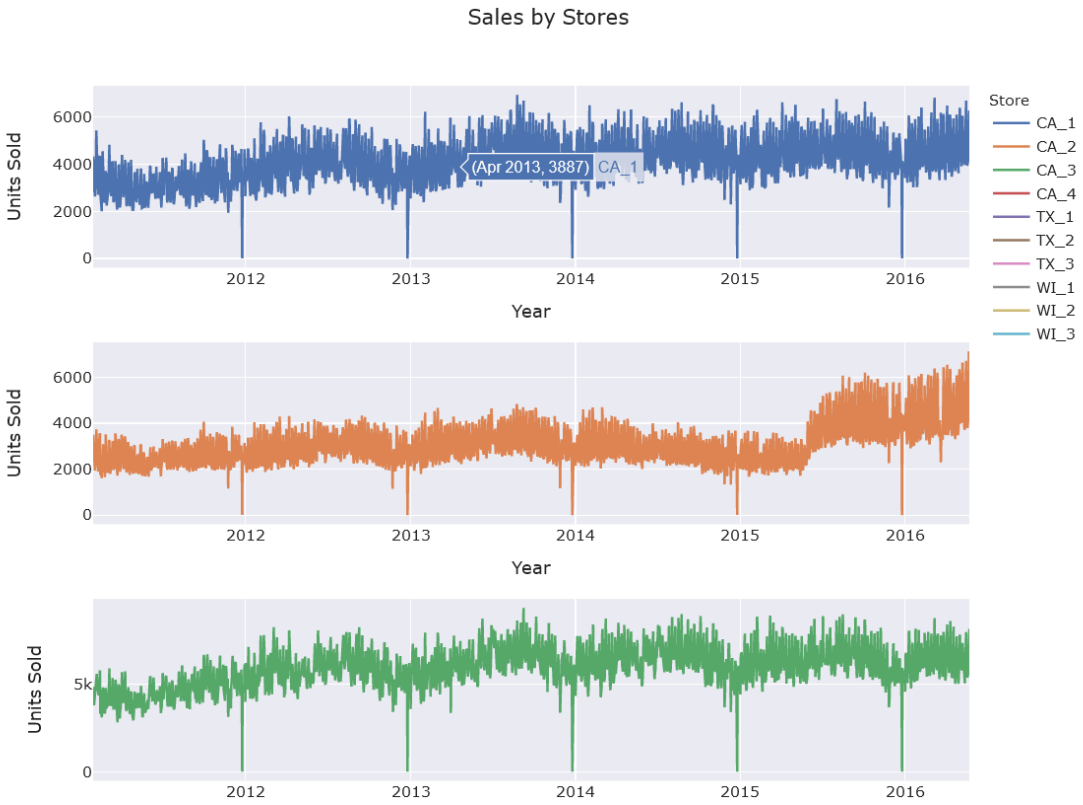

Figure 2. Sales by the stores: CA_1, CA_2, and CA_3

- Mean sales by stores over rolling windows:

Figure 3 shows mean sales by stores from 2012 to 2016 over rolling windows of sizes 7 and 30 days. Again, we get an intuition that the sales go up at the beginning of each year and come down towards the end of the year.

