# Machine Learning Engineer Nanodegree

## Capstone Project

AbdulRahman KASSAR
January 25th, 2022

## I. Definition

### Project Overview

Accurate sales forecasting is critical for retail companies to produce the required quantity at the right time. But even if avoiding waste and shortage is one of their main concerns, retailers still have a lot of room for improvement. At least, that's what people working at Walmart think, as they launched an open data science challenge in March 2020— the M5 competition — to see how they could enhance their forecasting models.[1]

The Makridakis Open Forecasting Center (MOFC) at the University of Nicosia conducts cutting-edge forecasting research and provides business forecast training. The MOFC is well known for its Makridakis Competitions and In this problem, the fifth iteration(M5 forecasting) the hierarchical sales data from Walmart, the world's largest company by revenue is given, to forecast daily sales for the next 28 days. This has led to the M5 being a unique competition that closely parallels the difficulties and challenges associated with industrial applications of forecasting.[2]

## Problem Statement

The given task is to predict sales for all products (about 3000) in each of the ten stores, on the days right after the available dataset, which is from 2011 to 2016. It means that about 30,000 forecasts have to be made for each day in the prediction horizon, which is can be easily done using aws sagemaker given the large size of the dataset generated after preprocessing.[1]

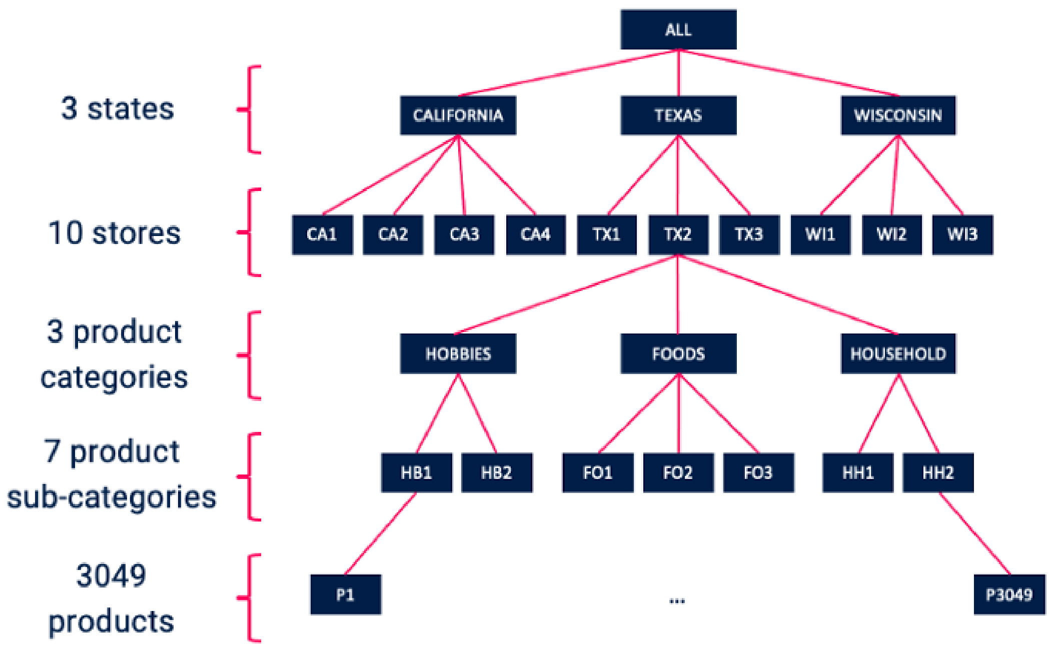


*Figure 1: Hierarchical Data (from areifact.com) [1]*

## Metrics

The performance metric is chosen for this problem which is Root Mean Squared Error (RMSE), also the well-known Mean Absolute Scaled Error (MASE) were used. Both of them are part of the sci-learn library.

# II. Analysis

## Data Exploration

The dataset provided is divided into three main files: sales, calendar and prices – all in csv format. After playing with sales data, we can see it contains the demand for each item for the given time period. This sales data also include state, store, category identifiers for each of these items. In total there are 3049 items, 3 main categories across 10 stores scattered in three states: California, Texas, and Wisconsin.

Before we can complete our data exploration, we have to reduce memory usage by downcasting each of the files given where we assigned appropriate data type for each column.

After downcasting these files we merged them to create one large dataset for our analysis and then for training our model.

Before we can merge calendar and prices data with sales data, we have to format sales data so we can merge it with calendar and prices. To do this, we have to convert its format from wide to narrow by melting it – making each item duplicated for each day.

