Annals of Operations Research (2022) 313:945–964
https://doi.org/10.1007/s10479-021-04253-3

S.I.: RISK MANAGEMENT DECISIONS AND VALUE UNDER UNCERTAINTY

Concurrent neural network: a model of competition between times series

Rémy Garnier[1]

Accepted: 30 August 2021 / Published online: 18 September 2021

Abstract
Competition between times series often arises in sales prediction, when similar products are on sale on a marketplace. This article provides a model of the presence of cannibalization between times series. This model creates a "competitiveness" function that depends on external features such as price and margin. It also provides a theoretical guaranty on the error of the model under some reasonable conditions, and implement this model using a neural network to compute this competitiveness function. This implementation outperforms other traditional time series methods and classical neural networks for market share prediction on a real-world data set. Moreover, it allows controlling underprediction, which plagues traditional forecasts models.

Keywords High dimensional times series · Multivariate count times series · Non-stationnary times series · Sales forecasting · Cannibalization · Competition modeling · E-commerce data

1 Introduction

Forecasting multiple time series is a useful task, which has many applications in finance (Tay and Cao 2001), load forecast (Zhang et al. 2018), health care (Lin 1989), retail operations (Ren et al. 2020) or in supply chain management (Alon et al. 2001; Vandeput 2021). This is however a complex task because the number of possible interactions between times series grows likes the square of the number of time series. For simple models such as Vector Autoregression (VAR) (Lutkepohl 2005), it means that the number of times series should remain small. That is why practitioners tend to use tricks like sparsity (Davis et al. 2016). However, most of the time, they forecast each series individually (see classical models in (Vandeput 2021) for instance), or build an auto-regressive framework treating each point of data in a similar way (Bandara et al. 2019; Garnier and Arnaud 2019). However, in both cases, these methods neglect the possible interaction between time series.

✉ Rémy Garnier
remy.garnier@ext.cdiscount.com

[1] UFR Sciences et Techniques, Universite de Cergy-Pontoise, Cergy-Pontoise, France

In this article, we are interested in a specific type of interaction between time series: cannibalization. Product cannibalization has been defined as "the process by which a new product gains sales by diverting sales from an existing product" (Srinivasan et al. 2005). In this paper, we want to model and forecasts the sales of different products in presence of cannibalization. We observe multiple time series that represent similar assets (or products) that compete with each other. Our objective is to forecast the future values of this time series by taking cannibalization into account to improve the demand forecast. Therefore, we need to introduce external covariates (for instance, the prices of different products) that may explain the cannibalization.

We will apply this model to E-commerce sales data. These data are organized in a hierarchy of categories. For instance, in the family 'HOME', there is a subfamily 'Home Appliance' which contains a category 'Fridge', which can also be further subdivided. It is generally easier to predict aggregated sales for a category than to predict the sales of each product in this category. One of the reasons is the competition between the different products, and the other cross products effects. For instance, the cheapest products and the best-ranked products in the research engine achieve a competitive advantage. However, these advantages do not last forever, with the introduction of new products on the markets. Furthermore, prices and ranking in search engines change every day. Therefore, the competitiveness of each product changes every time step.

In Sect. 2, we present the model used to predict E-commerce sales. In Sect. 3, we establish an oracle bound on the estimation risk of our model. In Sect. 5, we present the application of our model on the various dataset provided by the French E-Commerce retailer *CDiscount.com*.

1.1 Previous work

Managers are generally aware of the presence of cannibalization and competition between assets (Basuroy et al. 2001), but there are few attempts to model and estimate the impact of cannibalization.

In particular, the very well-known Berry et al. (1995) proposes a model of the cross-price elasticity of different products in the U.S. automobile market. They consider the sales of the different models and use household data, such as the composition of the household, its income, its location to model the choice of the consumer. This information is aggregated at the geographic level. This work has been extended by Berry et al. (2004), which considers individual information on each client, instead of aggregated data. This type of models is often used in the automobile industry to forecast sales. It concerns however a large amount of data to be used and is generally used only for long-term prevision.

There has also been some work to identify the presence of cannibalization in a different context, for instance in the beverage industry (Srinivasan et al. 2005), or in presence of innovative products (Van Heerde et al. 2010). The interested reader may also refer to this last paper to have a more detailed overview of cannibalization identification.

The originality of the method proposed in this paper is to use a machine learning approach for modeling competition and to use external covariates to explain cannibalization. We do not use any information on the behavior of the consumer.

Notations We set $\mathbb{N} = \{0, 1, 2, \ldots\}$ and we also also $\|x\| = \sum_{i=1}^{d} |x_i|$, if $x = (x_i)_{i\in[1,d]} \in \mathbb{R}^d$.

2 Model

2.1 Observations

We observe a multi-dimensional time series $X_t = (x_{i,t})_{i\in[1,d]}$ in $\mathbb{N}^d$. For sales prediction, the integer $x_{i,t}$ represents the sales of the product i at the date t. Within this set, products are supposed similar and in competition with one another. For instance, it could be a single type of product or different types of products having the same usage.

We have n observations of this time series. In many cases, n has the same order of magnitude or is smaller than d.

We suppose that we know a non-negative estimator $s(t)$ of the total amount of sales $\sum_{i=1}^{d} x_{i,t}$ at time t given those n observations. Computing such an estimator is generally an easier task than the amount of sales for a specific product at time t, because it is always easier to predict aggregated values than to predict multiple values, and because the global behavior of the series is generally easier to predict than an individual one. Many classical uni-dimensional techniques could be used to compute such an estimator, and we will not discuss this aspect.

Method for prediction of aggregated sales can be found in Box et al. (2015) and Vandeput (2021). It is also possible to cite (Kumar and Patel 2010) for the specific case of aggregated sales prediction.