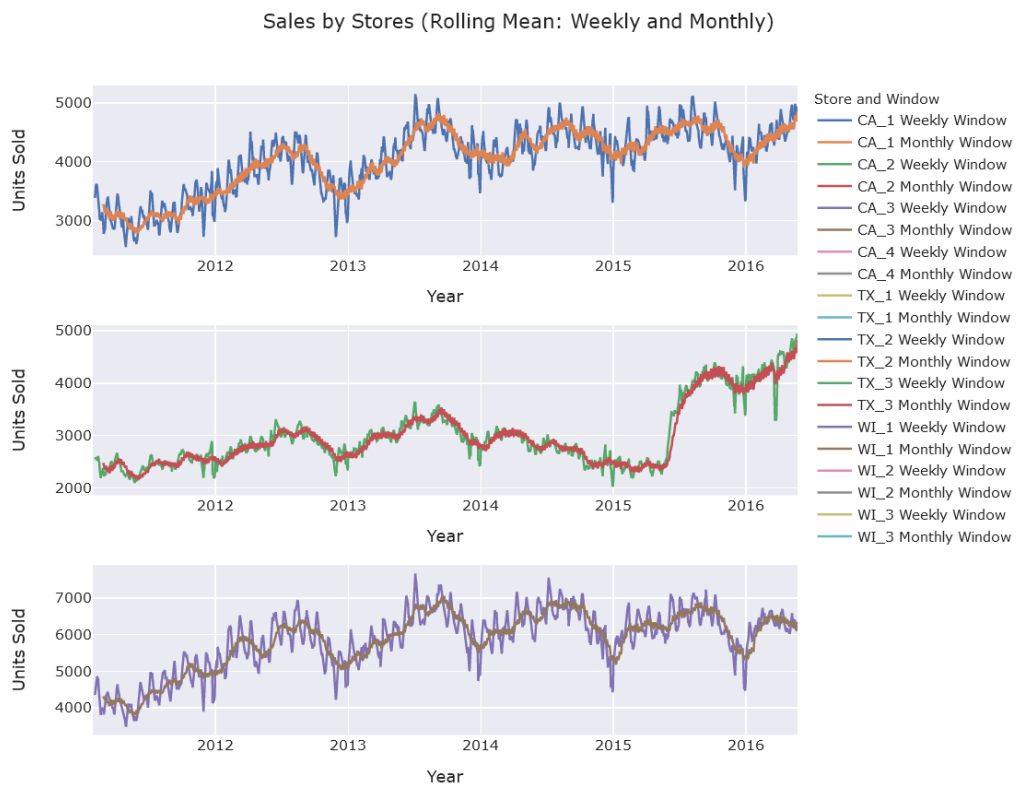

Figure 3. Rolling window mean of units sold over windows of sizes 7 and 30 days by each store

- Distribution of units sold:

Figure 4 shows the distribution of units sold by stores in each state. We can observe that CA_3, TX_2, and WI_2 are the top stores both in terms of the most units sold and the distribution of the units sold, in California, Texas, and Wisconsin respectively.

Figure 4. Distribution of units sold for each store

- Units sold by category over months:

Figure 5 shows an analysis of total units sold of each category in various states over months. We get an intuition that the sales of FOODS in CA and TX peak around the third quarter of the year.

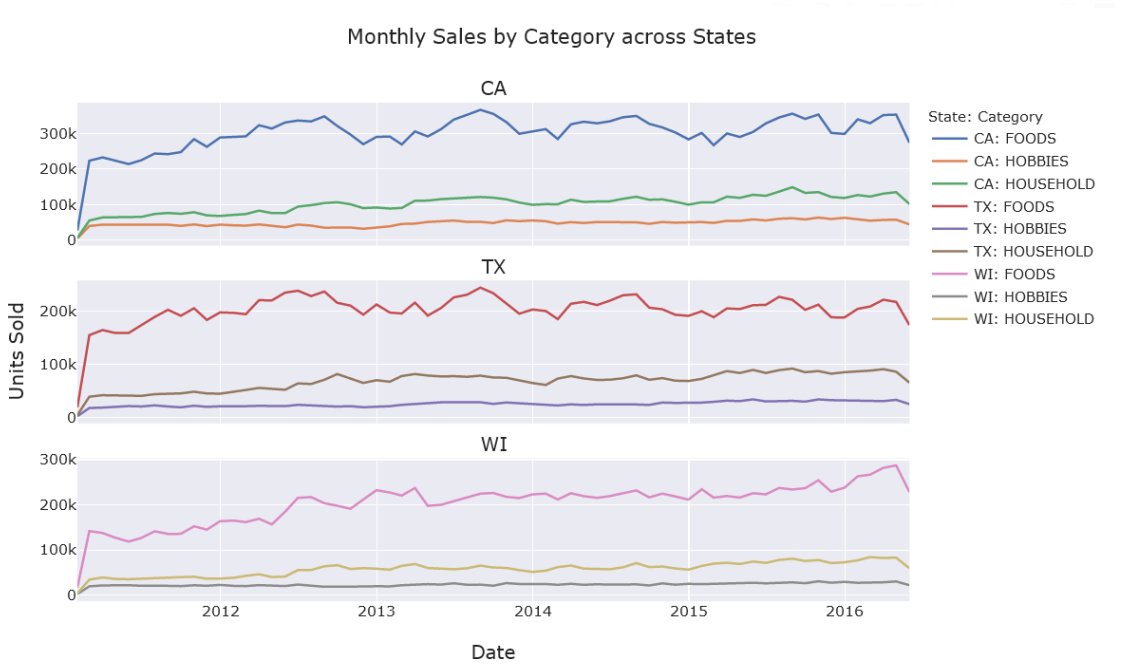

Figure 5. Total units sold of each category in a state over months

- Units sold by category and week day:

Figure 6 shows an analysis of total units sold each week of the day for various categories. We can observe that more items have been sold on the weekends.