Before we can merge calendar to sales data, we have to encode categorical columns to make ready before making large data manipulation like merging it with sales data.

After having calendar and sales data ready we merged them, then we merged prices with them as well to have our main dataset ready for this project.

## Exploratory Visualization

After I have the dataset ready, I started visualizing it using matplotlib. In forecasting the most important aspect is the trends, we can see in these plots that it is always up – whether by location or items. First, we can see sales trend is up in both states and stores:

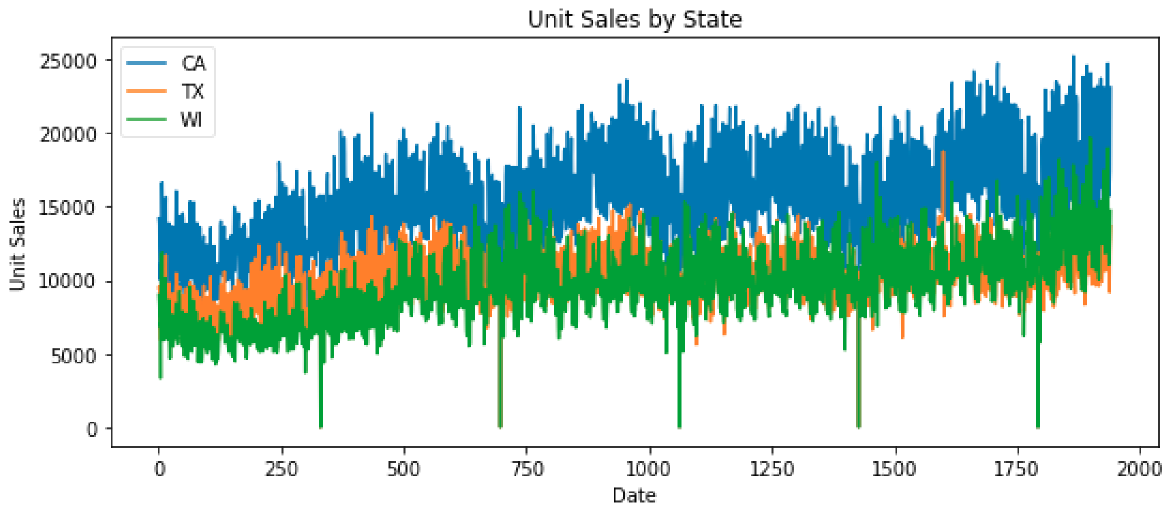


*Figure 2: Sales By State*

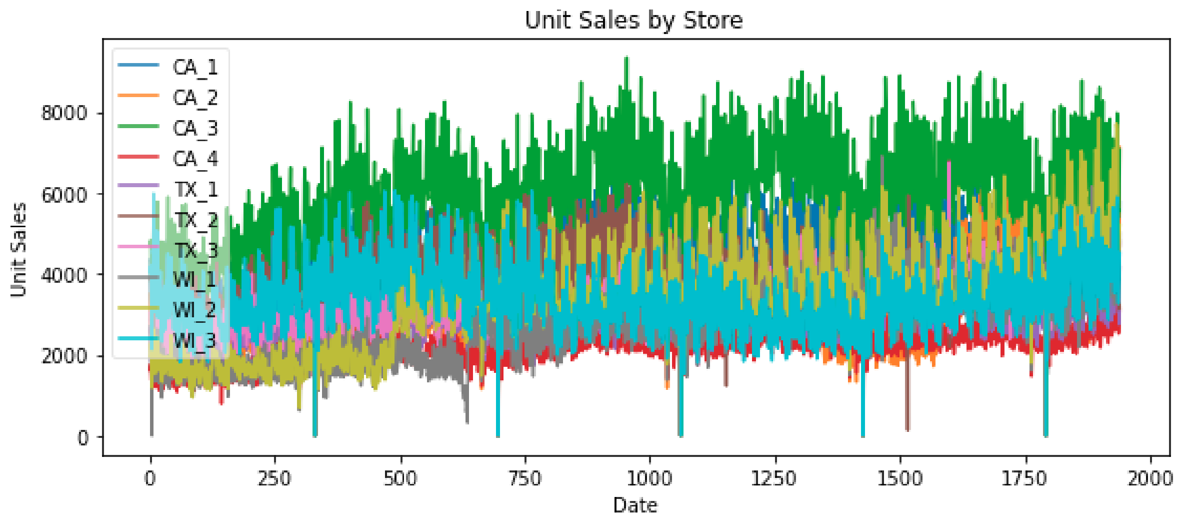


*Figure 3: Sales by Store*

Then we can see also that sales trend is up across all categories – specially in food category, while it is more stable in hobbies.

