Capstone Project

Machine Learning Engineer Nanodegree

Bryan Rosales
May 1st, 2021

Definition

Project Overview

Inventory forecasting is a challenge that many companies face and struggle to maintain an ideal inventory levels every day. Either being overstock or out of stock of any product are costly for a company. On one hand, overstock of an item means carrying more product than a company is able to sell and it is associated with over costs in storage, handling and cash flow which impact directly the profit of a business. On the other hand, out of stock could be more critical since the company runs out of a specified item where it is demanded by customers producing not only the loss of revenue, but also affecting customer satisfaction which most of the time results in a higher churn rate.

Problem Statement

The goal of this project is to create a forecasting solution to predict sales up to 4 months in the future using Machine Learning algorithms and techniques. Initially the analysis will be focus on top 6 best seller products, but extended to more products according to results. Forecasting 4 months of sales, the company will be able to planning containers in advance allowing allocate cash flow and most important to avoid running out of products with high demand.

Metrics

The metrics to use in this project will be MAE (Mean Absolute Error).

$$MAE = \frac{1}{n} \sum |e_t|$$

MAE is a metric no scaled, therefore it is a very simple to use and understand giving a quick point of comparison against our benchmark model which will be defined in Benchmark section. The disadvantage might be the fact that the metric does not provide a way to realize if the algorithm is over or under shooting the forecasted quantity.

Analysis

Data Exploration

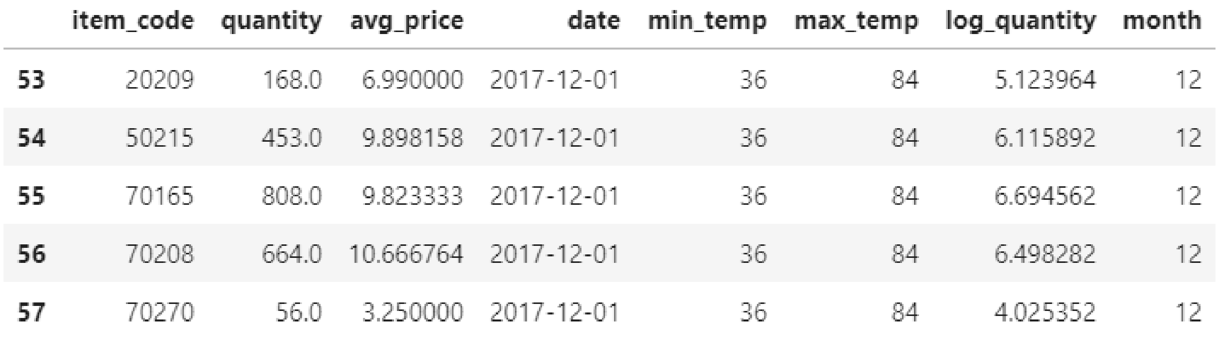
The dataset used in this project contains daily sales data transactions for around 400 items of the portfolio, but for this analysis purposes we will use top 6 best seller products for the company. Then, we will use average monthly sales to build a dataset with 40 months for each time series which represent the recent behavior of sales. At the end, our dataset will have a total of (6 x 40) 240 rows or 40 rows for each product.

In addition to the number of rows, the dataset includes 7 features as follow:

- “item_code”: Describe the internal reference of the product.
- “quantity”: Number of bottles sold in each sales transaction.
- “avg_price”: Correspondent to the monthly average price of the product.
- “date”: Date when the sales transaction was made (Only describe year and month since data points are grouped by month).
- “min_temp”: Minimum temperature to the month in Phoenix, AZ metropolitan area.
- “max_temp”: Maximum temperature to the month in Phoenix, AZ metropolitan area.
- “log_quantity”: Transformation logarithmical for quantity feature.

The 6 items chosen for this analysis are the following wines:

1. 70270, Le Contesse, Prosecchino Brut, 187ml
2. 20209, Alverdi, Pinot Grigio
3. 70165, Santome, Prosecco Extra Dry
4. 50215, Carpineto, Dogajolo Rosso
5. 70208, Italo Cescon, Pinot Grigio
6. 70271, Data Points:43, Le Contesse, Pinot Noir Rose Brut