Let $y_{i,t} = x_{i,t}/s(t)$. It is the *market share* of the product i at the date t.

We also observe a series of covariates $(\theta_{i,t}) \in \mathbb{R}^p$. These covariates are correlated to the values of the series $x_{i,t}$. In the sales forecasting setting, it could for instance represent the price or the profit margin of the product.

The forthcoming section will lead to a rigorous construction of the model.

2.2 Modeling dispersion

The first step is to model the dispersion of the series. It is natural to suppose that $x_{i,t}$ are drawn from a Poisson distribution with parameter $\lambda_{i,t}$.

More precisely, we suppose that there exists $(\epsilon_{i,t})$ independent unit Poisson processes such that $x_{i,t} = \epsilon_{i,t}(\lambda_{i,t})$. We need to introduce such process $(\epsilon_{i,t})$ in order to distinguish the "natural" stochastic dispersion of the random variable and the variation of the parameter $\lambda_{i,t}$. This distinction would be useful later, because, we hope to explain the variation of the parameter $\lambda_{i,t}$.

In the case of E-commerce sales forecasting, the choice of Poisson distributions has already been suggested in Bandara et al. (2019): it has several advantages.

Firstly, we observed that the sales time series are strongly heteroscedastic and that the local variance of the series is strongly correlated with the local mean of the time series. This phenomenon also appears when the dispersion is modeleded by Poisson processes.

Secondly, it allows us to restrict the effects of the presence of outliers in our data. Indeed, higher values are more likely than with a Gaussian white noise modeling for instance.

Thirdly, positive integer values are naturally modeled by a counting process. We can suppose, that for each week t and each product i, the arrival of clients follows a Poisson process and that the parameter of this Process change for each time period t. It implies, that arrival time of the different clients are independent conditionally to the parameter $\lambda_{i,t}$.

We could also have use binomial negative distribution, in order to model over-dispersion. However, this would add another parameter to estimate and likely increase the risk of over-fitting.

2.3 Modeling competition

Now, we want to model the competition and the cannibalization between the different time series. The main idea is to introduce weights for each product and each date. Such a weight represents the competitiveness of each product, which may vary over time. Then, we distribute the sales proportionally to this weight.

More formally, for each series i at each date t, we introduce a weight $w_{i,t}$. Then the parameters $\lambda_{i,t}$ in the previous section are defined as:

$$\lambda_{i,t} = s(t) \cdot \frac{w_{i,t}}{1 + \sum_{j=1}^{d} w_{j,t}}$$

The "+1" is here to ensure that the magnitude of the weight remains the same for all the observed periods. Therefore :

$$\mathbb{E}\left[\sum_{i=0}^{d} x_{i,t} | w_{1,t}, \dots w_{d,t}\right] = s(t) \cdot \frac{\sum_{i=0}^{d} w_{i,t}}{1 + \sum_{i=1}^{d} w_{i,t}}$$

If the sum of the weights is large enough, then $s(t)$ is a good estimator of the sum $\sum_{i=0}^{d} x_{i,t}$.

It is easy to add a new product in this setting just by adding a new weight. It is useful, because of the short sales cycle of numerous products.

2.4 Modeling temporal evolution

In this section, we explain how the weight of the previous section are computed and how they vary time. We suppose that there is a function ϕ, such that

$$w_{i,t} = \phi(y_{i,t-1}, \theta_{i,t}) \tag{1}$$

. Let us explain the assumption behind this relation. To begin, we should note that the function ϕ is applied to two different parameters. The first one relies on the past values of the market share. This is an important value because many intrinsic aspects of the product are coded within the past values. For instance, its quality, its notoriety, its position on the market are reflected in the past sales and they do not change rapidly. In practice, we would consider more than 1 value in the past, which mean that we would have: $w_{i,t} = \phi(y_{i,t-k}, \dots, y_{i,t-1}, \theta_{i,t})$.

The second parameter is the vector of covariates $(\theta_{i,t})$. These covariates should explain the variation of competitiveness.

Finally, let's remark that we consider that the underlying behavior of each series is the same. Indeed, we use a unique function ϕ for all the series we observe instead of using a specific ϕ_i for every series. It means that the different series are interchangeable and have the same behavior. It allows them to share information between series and to adapt to newly introduce the product on the market.

However, this last assumption has some drawbacks. In particular, when some unknown or unrecorded features are relevant for the prediction of the sales, this could lead to changes in the market share that are not fully explained by this model.

2.5 Summary

The model may thus be written as:

$$\begin{cases} X_1 \sim \mathcal{P}_{X_1} & \\ x_{i,t} = \epsilon_{i,t}(\lambda_{i,t}) & \text{for } t > 1 \\ \lambda_{i,t} = s(t)\dfrac{\phi(x_{i,t-1}/s(t-1), \theta_{i,t})}{1 + \sum_{j\in[1,d]} \phi(x_{j,t-1}/s(t-1), \theta_{j,t})} & \end{cases}$$

where $\mathcal{P}_{X_1}$ denotes the distribution of the first values. With this model (X_t) is a (non-homogeneous) Markov chain with a transition function F_t such that

$$X_t = F_t(X_{t-1}, \varepsilon_t)$$

where $\varepsilon_t = (\epsilon_{1,t}, \dots \epsilon_{i,t})$. More precisely

$$F_t(X, \varepsilon_t) = \varepsilon_t \left(s(t) \frac{\Phi(\frac{X}{s(t-1)}, \theta_t)}{1 + \|\Phi(\frac{X}{s(t-1)}, \theta_t)\|} \right) \tag{2}$$

where Φ is the extension of ϕ to $\mathbb{R}_+^d$.

2.5.1 Comparison with other models

This model extends both univariate count model with exogeneous covariates (Agosto et al. 2016; Rasmus and Anders 2019; Doukhan et al. 2020) and non-linear univariate count models (Fokianos 2012). It is also an extension of multivariate count auto-regressive model (Fokianos et al. 2020), where we add non-linear relation between times series.