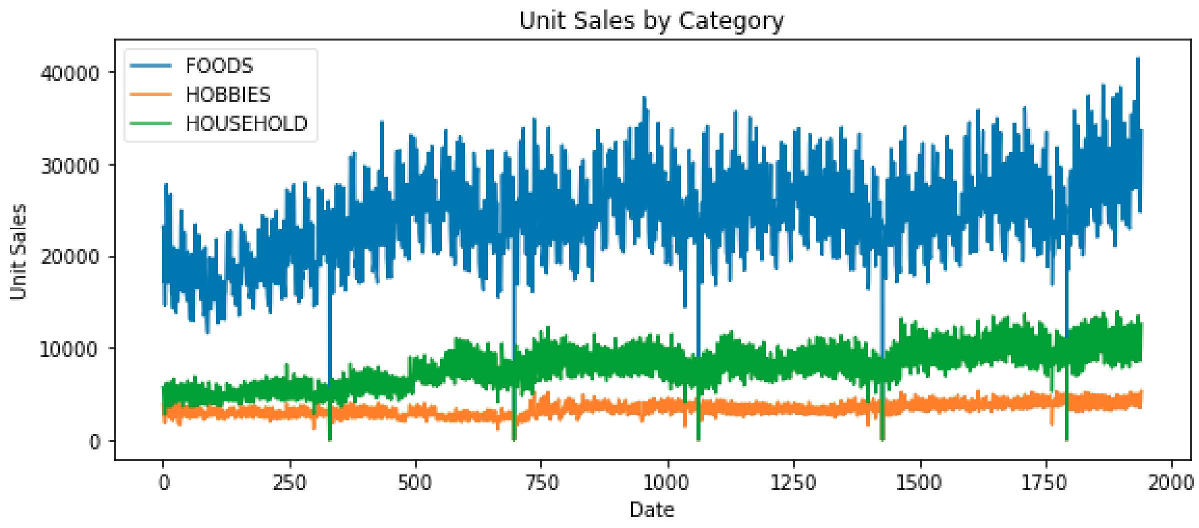


*Figure 4: Sales by Category*

Finally we can see categories trends reflects departments trends in this plot.

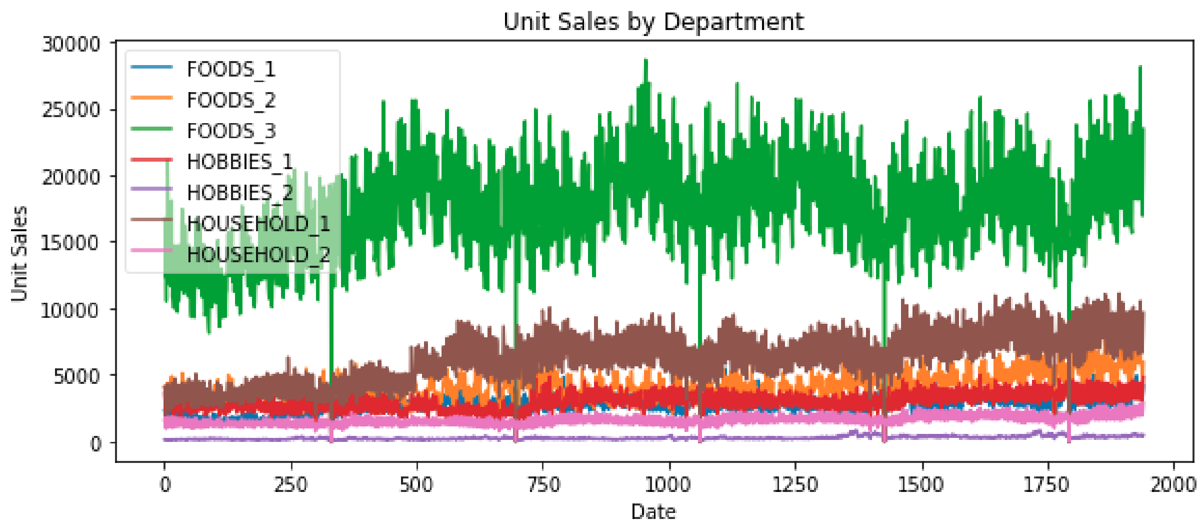


*Figure 5: Sales by Department*

This should be studied further to see whether the margins are rising as well and in which category to improve profitability.