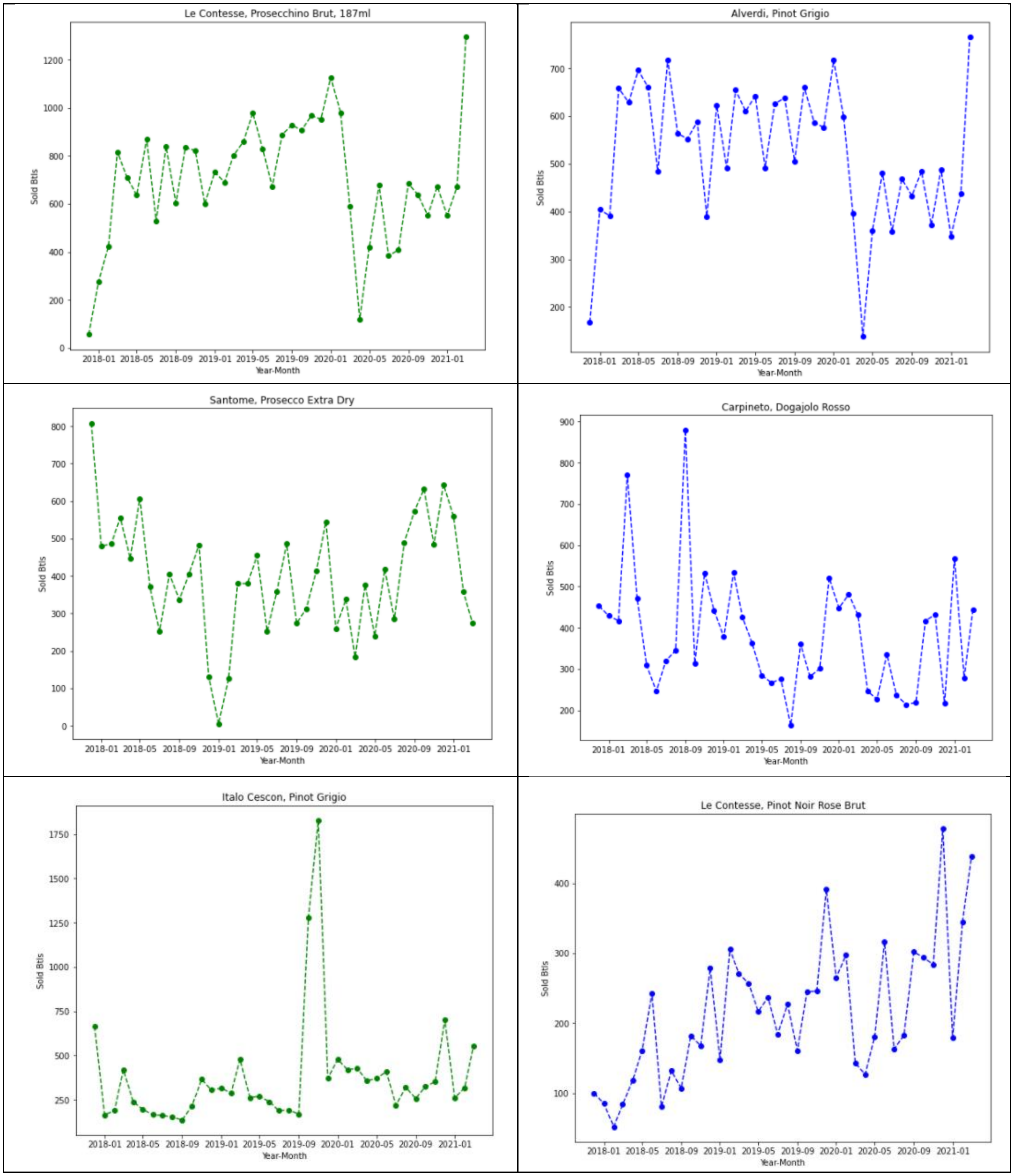


	item_code	quantity	avg_price	date	min_temp	max_temp	log_quantity	month
53	20209	168.0	6.990000	2017-12-01	36	84	5.123964	12
54	50215	453.0	9.898158	2017-12-01	36	84	6.115892	12
55	70165	808.0	9.823333	2017-12-01	36	84	6.694562	12
56	70208	664.0	10.666764	2017-12-01	36	84	6.498282	12
57	70270	56.0	3.250000	2017-12-01	36	84	4.025352	12

Fig. 1 Samples Data Points

Exploratory Visualization

The following are the plots for each product time series:



Le Contesse, Prosecchino Brut, 187ml

Alverdi, Pinot Grigio

Santome, Prosecco Extra Dry

Carpineto, Dogajolo Rosso

Italo Cescon, Pinot Grigio

Le Contesse, Pinot Noir Rose Brut

Fig. 2 Time Series by Product

Important facts:

- In general, the Series are very irregular and do not show a clear tendency. As an extraordinary influence the year 2020 was completely atypical because of pandemic.
- Most of the series show low sales quantities after March-2020 till June-2020 which is explained by Covid-19 pandemic. During that period the economy was shut down and restaurants were closed to dine in which impacted directly sales.
- In general, the year 2020 was affected by pandemic and it could be an obstacle to generalization of the model.
- The Product Santome Prosecco in Jan-2019 was out of stock because of lack of supply by vendor.
- Italo Cescon Pinot Grigio displays a higher demand than usual, during October and November 2019 as a consequence of running a promotion in retail segment. There were more promotions and programs where the company has not a clean historical record.