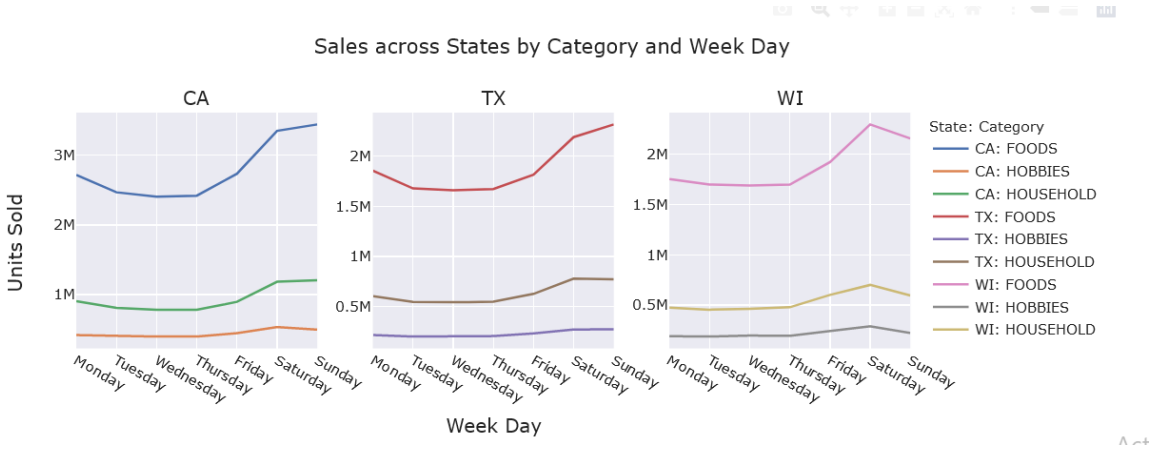

Figure 6. Units sold each day of the week for categories and states

- Mean prices of items by category:

Figure 7 shows the mean prices of items by categories over the weeks. We get an intuition that the mean prices of items in all three categories appear to be more or less uniform from 2014 to 2016. Also, the mean prices of items in the HOBBIES category appear to have increased, and the mean prices of items in the HOUSEHOLD category appear to have decreased over the period from 2011 to 2014. However, a deep analysis would be required before making any assumptions.