We also paralleled this model with the model proposed for auto-regressive categorical times series introduced by Fokianos and Truquet (2017). They extend this model using covariates, prooving the ergodicity and stationarity of Markovian model with covariates under mixing conditions. However, the main difference is the use of a logistic function instead of a simple ratio between weights. Logistic regression often appears in classification, because it tends to assign all the weight to a single class. Here we want to model an equilibrium between different classes and therefore we do not use a logistic function.

3 Estimation risk bounds on empirical risk estimator

In this section, we establish theoretical bounds on the estimation risk of our model. Contrary to Doukhan et al. (2012), we cannot use weak dependence hypotheses. As in Agosto et al. (2016) and Debaly and Truquet (2020), we will use introduce a contraction condition to control the dispersion of the model. The specific contraction and exponential inequalities we will use were introduced by Dedecker and Fan in (2015) and extended for the non-stationary times series in Alquier et al. (2019).

3.1 Contraction condition

In order to apply the result of Alquier et al. (2019), a contraction condition on the Markov transition function must be verified. More precisely, there must be a constant $\rho \in [0, 1[$ such that for all $X, X' \in \mathbb{R}^d$:

$$\sup_t \mathbb{E}\Big[\|F_t(X, \varepsilon_t) - F_t(X', \varepsilon_t)\|\Big] \leq \rho \|X - X'\| \tag{3}$$

This condition holds under additional assumptions over the function ϕ

Lemma 1 *Assume that we have a weight function ϕ and a seasonality s defining a transition function F_t as (2). If ϕ and s are such that:*

1. *There is a constant τ_s such that, for all t:*

$$\frac{s(t+1)}{s(t)} \leq \tau_s,$$

2. *There is a constant $\tau \in \mathbb{R}_+$ such that for all $x \in \mathbb{R}_+, \theta \in \mathbb{R}^p$*

$$\frac{\delta \phi}{\delta x}(x, \theta) \leq \tau,$$

3. *For all $X \in \mathbb{R}_+^d$ and $\theta \in \mathbb{R}^p$, $\|\Phi(X, \theta)\| \geq 1$*
4. *$\tau_s \tau < 1$.*

Then the random iterated system F_t fits the contraction condition (3) for any $\rho < 1$ such that $\tau_s \tau \leq \rho$.

Proof Note $G_t(X) = \frac{\Phi(\frac{X}{s(t-1)}, \theta_{i,t})}{1+\|\Phi(\frac{X}{s(t-1)}, \theta_i)\|}$

For $X, X' \in {\mathbb{R}_+}^d$

$$\begin{aligned}
\mathbb{E}\Big[\|F_t(X, \varepsilon_t) - F_t(X', \varepsilon_t)\|\Big] &\leq s(t)\|G_t(X) - G_t(X')\| \\
&= s(t) \frac{\|(1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)\Phi(\frac{X}{s(t-1)}, \theta) - (1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|)\Phi(\frac{X'}{s(t-1)}, \theta)\|}{(1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|) \cdot (1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)} \\
&\leq s(t) \frac{\|(1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)\Phi(\frac{X}{s(t-1)}, \theta) - (1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)\Phi(\frac{X'}{s(t-1)}, \theta)\|}{(1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|) \cdot (1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)} \\
&+ s(t) \frac{\|(1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)\Phi(\frac{X'}{s(t-1)}, \theta) - (1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|)\Phi(\frac{X'}{s(t-1)}, \theta)\|}{(1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|) \cdot (1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)} \\
&\leq s(t) \frac{\|\Phi(\frac{X}{s(t-1)}, \theta) - \Phi(\frac{X'}{s(t-1)}, \theta)\|}{(1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|)} \\
&+ s(t) \frac{\|\Phi(\frac{X'}{s(t-1)}, \theta)\|(\|\Phi(\frac{X'}{s(t-1)}, \theta)\| - \|\Phi(\frac{X}{s(t-1)}, \theta)\|)}{(1 + \|\Phi(\frac{X}{s(t-1)}, \theta)\|) \cdot (1 + \|\Phi(\frac{X'}{s(t-1)}, \theta)\|)}
\end{aligned}$$

$$\leq 2s(t)\frac{\|\Phi(\frac{X}{s(t-1)},\theta)-\Phi(\frac{X'}{s(t-1)},\theta)\|}{(1+\|\Phi(\frac{X}{s(t-1)},\theta)\|)}$$

Using the condition 2, $x \mapsto \phi(x,\theta)$ is τ-Lipschitz for each θ.

$$\mathbb{E}\Big[\|F_t(X,\varepsilon_t)-F_t(X',\varepsilon_t)\|\Big] \leq \frac{2}{(1+\|\Phi(\frac{X}{s(t-1)},\theta)\|)}\frac{s(t)}{s(t-1)}\tau\|X-X'\|$$

Using the above conditions 1, 3 and 4, we have:

$$\mathbb{E}\Big[\|F_t(X,\varepsilon_t)-F_t(X',\varepsilon_t)\|\Big] \leq \tau\tau_s\|X-X'\| \leq \rho\|X-X'\|$$

which concludes the proof. □

We now discuss the conditions in Lemma 1.

The first set a restriction on the regularity of the seasonality, which should not change too abruptly. In practice, this is not always verified. Indeed, some event like Black Friday creates drastic changes in product sales seasonality. Ideally, such event should be handled with other techniques. Otherwise, changes in seasonality are mostly smooth.

The second and the fourth condition limit the variations of the weight function. This is a constraint on the type of models that we use to build this weight function, since it should be smooth. In particular, tree-based models such do not fit this condition. However, for neural network this is similar to the stability condition described in Miller and Moritz (2019), where the author show that enforcing this conditions does not degrade the performance of recurrent neural network.

