DEMAND FORECASTING GROCERY STORES

When forecasting time series, the standard option comes down to statistical learning methods such as ARIMA, and several other models.
However, in case of intermittent demand and forecasting multiple time series at once, statistical learning methods fail to provide a high level of accuracy and can sometimes become computationally expensive. Deep learning algorithms can be applied to tackle both the issue of forecasting intermittent sales and the problem related to the computational cost.

Univariate vs Multivariate Time Series Models

Univariate Time Series models are models useful to forecast one target variable (e.g., sales). They simply put a mathematical formula on the past and try to project it to the future [1].
But in some cases, you want to predict multiple related variables at the same time. For example, on social media, you may want to forecast how many likes you will receive, but also how many comments. The first possibility for treating this is to build two models: one model for forecasting likes and another one for forecasting the number of comments. But some models allow benefiting from a correlation between target variables. These are called **multivariate models**, and they use the correlation between target variables in such a way that the forecast accuracy improves from forecasting the two at the same time.
However, it is important to underline that multiple models are sometimes more performant than one model that makes multiple predictions.

1. Statistical Learning Methods

In this section the order of exposition depends on the complexity of the method used. The Naïve Method, Auto Regression and Moving Average Method can be used in case of stationary variables. The stationarity of a time series means that a time series does not have a (long-term) trend: it is stable around the same average. If not, you say that a time series is nonstationary.

- ***Naïve Method***
  This method is also known as the last value method. It is the simpler methods that we can use. The forecast for the next time point is equal to the last observed value for the time series. Historical data are not relevant. This approach works well for forecasting products that are innovative or are changing their features at a rapid rate.

- ***Auto Regression (AR)***
  The prediction of output value is based on a linear combination of input values.
  A peculiar aspect of Auto Regression models is that the lag values of same variable are used as input values. Indeed, the term “autoregression” indicates that it is a regression of the variable against itself. For instance, to predict the temperature of tomorrow, you can use the temperature of today and yesterday to build a linear model equation.

  The AR model takes in one argument, p, which determines how many previous time steps will be inputted.

The order p can be determined by looking at the partial autocorrelation function (PACF). The PACF gives the partial correlation of a stationary time series with its own lagged values, regressed of the time series at all shorter lags.

- ***Moving Average Method (MA)***
  Rather than using past values of the forecast variable in a regression, a moving average model uses past forecast errors in a regression-like model.
  In this method, the most recent demand patterns are used to forecast sales in the future. For example, for a 4-month moving average, the data for the past 4 months is used to create a forecast for the next month.
  - *simple moving average method*: equal weight is assigned to the entire window of the demand forecast.
  - *weighted moving average method*: more weight is assigned to the demand that occurs recently.
    For example, in a 4-month weighted moving average, the most recent month will have a higher weight assigned as compared to the 3rd most recent one.

- ***ARIMA Model***
  The ARIMA model is a statistical forecasting method that predicts the future values of the time series based on historical data. It is obtained by combining differencing with autoregression and a moving average model. It is described by three parameters (p, d, q):
  - 'p' is the autoregressive order: the number of lag observations in the model, aka the lag order
  - 'q' is the moving average order: the size of the moving average window
  - 'd' is the order of differencing: the number of times that the raw observations are differenced. 'd' is used to make the time series stationary. ARIMA models describe the autocorrelation within the time series.
    - Autoregressive Process: the present value is obtained by a weighted average of its past processes. The parameter p can be identified using the Partial Auto Correlation function (PACF) plot. We can expect the PACF function to show a sharp decline in the plot crossing the confidence interval band, depicting that we no longer need the previous lags to describe the model as an Autoregressive model.
    - Moving Average Process: the present value is obtained by a linear combination of past errors. The errors for the moving average model are independently and identically distributed (i.i.d). To find the order of the moving average process we can exploit the Auto Correlation Function (ACF) plot. We can expect the PACF plot to have a strong correlation with its closest lags, and then a sharp fall that crosses the confidence band when no previous lags describe the present values of the series.