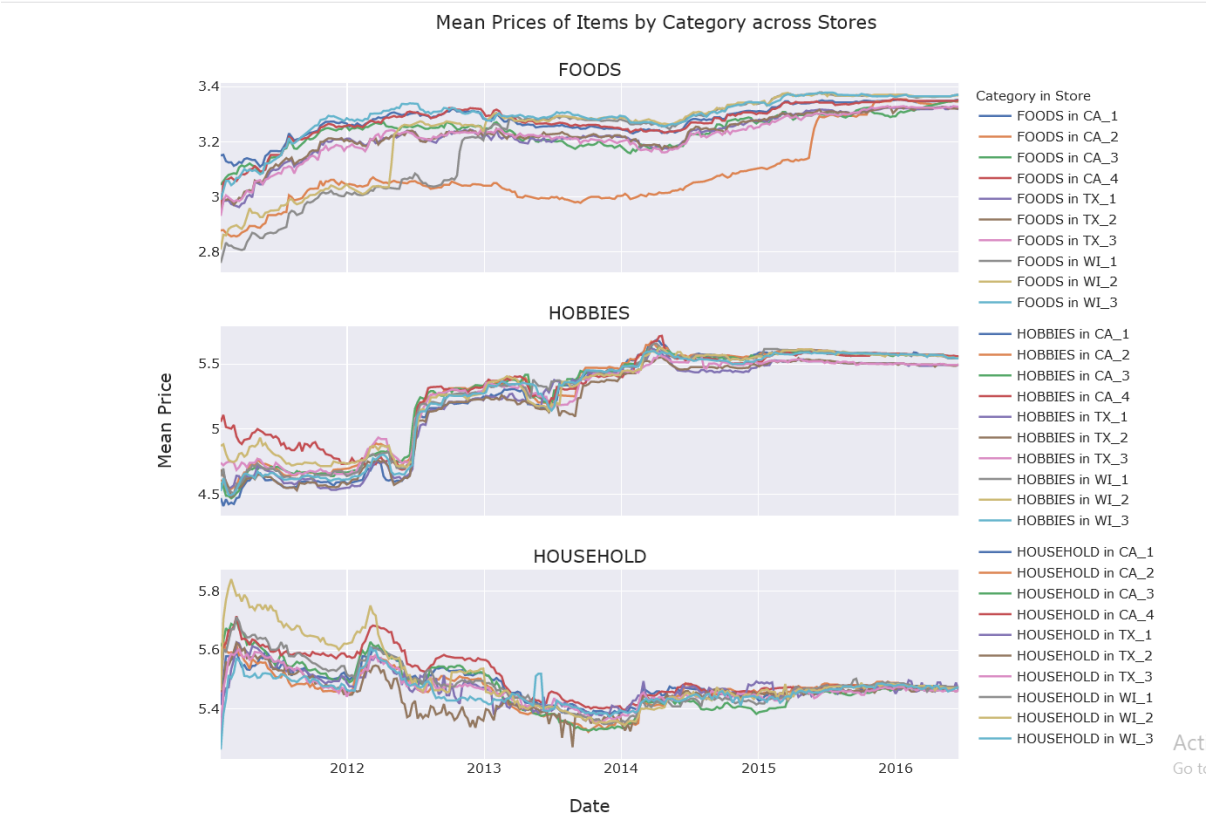

Figure 7. Mean prices of items of a category in different stores over the weeks

- Total sales on various events through the years:

Figure 8 shows the total units sold on each event day over the years from 2011 to 2015. Among the events, we observe the maximum sale of items in FOODS on Labor Day, followed by Super Bowl day. Also, we observe no sales on Christmas assuming that most of the stores are closed.

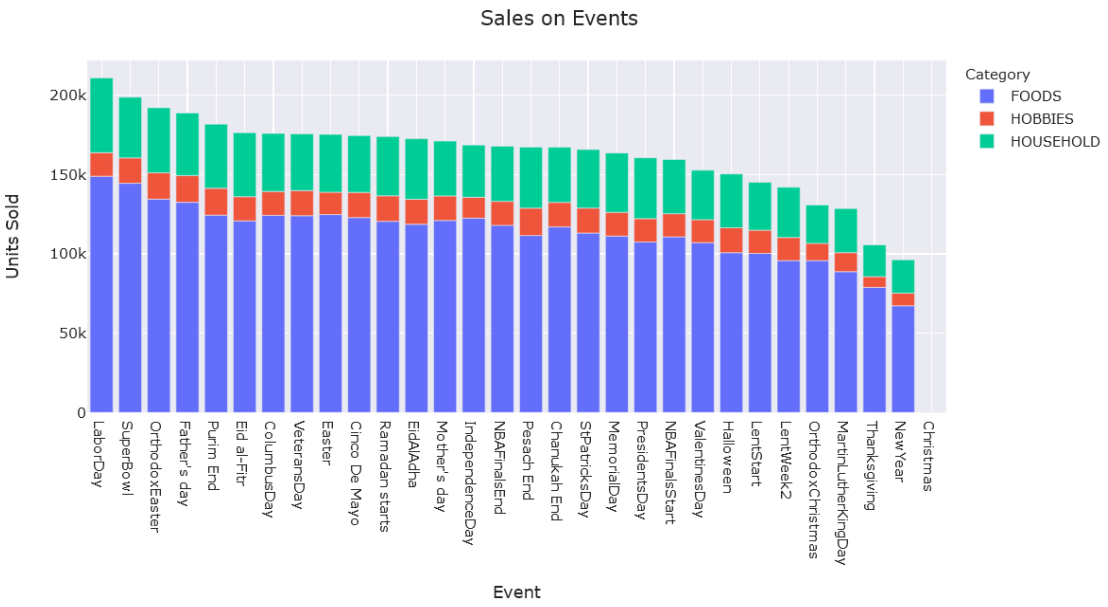

Figure 8. Total units sold of each category on event days

We did other analyses too that can be found in the “M5 Exploratory Data Analysis” notebook.