The third condition is a direct consequence of the '+ 1' trick we use for stabilizing the weight. It is necessary to ensure that this '+ 1' does not suppress the rest of the weights. In theory, this condition is very strong, because it implies a strong constraint on the whole domain of the function Φ to $\mathbb{R}_+^d$. However, in practice, the domain of application of this function is more restrained, and we just need to have this condition on this restrained set.

To compute a generalisation bound on our model we need to introduce:

$$G_{X_1}(x) = \int \|x-x'\| dP_{X_1}(dx')$$
$$H_{t,\varepsilon_t}(x,y) = \int \|F_t(x,y)-F_t(x,y')\| dP_{\varepsilon_t}(dy')$$

To have a Bernstein inequality, we need some constraints on the dispersion of our the times series. More precisely, we need to have the following inequalities for some constants $M > 0$, $V_1 > 0$ and $V_2 > 0$ such that, for all integer $k \geq 2$:

$$\mathbb{E}[G_{X_1}(X_1)^k] \leq \frac{k!}{2}V_1M^{k-2} \tag{4}$$
$$\forall t, \forall x, \mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] \leq \frac{k!}{2}V_2M^{k-2} \tag{5}$$

The condition (4) is satisfied if X_1 admits subgaussian moments. However, this condition is less restrictive than sub-Gaussianity. Now let us consider the second condition (5).

Lemma 2 *If there exists $R > 0$ such that, for each t, $s(t) \leq R$, and if we denote by $M = d\max\{1, eR\}$ and $V_2 = 4M^2 = 4d^2\max\{1, eR\}^2$, then :*

$$\mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] \leq \frac{k!}{2} V_2 M^{k-2}$$

Proof Let us consider

$$\mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] = \int \left(\int \|F_t(x,y) - F_t(x,y')\| dP_{\varepsilon_t}(dy') \right)^k dP_{\varepsilon_t}(dy)$$

From Jensen inequality we derive :

$$\begin{aligned}\mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] &\leq \int\int \|y(s(t)G_t(x)) - y'(s(t)G_t(x))\|^k dP_{\varepsilon_t}(dy') dP_{\varepsilon_t}(dy) \\ &\leq \mathbb{E}[\|Y - Y'\|^k]\end{aligned}$$

where $Y = (Y_i)$ and $Y' = (Y_i')$ are independent vector of independent random variables following Poisson distributions with parameters $s(t)G_t(x)$. Hence :

$$\mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] \leq \mathbb{E}[\|Y\|^k + \|Y'\|^k] \leq 2\mathbb{E}[\|Y\|^k].$$

As $\|G_t(X)\|_\infty \leq 1$, we have

$$\mathbb{E}[\|Y\|^k] \leq d^k \mathbb{E}[\|Y\|_\infty^k] \leq d^k \mathbb{E}[(y_t)^k]$$

where y_t is a random variable following a Poisson process with parameter $s(t)$.

Using Lemma 3, we obtain $\mathbb{E}[(y_t)^k] \leq k!\max\{es(t), 1\}^k$. This ensures

$$\mathbb{E}[H_{t,\varepsilon_t}(x,\varepsilon_t)^k] \leq 2d^k k!\max\{es(t), 1\}^k,$$

and this allows us to conclude. □

Thanks to this lemma, we just need the seasonality $s(t)$ to be bounded to have the condition (5). Nevertheless, we must note that we use loose bounds. In particular, V_2 may be great when the dimension of the serie d is important. Getting a better value for V_2 would be necessary to get a tighter risk bound.

3.2 Risk bounds on empirical risk estimator

In this section, a bound on model selection error is provided.

Let (X_t) be an $\mathbb{R}_+^d$ valued process with n observations following the model described in part 1 for a function ϕ^*. Let S be a set of functions respecting the condition of the Lemma 1 such that $\phi^* \in S$. For a function $\phi \in S$, we define an empirical risk :

$$R_n(\phi) = \frac{1}{n}\sum_{t=2}^{n} \|X_{t+1} - s(t+1)\frac{\Phi(\frac{X_t}{s(t)}, \theta_t)}{1 + \|\Phi(\frac{X_t}{s(t)}, \theta_t)\|}\|.$$

We also define:

$$R(\phi) = \mathbb{E}[R_n(\phi)]$$

We define the minimum empirical risk estimator:

$$\widehat{\phi} = \underset{\phi \in S}{\operatorname{argmin}}\ R_n(\phi)$$

It is possible to bound the estimation risk :

Theorem 1 *Let $K_t(\rho) = \frac{1-\rho^t}{1-\rho}$. If ϕ and $X = (X_i)$ verified the condition (1) to (4), then for $\delta > 0$ we have with probability $1 - \delta$:*

$$R(\widehat{\phi}) \leq R(\phi^*) + (1+\tau)\left(\frac{\sqrt{2V_2 \log(\frac{1}{\delta})}}{\sqrt{n}} + \frac{\sqrt{2V_1 \log(\frac{1}{\delta})}}{n} + \frac{2MK_{n-1}(\rho)\log(\frac{1}{\delta})}{n}\right)$$

Remark 1 We observe the usual decay in $\mathcal{O}(\sqrt{\frac{\log(\frac{1}{\delta})}{n}})$. If we use the values for V_1 established in the Lemma 2, we observe that the error grows linearly with the dimension d.