- ***SARIMA***
  It is an ARIMA model that considers also the seasonal.

- ***Croston Forecasting***
  This method is useful to forecast the intermittent demand of products. A time series is considered intermittent if many of its values are zero and the gaps between non-zero entries are not periodic.
  Croston method can be summarized in three steps:

- Evaluate the average demand level *a* when there is a demand occurrence *d* as follows:

$$\text{if } d_t > 0, \text{ then } a_{t+1} = \alpha d_t + (1-\alpha) a_t$$

$$\text{if } d_t = 0, \text{ then } a_{t+1} = a_t$$

where we use a learning parameter α ($0 < \alpha < 1$) to give more or less importance to the most recent observations or the historical ones.

- Estimate the time between two demand occurrences *p* (for periodicity), and the time elapsed since the previous demand occurrence *q*. We will only update p when we have a demand occurrence

$$\text{if } d_t > 0, \text{ then } p_{t+1} = \alpha q + (1-\alpha) p_t$$

$$\text{if } d_t = 0, \text{ then } p_{t+1} = p_t$$

- Forecast the demand using the mathematical formulation of the Croston model:

$$f_{t+1} = \frac{a_t}{p_t}$$

2. Neural Networks

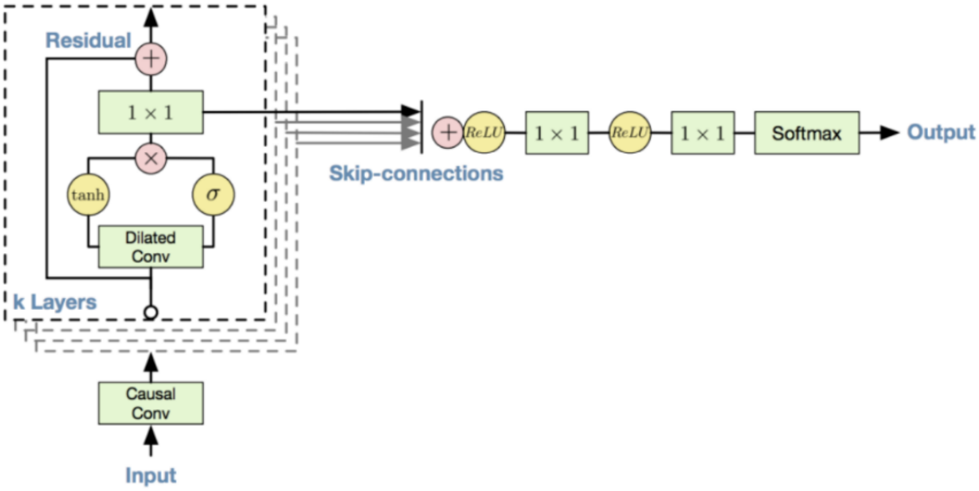
- ***Poisson Neural Network (PNN)***

  PNN is the combination of a Poisson regression model and NN model. PNN assumes that the predicted value follows the Poisson distribution. PNN uses the same structure as the ordinary NN, but adopts the exponential activation function f (x) = e^x [2].

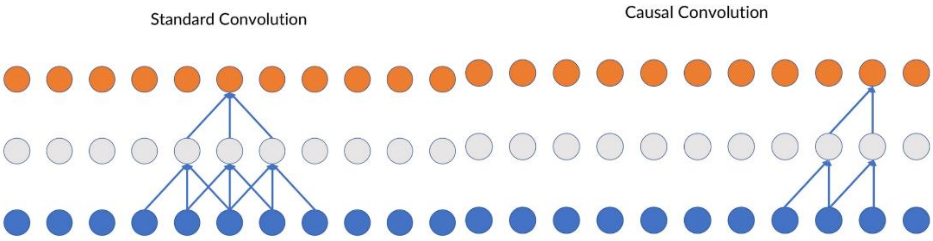
- ***Convolutional Neural Networks (CNN)***

  ○ ***Wavenet***

  It is a CNN architecture, that uses stacked convolution layers to model the conditional probability distribution. The image below shows the architecture [3].