Approach:

Our approach comprises of three main phases as follow:

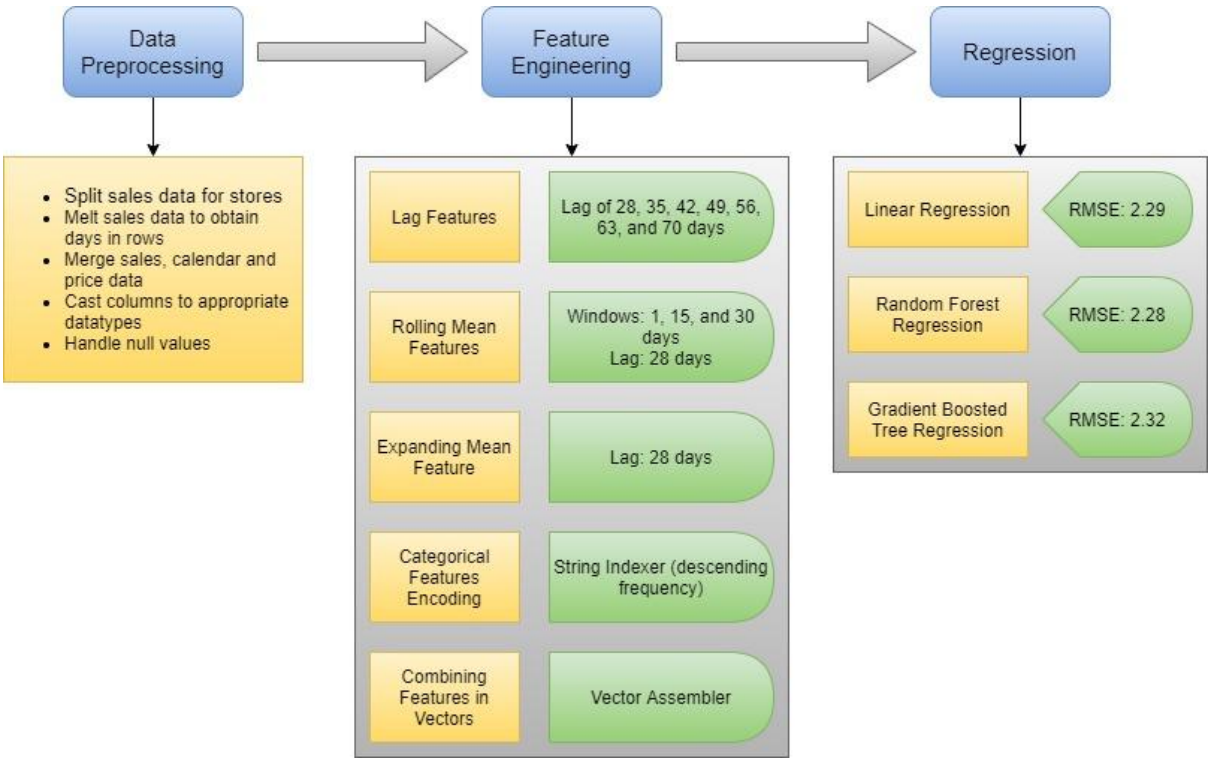

Figure 9. Workflow

Pre-processing phase:

This phase is comprised of the following follows:

1. Splitting and Melting: We split the sales data by stores, apply melt to convert the data from wide format to long format, and create separate files for each store.
2. Merging: We merge the data from all three files so that we have everything in one place.
3. Downcasting: We downcast the column types to the smallest possible data type that can store all the values of the column to reduce the amount of storage. With this conversion, we saved around 80% of the memory space.

Feature engineering Phase:

We need to reframe the time series dataset as a supervised machine learning dataset to apply regression algorithms. For that, we need to define input features that will have a strong relationship with our output variable. Therefore, this is one of the most crucial steps in the process. We introduced the following types of features to our dataset:

1. Lag features:

   Lag features are the classical time series features. We have introduced lags of 28, 35, 42, 49, 56, 63, and 70 days. Since we aim at making forecasts for the next 28 days, we need to have a minimum lag of 28 days, to ensure that we will not have null values in the features of the final 28 days we make the forecast for.

2. Rolling mean features:

   We have introduced rolling mean features using window sizes of 7, 15, and 30 days, lagged by 28 days.