Algorithms and Techniques

To solve the forecasting problem, the algorithm chosen is Temporal Fusion Transformer which is part of Pytorch Forecasting package. It is able to manage several features and subcategories like products and subproducts, stores by locations and sales territories, which benefits our case of study.

Pytorch Forecasting yields excellent advantages to accelerate the training process. One of them is the Class TimeSeriesDataSet where all the variables and targets are set to create a dataset that speaks the Trainer language. Therefore, all the features to be included in the model are classified as categorial and reals ('continue' variables), and setting the number of predictions points and data points to use in the training stage. In our case, we used 36 months for training and 4 months for validation ('prediction'). After that, the TimeSeriesDataSet is converted to Dataloader and fitted into the model.

Moreover, to tune the model in a fast way, the functionality 'optimize_hyperparameters' can be used, which allow us to define ranges for each parameter resulting in the best values to start the training process. After running 100 trails, the results obtained are:

- 'gradient_clip_val': 1.5784,
- 'hidden_size': 64,
- 'dropout': 0.3,
- 'hidden_continuous_size': 24,
- 'attention_head_size': 3,
- 'learning_rate': 0.3

Neural Network Architecture

- ✓ Number of LSTM Layers: 2
- ✓ Number of Hidden Layers: 12

- ✓ Number of Hidden Continues Layers: 8

For training the algorithm, I am using batch size of 64 which is managed by the Dataloader. Only CPU was used to this project, but GPU and TPU might be used rending a better training time. Using the CPU and 50 epochs the model was trained in less than 90 seconds since the number of data points is relatively low (36 data points x 6 Time Series = 216 data points or months)

Benchmark

The initial benchmark for the project was a Simple Moving Average 4 Months, which is being currently used for the company to forecast sales. The idea is to compare MAE metric against the values obtained when using the model.