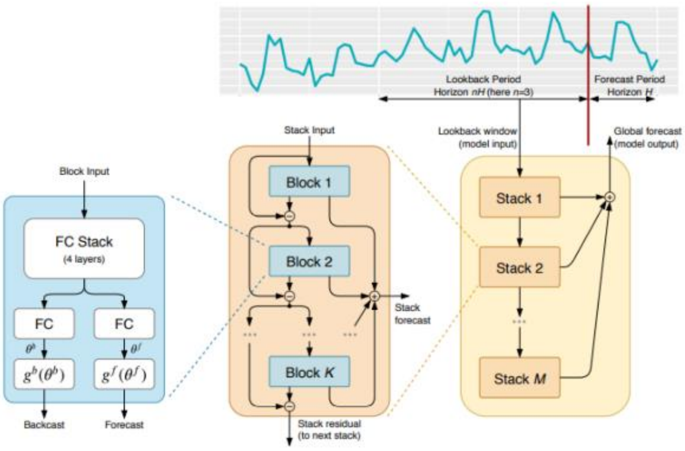
An important aspect is that Wavenet uses both causal and dilated convolutions.
Causal convolution is used to deal with the time ordering of samples and guarantee that the model at timestep t cannot depend on any future values $x(t+1)$. Indeed, while standard convolution does not take the direction of convolution into account, causal convolution shifts the filter in the right direction



To increase the receptive field size, thus increasing the amount of information from past samples, Wavenet makes use of dilated causal convolutions, which skip over inputs while convolving. In dilated convolution the filter only accesses every nth element, allowing the model to have a larger receptive field size, using the same number of layers, and the same number of parameters

- ***NBEATS***

This model is defined as a deep neural architecture based on backward and forward residual links and a deep stack of fully connected layers [4].
Indeed, unlike RNN, it takes an entire window of past values, and compute many forecast timepoints values in a single pass. For doing so, it uses extensively fully connected layers.



More precisely, it consists in several blocks connected in a residual way: the first block tries to model past window (backcast) and future (forecast) the best it can, then the second block tries to model only the residual error of the past reconstruction done by the previous block (and also updates the forecast based only on this error) and so on. Such residual architecture allows to stack many blocks without risk of gradient vanishing, and also has the advantages of boosting technique: the forecast is the sum of predictions of several blocks, where the first block catches the main trends, the second specializes on smaller errors and so on.

- ***MQCNN***

  It belongs to the category of “Sequence-to-Sequence” models.
  Sequence-to-sequence” models are composed of two main components: the encoder network, that encodes information about context interval in a latent state, and the decoder network, which generates the forecast of the prediction interval by combining the latent information with the features from the prediction range[5].
  In MQCNN, the encoder is a Convolutional Neural Network and the decoder a Multi-layer Perceptron. It is important to underline that the output is not a distribution of probability but the quantiles of this distribution.

- ***Recurrent Neural Network (RNN)***

  - ***MQRNN***

    The model is the same than MQCNN excepting that the encoder is a Recurrent Neural Network.

  - ***DeepAR***

    DeepAR is an algorithm by Amazon that combines Deep Learning techniques with probabilistic forecasting [6]. It forecasts univariate or multivariate time series using RNNs. The main advantage of DeepAR is that it fits a single model on all the time series at the same time. It is strictly connected to the covariance of the series and it is designed to benefit from correlations between multiple time series, and therefore it is a great model for multivariate forecasting.
    Another advantage of DeepAR is that, like any other deep learning model, it offers an automatic feature engineering. The basic idea is simply to feed the time-series and the model can generate features through LSTM cells to produce a forecast of a desired length.
    However, DeepAR does not directly forecast a point value for each time point, but it uses the output from the RNN to predict a standard deviation and a mean of future probability distribution.