Proof First, let's recall the usual argument to bound the excess risk :

$$\begin{aligned} R(\widehat{\phi}) - R(\phi^*) &= R(\widehat{\phi}) - R_n(\widehat{\phi}) + R_n(\widehat{\phi}) - R_n(\phi^*) + R_n(\phi^*) - R(\phi^*) \\ &\leq |R_n(\phi^*) - R(\phi^*)| + |R(\widehat{\phi}) - R_n(\widehat{\phi})| \qquad \text{(from the definition of } \widehat{\phi}) \end{aligned}$$

Therefore for all $t > 0$, it holds :

$$\begin{aligned} \mathbb{P}[R(\widehat{\phi}) - R(\phi^*) \geq t] &\leq \mathbb{P}[|R_n(\phi^*) - R(\phi^*)| + |R(\widehat{\phi}) - R_n(\widehat{\phi})| \geq t] \\ &\leq \mathbb{P}[|R_n(\phi^*) - R(\phi^*)| \geq \frac{t}{2}] + \mathbb{P}[|R(\widehat{\phi}) - R_n(\widehat{\phi})| \geq \frac{t}{2}] \end{aligned}$$

Thus:

$$\mathbb{P}[R(\widehat{\phi}) - R(\phi^*) \geq t] \leq 2 \sup_{\phi \in S} \mathbb{P}[|R(\phi) - R_n(\phi)| \geq \frac{t}{2}] \tag{6}$$

Then, we aim at bounding the difference $R_n(\phi) - R(\phi)$ for all possible functions ϕ.

Let $(\mathcal{F}_k)$ be the natural filtration of the chain (X_k)

R_n is $\frac{(1+\tau)}{n}$ Lipschitz separable. Therefore for $\epsilon > 0$, we can apply the theorem 3.1 of Alquier et al. (2019) to $\frac{R_n}{(1+\tau)}$. Actually, we use a slightly different version, as the space of Poisson processes are not actually separable. However, being able to bound $\mathbb{E}[H_{t,\varepsilon_t}(x, y)^k]$ suffice to use their version of Bernstein inequality.

$$\mathbb{P}[|R_n(\phi) - R(\phi)| \geq \frac{(1+\tau)}{n}\epsilon] \leq \exp\left(\frac{-\epsilon^2}{2V_1 + 2(n-1)V_2 + \epsilon M K_{n-1}(\rho)}\right)$$

Hence, we have with probability at least $1 - \delta$:

$$|R_n(\phi) - R(\phi)| \leq \frac{(1+\tau)\sqrt{(2V_1 + 2(n-1)V_2)\log(\frac{1}{\delta})}}{2n} + \frac{(1+\tau)MK_{n-1}(\rho)\log(\frac{1}{\delta})}{n}$$

Therefore, using (6), with probability at $1-\delta$, we have:

$$R(\widehat{\phi}) \leq R(\phi^*) + (1+\tau)\left(\frac{\sqrt{2V_2\log(\frac{2}{\delta})}}{\sqrt{n}} + \frac{\sqrt{2V_1\log(\frac{2}{\delta})}}{n} + \frac{2MK_{n-1}(\rho)\log(\frac{2}{\delta})}{n}\right)$$

□

4 Implementation of the model

Now we present the implementation of the theoretical model proposed in Sect. 2 and we show how it could be used for times series prediction. The code is available on Github (https://github.com/garnier94/Concurrent_Neural_Network).

4.1 Empirical risk minimization

We want to adapt our model to a classical machine learning setting, using empirical risk minimization, in order to be able to use an efficient optimization algorithm. To do so, we introduce a set Ψ of possible weight function ϕ. We will search for the optimal function in this set.

We will perform a prediction at an horizon $h \geq 1$. For a chosen weight function ϕ a covariate vector $\theta_{i,t}$ and known past value of the market share $y_{i,t-h}$, the next value will be predicted by :

$$\hat{y}_{\phi,i,t} = \frac{\phi(y_{i,t-h}, \theta_{i,t})}{1 + \sum_{j\in[1,d]} \phi(y_{j,t-h}, \theta_{j,t})}$$

We will then perform the empirical risk minimization for a loss L:

$$\hat{\phi} = \underset{\phi\in\Psi}{\operatorname{argmin}} \sum_{t=1}^{n-h}\sum_{i=1}^{d} L(y_{i,t}, \hat{y}_{\phi,i,t}) \tag{7}$$

Note that, when L is the Poisson loss function $L(y,\hat{y}) = \hat{y} - x\log\hat{y}$, the empirical risk minimizer is also the function ϕ which minimize the log likelihood of the model presented in Sect. 2. However, we will also use the more standard L_1-loss function to show the interest of the Poisson distribution.

4.2 Concurrent neural network

The most complex choice is that of the set Ψ among which we choose the weight function ϕ. It should be able to satisfy several properties.

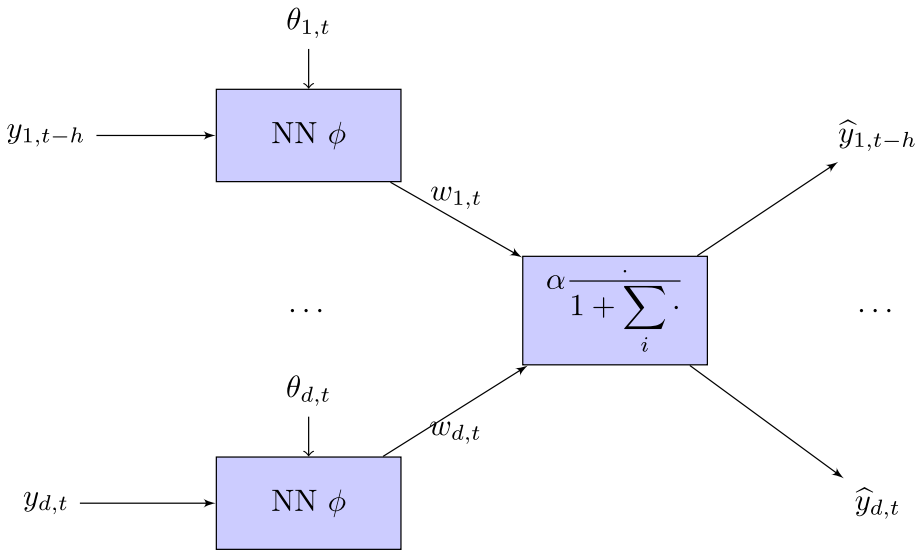

Fig. 1 Concurrent NN model

Firstly, it should be complex enough in order to handle non linear behavior. Indeed, we want to model complex behavior, that depends interacting and sometimes correlated parameters.