## Algorithms and Techniques

The project done on aws sagemaker notebooks, this proved to be the best option for a project this size and complexity. I used built-in XGBoost algorithm to make predictions for forecasting the 28-day period after the given five years data.

AWS XGBoost built-in model's input must be in certain format: two CSV files, one for training and the other for validation. Both must have no index or header and the first column is the target column.

To do this I used pandas and numpy, which allowed me to do data manipulation very easily using aws sagemaker notebook instances that are large enough to handle this large dataset.

The on-demand notebook instance used was ml.m5.4xlarge, with 64GB in RAM and 16 vCPU cores, which handled the dataset of 3+GB very easily.

After melting and merging sales, calendar, and prices data; I added features that train the model on anticipating future values: lag features which are sales shifted by a given period of time. These values depend on where you stand in the forecasting horizon. The sales made on a particular day D can be considered as a 1-day lag if you're predicting one day ahead, or as a 28-day lag if you're predicting 28 days ahead. I used lagging of 28, 35, 42, 49, and 56 days and rolling median for 7,14,30,60,180 days. The final columns was as the following:

[demand, item_id, dept_id, cat_id, store_id, state_id, d, event_name_1, event_type_1, event_name_2, event_type_2, snap_CA, snap_TX, snap_WI, sell_price, date_day, date_week, date_month, date_year, date_week_m, lag_28, lag_35, lag_42, lag_49, lag_56, rolling_median_7, rolling_median_14, rolling_median_30, rolling_median_60, rolling_median_180]

## Benchmark

To compare the accuracy of my model, I will use the simplest form of forecasting as benchmark; which is the same demand as the last period.

# III. Methodology

## Data Preprocessing

Data preprocessing was the most challenging part of this project, it was a learning curve to know about how to make data ready for time-series forecasting – specially for large dataset like this one.

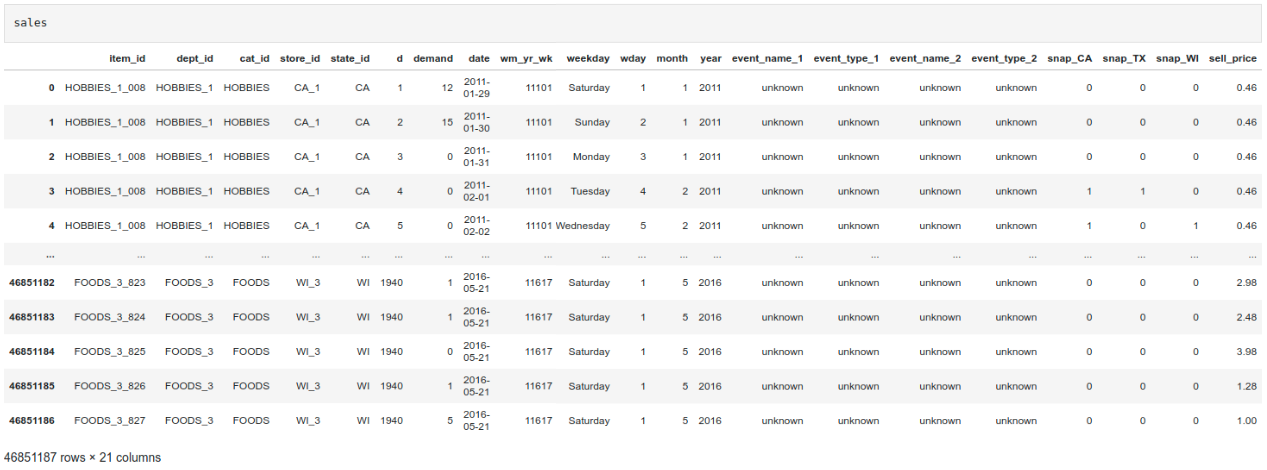
My goal is to have all three files in one large dataset to feed it to XGBoost. That requires merging sales, calendar, and prices in one dataframe.  Before we can merge calendar and prices data with sales data, we have to format sales data  so we can merge it with calendar and prices. To do this, we have to convert its format from wide to narrow by melting it – making each item duplicated for each day.

Another issue was categorical data in calendar file, before we can merge calendar to sales data, we have to encode categorical columns to make ready before making large data manipulation like merging it with sales data.

After having calendar and sales data ready we merged them, then we merged prices with them as well to have our main dataset ready for this project.