- ***Transformers***
  Transformers’ architecture enables models to propagate very important information over long sequences and thus they are generally very good at capturing the long-term seasonal behaviors and dependencies [7]. Transformer-based models do neither include any RNN-like networks nor process data in any specific order, but they simply rely on self-attention mechanisms to learn dependencies in the sequence. For example, a model trained to forecast prices based on the data of the past week would not necessarily process past prices in order, but it might first start with the data of the last Friday before last Monday.
  One of the advantages of using attention mechanisms is that they can reduce the training time. In fact, as the model is now able to process calculations at the same time rather than in sequential steps, it can parallelize the calculations.

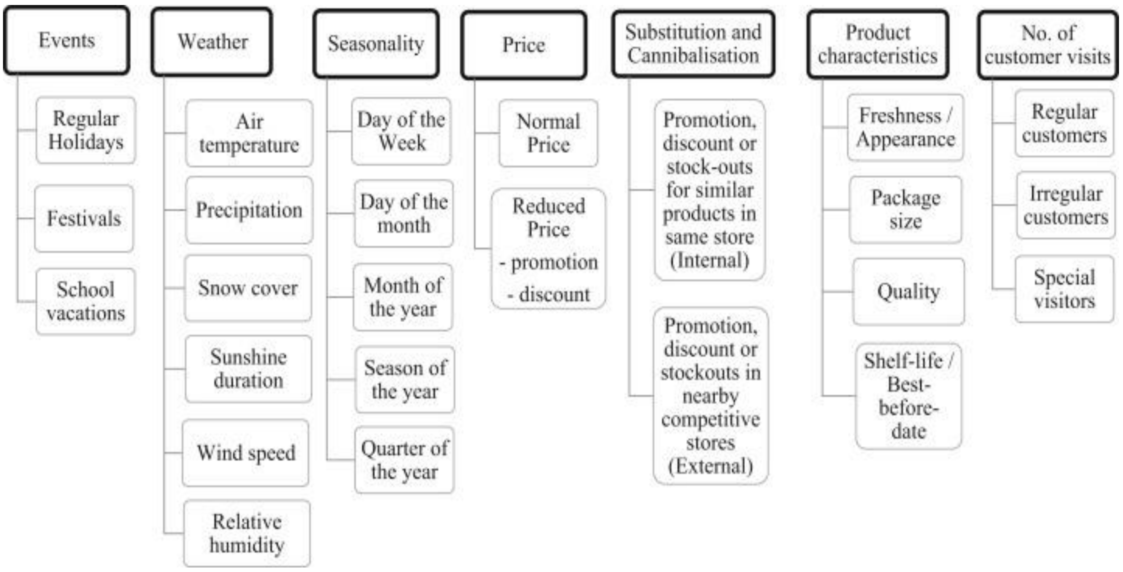
Finally, there is one element worth mentioning: in order to make predictions one step at a time using only previous data points, the model masks all data points that come after the data point currently being predicted, and this prevents the model from cheating.

- ***Prophet***

  The Prophet model by Facebook is not exactly a model, but rather an automated procedure for building forecasting models.
  A quote from the developers explains the goal of Facebook's Prophet: "*We use a simple, modular regression model that often works well with default parameters, and that allows analysts to select the components that are relevant to their forecasting problem and easily make adjustments as needed. The second component is a system for measuring and tracking forecast accuracy, and flagging forecasts that should be checked manually to help analysts make incremental improvements"[8].*
  Prophet is a modular regression model with interpretable parameters [9].
  It is a sum of trend g(t), seasonality s(t), and holidays h(t) components as follows: $z(t) = g(t) + s(t) + h(t) + \varepsilon_t$, where ε_t is a normally distributed error term. The seasonality model s(t) is a general Fourier series with a regular period that we manually set.