Secondly, the considered functions must be differentiable. This is a condition necessary to use powerful optimization algorithm. This condition disqualifies most of tree-based models, often used to predict sales in different context (Vandeput 2021).

This constraint leads us to use feed-forward neural network as sub-models ϕ. Around them we build a structure that we called a concurrent neural network model that we will note Conc-NN to distinguish it from Convolution Neural Network traditionally abbreviated CNN. We summarize this approach on Fig. 1.

We introduce a scale factor $\alpha < 1$. This is because the sums $\sum_{i=1}^{d} y_{i,t-h}$ may be smaller than 1 in some case from of the presence of newly introduced product between the date where the prediction is made $t - h$ and the date where the prediction is actualized t.

In order to stay simple and not introduce any bias in the comparison between models, we do not choose a data-driven values for α. For short and medium-term horizon, $\alpha = 1$, but for long term horizon, we consider $\alpha = 0.85$.

4.3 Neural network architecture and training

Four neural network estimators are presented in the results.

- *FF-NN* Classical Feed-forward neural network trained with L1 Loss
- *L1-Conc-NN* Concurrent neural network presented in the last subsection trained with L1-Loss
- *P-Conc-NN* Same model trained with Poisson Loss
- *L1-Pre-Conc-NN* Concurrent neural network trained with L1-Loss using the pre-trained weight of FF-NN

The training methods are the same for all those models. We use simple feed-forward architecture with less than four layers, and less than 32 neurons per layers. We use a RELU

Table 1 Datasets descriptive statistics

Family	d	Total weekly sales average	Weekly sales average by product	Max sales	SD	NSD	DS-NSD
Baby chair	127	483.0	9.8	238	16.3	1.66	1.50
Freezer	139	1364.0	23.9	567	39.3	1.64	1.45
Keyboard	68	267.8	9.3	374	19.2	2.06	1.55
Lawn Mower	81	369.3	13.5	455	27.4	2.02	1.30
Scooter	589	1927.1	8.8	785	20.3	2.30	1.93
SD Cars	45	288.6	16.4	542	36.1	2.2	1.64
Smartphone	1055	8352.3	29.8	2886	81.6	2.74	2.37
TVs	535	8004.9	64.8	5547	148.2	2.29	1.95

activation function for every layer except the last one where we use a SoftPlus activation. For every category of products, we use a validation period to perform model selection among 10 different neural network architectures. Data are introduced by batch in a random order. Every batch correspond to a week, to allows an easy rescaling in the case of Concurrent Neural Network.

5 Application to e-commerce sales dataset

In this section, we will try to apply our method to a real dataset of E-commerce sales. First, we present the data used in our application.

5.1 Datasets

We consider different data sets coming from the E-commerce company *Cdiscount*. It is a French E-commerce retailer selling a large variety of products.

We use the data available for different families of products sold by CDiscount. These categories have been selected to represent various types of products. The hyper-parameters of the models were chosen using other families. The product categories are presented in Table 1, along with the number of products d and some descriptive statistics. SD stands for Standard Deviation, where NSD (Normalized Standard Deviation) is the ratio between SD and the Weekly sales average by product. We also comupute DS-NSD (Deseasonalized Normalized Standard Deviation) the same ratio for deseasonalized weekly sales. Average and standard deviation are computed only when the products are actually proposed on the website.

These datasets can be roughly separated into three categories. The first one is the product that presents regular seasonality and where the demand is relatively insensitive to price variation. It is the case for Baby chairs and Freezers, which have a small NSD and DS-NSD, and where NSD and DS-NSD are relative similar. The second one is products that present strong seasonality factors, such as Lawn Mowers and Scooters, but are not highly sensitive to price changes. In this case, we observe an important difference between NSD and DS-NSD due to the seasonality factor. The last ones are products such as Smartphones and TVs, which present short sales cycles and/or are very sensitive to price changes, which translates into important NSD and DS-NSD.

We consider the weekly sales starting from January 2017 to December 2020. The first three years are used to train the models, which are evaluated on the last year of data. Added external features include the margin practiced on the product and their prices.

5.2 Evaluation

We want to predict the weekly sales shares of different products for an horizon of h weeks. We will evaluate the prediction using the usual Mean Absolute Error(%MAE) . For a prediction $\widehat{y}_{i,t+h}$, it is defined as:

$$\%MAE = 100 * \frac{\sum_{i=0}^{d} \sum_{t=0}^{T-h} |\widehat{y}_{i,t} - y_{i,t}|}{\sum_{i=0}^{d} \sum_{t=0}^{T-h} y_{i,t}}$$

The %MAE has two main advantages. First, it can deals with outliers, which tends to be over-weighted with other metrics such as RMSE and NMRSE. First, it scales with the level of sales, and so can be used to compare the prediction for the different product categories. However, %MAE error tends to favor under-estimated predictions.

We will use other predictors to get a benchmark of prediction.

- *Last value (LV)* Use the last known value to predict future market share.
- *Moving average (MA)* Use a moving average model to predict the future values. Hyper-parameters are calibrated on a validation period.
- *Random forest (RF)* Random forest using the same features as ConcNN. Different architectures are cross-validated on a validation period.
- *Scaled random forest (S-RF)* We also use Random Forest where the prediction are scaled to match the total number of sales. This is useful to compare our models with the results when we perform a simplistic re-scaling after a model is trained.