Then it was time for adding features that make our model understand the data better: lagging and rolling median features, it took long time to complete this because of the size of our dataset.

This was the resulting sales dataset:



```
sales
```

| | item_id | dept_id | cat_id | store_id | state_id | d | demand | date | wm_yr_wk | weekday | wday | month | year | event_name_1 | event_type_1 | event_name_2 | event_type_2 | snap_CA | snap_TX | snap_WI | sell_price |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | HOBBIES_1_008 | HOBBIES_1 | HOBBIES | CA_1 | CA | 1 | 12 | 2011-01-29 | 11101 | Saturday | 1 | 1 | 2011 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 0.46 |
| **1** | HOBBIES_1_008 | HOBBIES_1 | HOBBIES | CA_1 | CA | 2 | 15 | 2011-01-30 | 11101 | Sunday | 2 | 1 | 2011 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 0.46 |
| **2** | HOBBIES_1_008 | HOBBIES_1 | HOBBIES | CA_1 | CA | 3 | 0 | 2011-01-31 | 11101 | Monday | 3 | 1 | 2011 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 0.46 |
| **3** | HOBBIES_1_008 | HOBBIES_1 | HOBBIES | CA_1 | CA | 4 | 0 | 2011-02-01 | 11101 | Tuesday | 4 | 2 | 2011 | unknown | unknown | unknown | unknown | 1 | 1 | 0 | 0.46 |
| **4** | HOBBIES_1_008 | HOBBIES_1 | HOBBIES | CA_1 | CA | 5 | 0 | 2011-02-02 | 11101 | Wednesday | 5 | 2 | 2011 | unknown | unknown | unknown | unknown | 1 | 0 | 1 | 0.46 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **46851182** | FOODS_3_823 | FOODS_3 | FOODS | WI_3 | WI | 1940 | 1 | 2016-05-21 | 11617 | Saturday | 1 | 5 | 2016 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 2.98 |
| **46851183** | FOODS_3_824 | FOODS_3 | FOODS | WI_3 | WI | 1940 | 1 | 2016-05-21 | 11617 | Saturday | 1 | 5 | 2016 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 2.48 |
| **46851184** | FOODS_3_825 | FOODS_3 | FOODS | WI_3 | WI | 1940 | 0 | 2016-05-21 | 11617 | Saturday | 1 | 5 | 2016 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 3.98 |
| **46851185** | FOODS_3_826 | FOODS_3 | FOODS | WI_3 | WI | 1940 | 1 | 2016-05-21 | 11617 | Saturday | 1 | 5 | 2016 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 1.28 |
| **46851186** | FOODS_3_827 | FOODS_3 | FOODS | WI_3 | WI | 1940 | 5 | 2016-05-21 | 11617 | Saturday | 1 | 5 | 2016 | unknown | unknown | unknown | unknown | 0 | 0 | 0 | 1.00 |

46851187 rows × 21 columns

### Implementation

After I have the dataset ready for training, I started using aws built-in XGBoost model to build the forecasting model for our project. Also, I generated same-period prediction for bench-marking our model.

### Refinement

After some experiments with the model, I chose the objective as: rmse to improve the model accuracy, which was good from the beginning.

## IV. Results

### Model Evaluation and Validation

After having the model trained, I did batch transform job to determine the RMSE. The RMSE for the model using test data was 6. Because there a lot of data points I needed to try how accurate the model for individual items, so I deployed the model to make inferences via a web app. The resulting predictions was accurate most of the time.

### Justification

After having both model and benchmark ready, I calculated the rmse for both. The model was better than the benchmark by more than four-folds.

Benchmark RMSE : 26 Model RMSE: 6

This validates how much accuracy in prediction we can gain from using machine learning models in forecasting sales.

## V. Conclusion

### Free-Form Visualization

Instead of visualization, I made a web app that predict the demand for individual items using the deployed model.

### Reflection

After completing this project, I realized how much machine learning model can make business processes better by having accurate predictions. This requires a lot of work, collecting and saving large amount of data, processing the data to be digestible for these data hungry machine learning algorithms, visualizing the data to understand it to select the best algorithm, then testing these algorithms against appropriate benchmarks.

### Improvement

There are many improvements can be implemented to this project, including trying new algorithms like lightGBM and CatBoost, which shown very promising results. Also, the web app can be improved by having a better user interface that gives more intuitive data entry. Another important improvement is to make api that interfaces with walmart procurement to make automated purchasing and distribution decisions.