INFLUENCING FACTORS

The paper [10] presents the demand influencing factors, that the authors collected based on the literature review and opinions from the store managers in order to understand what the variables are useful to build a model *to forecast daily sales of a perishable food.*
The influencing factors are summarized in the figure below.



The demand influencing factors are classified into events, weather, seasonality, price, substitution and cannibalization, product characteristics and number of customer visits.

- the price and the product characteristics are internal forces, which are controllable.
- substitution and cannibalization are partially internal, which cannot be controlled entirely.
- the other factors are external forces, which are uncontrollable.

For each influencing factors, they did a categorization:

- The events are categorized into regular holidays (except festivals), festivals and school vacations.
- The weather is categorized into air temperature, precipitation, snow cover, sunshine duration, windspeed and relative humidity.
- The seasonality is categorized into day-of-the-week (weekly seasonality), day of the month (monthly seasonality), month-of-the year (yearly seasonality), yearly seasons and yearly quarters.
- The price is categorized into normal price and reduced price. The price reduction is classified into promotion and discount.
  An observation is that the price reduction of certain product may sometimes produce cannibalization effect on similar type of products in the same store and/or in the competitive stores.
- The substitution and cannibalization are categorized into promotion, discount or stock-out for similar type of products in the same store and/or in the competitive stores.
  The promotion for similar type of products (in the same store and/or in the competitive stores) usually produces cannibalization effect on the selected product. The discount for similar type of products (in the same store and/or in the competitive stores) produces both substitution and cannibalization effects on the selected product. Likewise when there are stock-outs for similar type of products (in the same store and/or in the competitive stores), the selected product acts as a substitute.
- The product characteristics are classified into product freshness/appearance, package size, quality and shelf-life/best-before-date. In particular:
  - The product freshness/appearance depends on the perceptions by the store managers (while discounting or discarding) and the consumers.
  - The product quality implies the physical condition of a product like damage, disease and mold.
  - The shelf-life indicates the life of a product after its arrival in the store.
  - The number of customer visits is categorized into different type of customers visiting the stores such as regular customers, irregular customers, and special visitors (e.g., tourists).

3. DATASETS

The following list contains the dataset we found so far:

- M5 Walmart https://www.kaggle.com/competitions/m5-forecasting-accuracy/overview
- Rossman Store Sales https://www.kaggle.com/competitions/rossmann-store-sales
- Tesco https://www.nature.com/articles/s41597-020-0397-7/
- Walmart https://www.kaggle.com/datasets/yasserh/walmart-dataset
- Favorita Grocery Sales https://www.kaggle.com/c/favorita-grocery-sales-forecasting
- Supermarket sales https://www.kaggle.com/datasets/aungpyaeap/supermarket-sales

Bibliography

[1] Joos Korstanje, Advanced Forecasting with Python: With State-of-the-Art-Models Including LSTMs, Facebook's Prophet, and Amazon's DeepAR,

[2] H. Huang, M. Jiang, Z. Ding and M. Zhou, "Forecasting Emergency Calls With a Poisson Neural Network-Based Assemble Model," in IEEE Access, vol. 7, pp. 18061-18069, 2019, doi: 10.1109/ACCESS.2019.2896887.

[3] https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio

[4] Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. arXiv preprint arXiv:1905.10437, 2019

[5] Wen, R., Torkkola, K., and Narayanaswamy, B. (2017). A multi-horizon quantile recurrent forecaster. arXiv preprint arXiv:1711.11053.

[6] Salinas, D., Flunkert, V., & Gasthaus, J. (2017). DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. arXiv e-prints, art. arXiv preprint arXiv:1704.04110.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. CoRR, abs/1706.03762, 2017.

[8] https://peerj.com/preprints/3190/

[9] http://lethalletham.com/ForecastingAtScale.pdf

[10] Arunraj, N. S., & Ahrens, D. (2015). A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. International Journal of Production Economics, 170, 321–335. https://doi.org/10.1016/j.ijpe.2015.09.039