5.3 Results

We present the results in the Table 2 for the horizon $h = 4$ (1 month ahead),in the Table 3 for horizon $h = 8$ (2 months ahead), and in the Table 4 for the horizon $h = 12$ (3 months ahead). In some cases, FF-NN were not able to produce any meaningful prevision and only predict 0. In this case, we put a star (*) in the columns. The best model for every set of products and every horizon is in *bold*.

L1-Pre-Conc-NN also failed to produce any prevision other than 0. It is of course the case when FF-NN provides zero predictions, but it can also happen when pretrained weights are too small. We also denote this case by a (*).

General Remarks As expected, the error increases with the horizon of prediction h. It is expected, as long-term previsions are generally more complex than short-term previsions. However, let us remark that this increasing complexity is not the same for every category. When sales cycles are short, for instance for TVs, the influx of new products makes long-term prediction even harder.

Comparison with LV, MA The different Conc-NN models outperform both classical times series estimators LV and MA for almost every horizon and products sets. In particular, traditional time series estimators. It means that Conc-NN can exploit external features.

Comparison with RF The different Conc-NN models outperform random forests (RF) for almost every product for short-term prediction, but RF becomes better for longer-term

Table 2 %MAE Results on the market share prediction for an short-term horizon $h = 4$ weeks

Category	LV	MA	RF	S-RF	FF-NN	L1-Conc-NN	P-Conc-NN	L1-Pre-Conc-NN
Baby chair	76.7	73.8	70.7	70.5	67.7	**59.9**	63.5	*
Freezer	88.1	85.3	69.7	70.1	67.1	66.3	**65.2**	68.3
Keyboard	87.6	81.7	76.7	77.8	**67.1**	72.7	70.3	*
Lawn Mower	83.2	81.5	72.8	75.7	74.6	75.6	**72.5**	74.1
Scooter	86.3	84.1	78.5	82.0	75.2	**73.6**	75.7	74.0
SD Cars	83.7	79.3	88.9	90.3	**74.0**	75.8	77.9	*
Smartphone	84.0	81.6	79.1	84.3	81.3	**75.6**	75.8	75.7
TVs	78.5	80.7	76.2	78.2	*	**71.7**	77.8	*

Table 3 %MAE Results on the market share prediction for an medium-term horizon $h = 8$ weeks

Category	LV	MA	RF	S-RF	FF-NN	L1-Conc-NN	P-Conc-NN	L1-Pre-Conc-NN
Baby chair	85.03	82.4	79.8	80.4	81.2	78.3	**75.6**	*
Freezer	97.5	94.6	76.7	80.2	**70.6**	71.4	71.0	72.3
Keyboard	82.9	79.5	76.5	76.6	**68.0**	70.1	75.3	70.6
Lawn Mower	96.0	92.3	**76.8**	83.0	*	83.0	82.1	*
Scooter	99.5	97.1	79.6	86.7	**76.5**	77.8	76.9	77.9
SD Cars	89.1	86.7	89.5	89.9	79.0	**78.3**	79.0	*
Smartphone	93.6	90.1	88.2	95.6	88.3	85.4	85.3	**84.5**
TVs	103.5	104.0	89.9	97.4	*	**81.4**	90.8	*

Table 4 %MAE Results on the market share prediction for an long-term horizon $h = 12$ weeks

Category	LV	MA	RF	S-RF	FF-NN	L1-Conc-NN	P-Conc-NN	L1-Pre-Conc-NN
Baby chair	88.9	86.0	78.2	77.8	77.1	**76.1**	80.0	85.9
Freezer	97.7	94.7	73.1	76.0	72.3	70.9	**70.0**	78.1
Keyboard	91.5	87.0	79.5	79.1	**69.4**	69.9	74.8	*
Lawn Mower	99.3	96.9	**80.2**	86.4	82.7	84.5	81.5	84.5
Scooter	99.5	97.1	**79.8**	87.1	80.5	85.4	84.0	83.8
SD Cars	95.3	91.6	86.4	89.4	82.2	**77.2**	79.4	*
Smartphone	97.9	95.0	91.5	100.9	89.7	85.6	**85.3**	90.5
TVs	117.4	116.8	**97.5**	115.1	*	101.8	110.6	*

horizon. One way to explain this fact is that RF tends to under-predict sales, which favors it for %MAE evaluation. Example of such under-prediction are presented in Fig. 2

Note that S-RF prediction strongly under-performs RF. Therefore, simply rescaling prediction is not enough to correctly distribute market share.

Comparison with FF-NN

For smartphones and TVs, Conc-NN outperforms FF-NN. These categories are considered as the most competitive, with a lot of price changes and short sales cycles. It may be the proof that our model correctly describes the competition mechanisms in this category.