3. Expanding mean features:

   We have an expanding mean feature with a lag of 28 days.

4. Categorical features:

   We have encoded the categorical features using PySpark's StringIndexer, encoding the labels by the descending order of their frequency in the dataset.

5. Mean features:

   We have used item mean, department mean, category mean, and store mean.

We also need to transform these features into vectors so that we can pass them to PySpark's ML library. We used PySpark's VectorAssembler to achieve that.

Regression Phase:

We applied three Machine Learning algorithms from the PySpark's ML library, which are, Linear Regression, Random Forest Regression, and Gradient Boosted Tree Regression. For evaluating the results, we used RMSE and NRMSE (normalized with standard deviation) metrics. We obtained the best results from Random Forest Regression. So, we tried to further optimize this model by performing hyperparameter tuning. However, it was inefficient as there is no time series cross-validation functionality in PySpark. Also, PySpark would crash if we tried to perform an exhaustive hyperparameter search.

Approach Improvement:

We realized that PySpark's ML library is not very efficient for the task at hand, and our results could further be improved. So, we decide to add a few tweaks to our approach, which are:

- Implementing a recursive forecasting model, which would allow us to add lag features with lags lesser than 28 days.
- The recursive model allowed us to add more lag features with a lesser amount of lag. We introduced:
    - A classical lag feature of 7 days.
    - Rolling mean features with window sizes 7 and 365 days, lagged by 7 days
- Introducing rolling mean features for price in addition to units sold.
- Applying LGBM Regressor in the vanilla python environment.

Since we are using lags of 7 days, we need to follow a recursive approach to determine forecasts for 28 days. We can calculate the forecasts for the first 7 days at once since we have all lag features populated. Now, for the next 7 days, we need to recalculate the features using the forecasts of the first 7 days before we can make a forecast for them, and so on.

We did the feature engineering in PySpark and took our final dataset to the vanilla python environment. This allowed us to exploit the scikit-learn library's TimeSeriesSplit and RandomizedSearchCV for hyperparameter tuning, and the LightGBM framework for machine learning. With this approach, we were able to obtain a better result.

Results:

1. Linear Regression model (PySpark MLLib):
    - 1.1. Hyperparameters:
        - 1.1.1. max_Iter = 15
        - 1.1.2. reg_Param = 0.3
    - 1.2. Result:
        - 1.2.1. RMSE : 2.29
        - 1.2.2. NRMSE : 0.637

2. Random Forest Regression (PySpark MLLib):
    2.1. Hyperparameters:
        2.1.1. maxDepth = 10
        2.1.2. numTrees = 15
        2.1.3. subsampling Rate = 1
    2.2. Result:
        2.2.1. RMSE: 2.28
        2.2.2. NRMSE: 0.635

3. Gradient Boosted Tree (PySpark MLLib):
    3.1. Hyperparameters:
        3.1.1. maxDepth = 10
    3.2. Result:
        3.2.1. RMSE: 2.32
        3.2.2. NRMSE: 0.646

4. Light GBM (LightGBM Framework):
    4.1. Hyperparameters:
        4.1.1. objective = 'tweedie',
        4.1.2. tweedie_variance_power = 1.3,
        4.1.3. n_estimators = 1000,
        4.1.4. num_leaves = 100,
        4.1.5. max_depth = 30,
        4.1.6. learning_rate = 0.03,
        4.1.7. feature_fraction = 0.7,
        4.1.8. bagging_fraction = 0.7,
    4.2. Result:
        4.2.1. RMSE: 2.18
        4.2.2. NRMSE: 0.6

In conclusion, LightGBM gave the best performance.