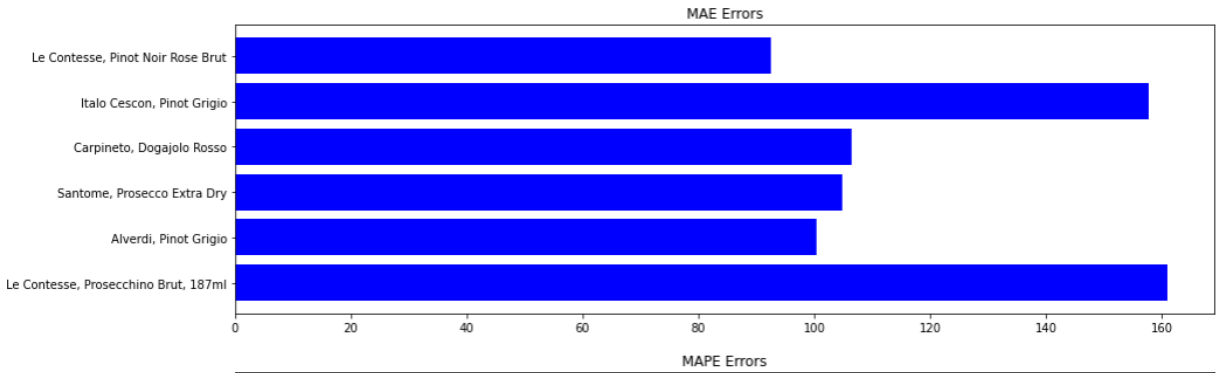


Fig. 3 MAE

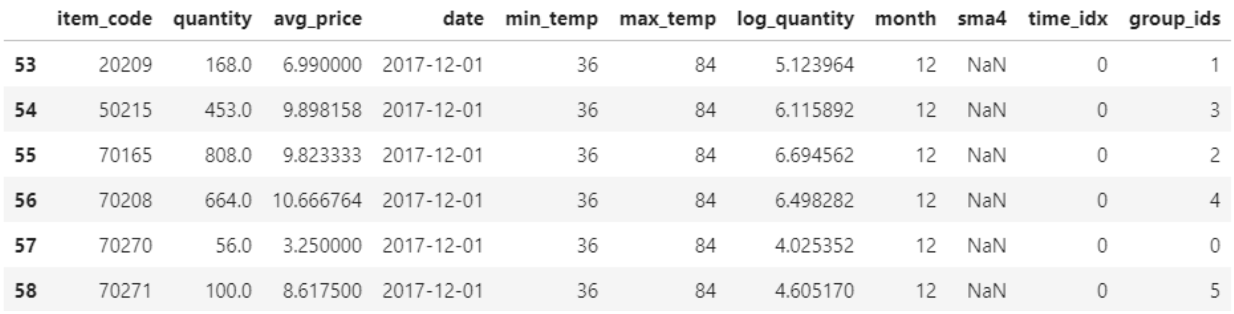
Methodology

Data Preprocessing

The following are the steps performed in order to get the dataset clean and ready for modelling:

1. Cleaning the datasets to remove useless columns, rows with Nan's values and clean the item code extracting the numeric part only.
2. Transformation from daily sales to monthly sales, applying a "groupby" operation.
3. Addition of new features like min and max temperatures and transformation of target "quantity" to "log_quantity" applying Log function.
4. Imputations and adjustment to smooth the impact of Covid-19 during year 2020.

All the previous tasks were performed in the notebook "1_Data_Preprocessing". Another important step that was not mentioned as part of the data preprocessing is the calculation of Simple Moving Average 4 Months which is part of the same notebook. The final dataset looks like:



	item_code	quantity	avg_price	date	min_temp	max_temp	log_quantity	month	sma4	time_idx	group_ids
53	20209	168.0	6.990000	2017-12-01	36	84	5.123964	12	NaN	0	1
54	50215	453.0	9.898158	2017-12-01	36	84	6.115892	12	NaN	0	3
55	70165	808.0	9.823333	2017-12-01	36	84	6.694562	12	NaN	0	2
56	70208	664.0	10.666764	2017-12-01	36	84	6.498282	12	NaN	0	4
57	70270	56.0	3.250000	2017-12-01	36	84	4.025352	12	NaN	0	0
58	70271	100.0	8.617500	2017-12-01	36	84	4.605170	12	NaN	0	5

Fig. 4 Final Dataset

Implementation

As mentioned previously the algorithm chosen was Temporal Fusion Transformer and the logic to implement the model follows the next pipeline:

- Create two TimeSeriesDataSet object to divide training and validation sets. Training was defined to encode between 24 and 36 months for each item whereas validation uses a range of 2 to 4 data points per product. In orders words, the most recent 4 months are used to validate the model.
- Implement Dataloader objects for training and validation sets in order to pass the Dataloader to the Trainer object.
- Make use of "optimize hyperparameters" to find a decent starting point and help the model to fit the data correctly quickly.
- Train the model using the hyperparameters obtained from last step. The loss function used was MAE (Mean Absolute Error).
- Finally best model is selected.

Refinement

Initially the model was very inaccurate yielding a val_loss greater than 200. Then, a technique to optimize parameters was used "optimize_hyperparameters" giving a critical starting point to improve the model to val_loss=161 and loss=96. After that, with around 120 training jobs, the model did not improve significatively.

Moreover, there are more points that might be addressed in order to improve the time series, but they are out of the scope of this project since they require time and some research inside the company. For instance, creating a feature to indicate when a product was under a special program or promotion which push up the sales quantities for a specific item.

Results

Model Evaluation and Validation

To evaluate the model, a validation set of 4 data points by product was used. Then, the MAE metric was compared in 2 ways:

- Overall, MAE value for all items

SMA4	Temporal Fusion Transformer
120.44	127.19

Table 1.0 Overall MAE comparison

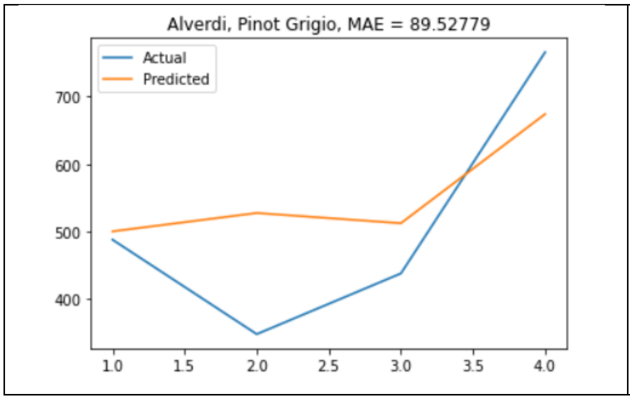
- MAE value by item

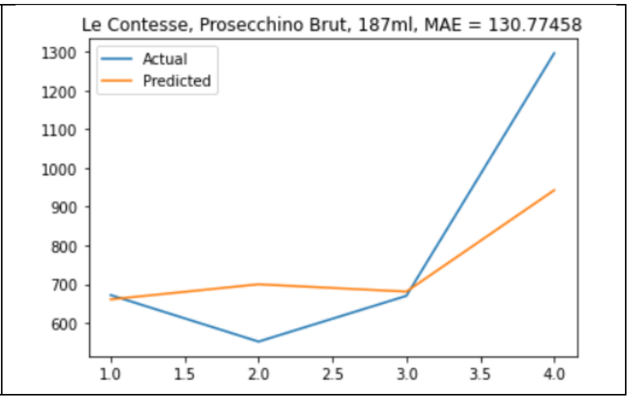
Item_code	Item_name	SMA4	Temporal Fusion Transformer
70270	Le Contesse, Prosecchino Brut, 187ml	161.00	130.77
20209	Alverdi, Pinot Grigio	100.31	89.52
70165	Santome, Prosecco Extra Dry	104.75	169.82
50215	Carpineto, Dogajolo Rosso	106.43	115.80
70208	Italo Cescon, Pinot Grigio	157.75	118.43
70271	Le Contesse, Pinot Noir Rose Brut	92.43	138.82

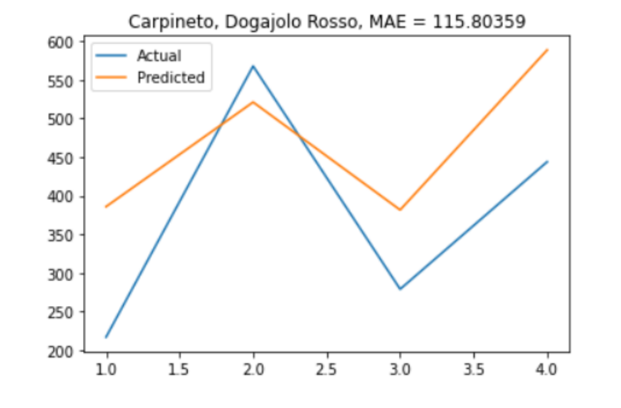
Table 1.1 MAE comparison by Item

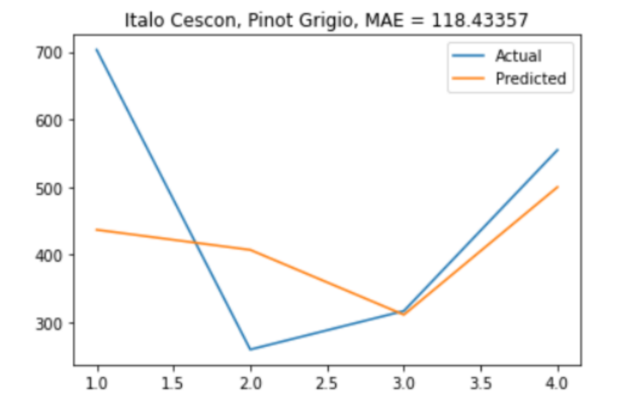
According to previous table the TFT model is able to perform better than SMA4 in items 70270, 20209, and 70208 showing a significant accuracy improvement. However, for the rest of items the performance of TFT was worse than SMA4.

Here some graphs with the predictions of TFT model for some items:









Justification

Even thought the TFT model was not able to improve the benchmark in some of the products, the model is useful as an additional input when making inventory purchasing and sales estimations. Also, it could be used in conjunction with SMA4 having two different models which can help to make a better inventory decision. As an advantage this TFT model can predict till 4 months in future allowing enough time to plan containers with lead time of 2.5 months.

As I mentioned previously, the model might be improved if more engineering hours are dedicated to focus on year 2020 since Covid-19 represents an atypical period which being the most recent data impacts directly in the performance and generalization of the model.

Moreover, other enhancements are required in order to get better results. For instance, it is required to have features that are able to give information about special promotions or programs where sales are sharply increased. Similarly, when a product is out of stock for any reason, there should be a way through a feature to denotate the lack of the product during a specific period.

These features not only may help the model to identify patterns, but also to generalize and output better results.

References

Jan Beitner. PyTorch Forecasting Documentation — pytorch-forecasting documentation. (2020). PyTorch Forecasting. https://pytorch-forecasting.readthedocs.io/en/latest/index.html

William Falcon. (2018–2021). PyTorch Lightning Documentation — PyTorch Lightning 1.4.0dev documentation. PyTorch Lighting. https://pytorch-lightning.readthedocs.io/en/latest/index.html