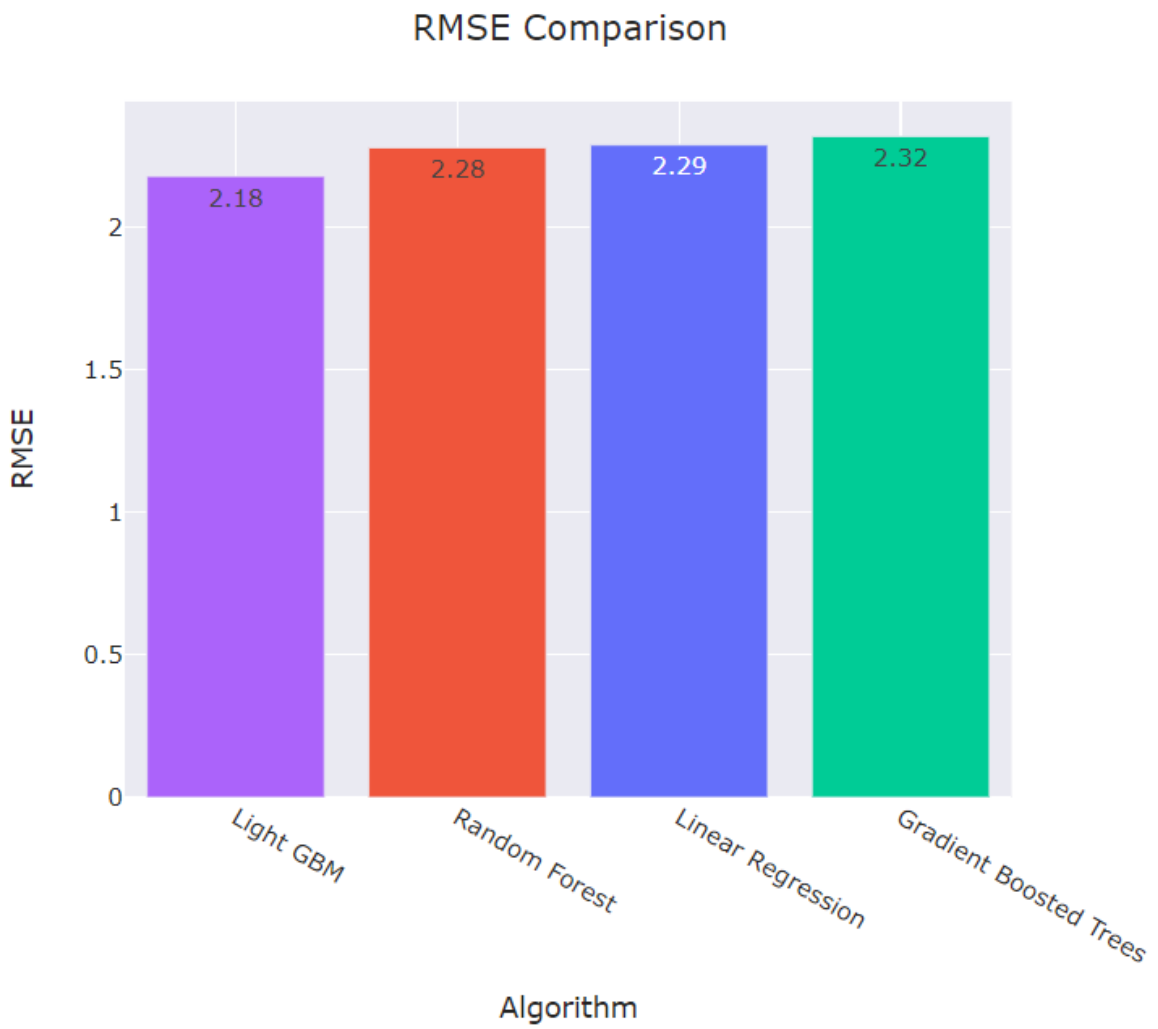

Figure 10. Comparison of results obtained from various algorithms

Conclusion and Future Work:

We were able to develop a prediction system to generate unit sales predictions for the next 28 days with satisfactory performance. In the process, we experimented with various features and machine learning algorithms in an attempt to improve the performance of the model. As we developed the project on personal computers, we faced several challenges with lack of memory being the most significant one. Dask did exceptionally well in summarizing the data for exploratory data analysis, however, it was not suited for data processing on our infrastructure. At the same time, while PySpark did exceptionally well for data processing and feature engineering, its machine learning library was very restrictive, especially for time series analysis. Also, an exhaustive hyperparameter search with PySpark on our infrastructure was close to impossible. As we moved from PySpark to scikit-learn and LightGBM, our lives were instantly made easier with their machine learning capabilities. In the future, we can further improve the performance of our model by experimenting with feature engineering in PySpark and machine learning algorithms in scikit-learn.

References:

[1]. https://mofc.unic.ac.cy/m5-competition/
[2]. https://www.researchgate.net/publication/344487258_The_M5_Accuracy_competition_Results_findings_and_conclusions
[3]. https://www.kaggle.com/anshuls235/time-series-forecasting-eda-fe-modelling/