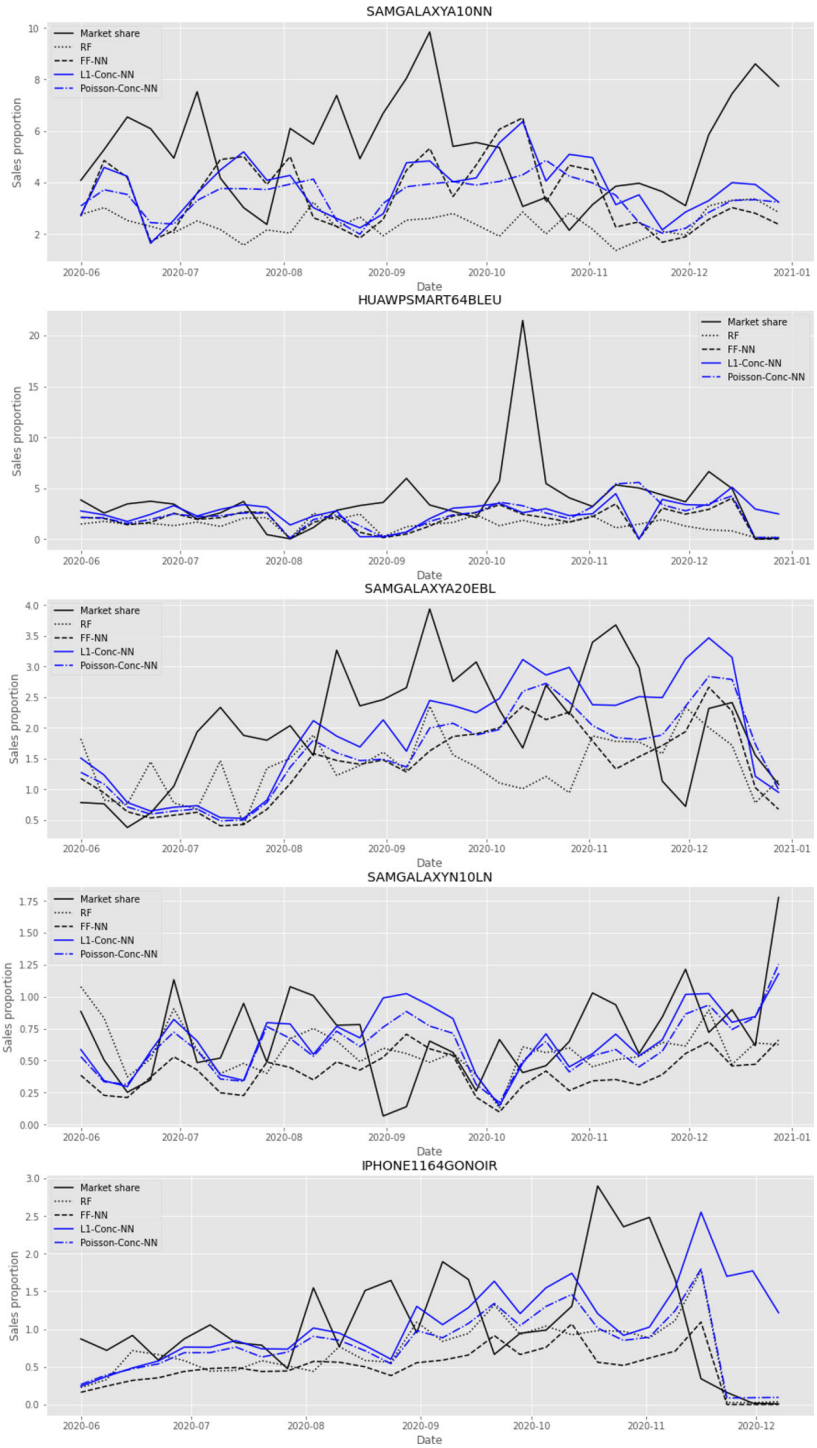

Fig. 2 Market prediction for some popular smartphones for an horizon $h = 4$

There is maybe another explanation, however. Conc-NN also performs well for Scooters, and the three categories (Scooter, TVs, and Smartphones) are also the categories with the most products. They also have good performances on SD Cards, which is also a very competitive category with a few products.

FF-NN outperforms Conc-NN models for keyboards for every horizon.

FF-NN shares a drawback with RF. It under-predicts some products. This may be also due to the L_1-Loss minimization, which tends to favor under-predictive models. We also show some examples in Fig. 2.

Pretrained Model L1-Pre-Conc-NN Pretrained Concurrent model generally under-performs other Conc-NN. Most of the time, they also under-perform the FF-NN used for pretraining. They also tend to predict 0 a lot.

Therefore, pretraining models do not seem to be generally useful. However, they obtain generally good results for Smartphones for all horizons.

Poisson Loss VS L1 Loss Poisson and L1 Loss leads generally to similar performances. It is hard to see any pattern in the relative performance of both models. Let us just notice that L1 Loss outperforms Poisson for every horizon for TVs, whereas Poisson Loss outperforms L1 for Lawn Mowers, but it could be explained by accident.

When we observe the prediction in Fig. 2, we can notice that L1-Loss-based prediction tends to present higher variation than Poisson-Loss-based prediction. This sensitivity to variation may explain the higher performance of L1-Loss for TVs.

5.4 Features importance

Our models take into account two external features, the price and margin practiced on the product. To understand how our model treats the covariate, we consider the partial dependence of our models to see if we could explain the variation of underlying weight function ϕ. To compute the partial dependance graph for a feature θ, we use the following procedures:

- We consider the distribution of a given feature θ in the training set, and we split this distribution into 100 bins.
- For each bin i we compute the average value θ_i of the feature θ on the bins.
- We compute the average weight on the test set of $\phi(\tilde{\theta}_{i,t})$, where $\tilde{\theta}_{i,t}$ is the usual point of data of the test set where the feature θ has been replaced by θ_i
- We then plot all the couple (average bin , average weight) .

All the partial dependence are computed for the smartphones category, for an horizon $h = 4$.

Partial dependence w.r.t. past proportion On Fig. 3, we present the partial dependence of the weight with respect to the last known market share. Logically, it is increasing. When the past sales proportion of the sales were important, it is a strong indication of the competitiveness of a product.

Partial dependence w.r.t. prices On Fig. 4, we present the partial dependence of the weight with respect to the price of the smartphone.

Smartphones reach their peak competitivity around 150 €, then the average weight decrease. Lower prices seem to indicate a lower quality of the smartphones, therefore reducing the attractivity for customers. Logically, higher prices also decrease attractivity.

Partial dependence w.r.t. margin On Fig. 5, we present the partial dependence of the weight with respect to the margin of the smartphone. Margin is a better indicator than prices, because

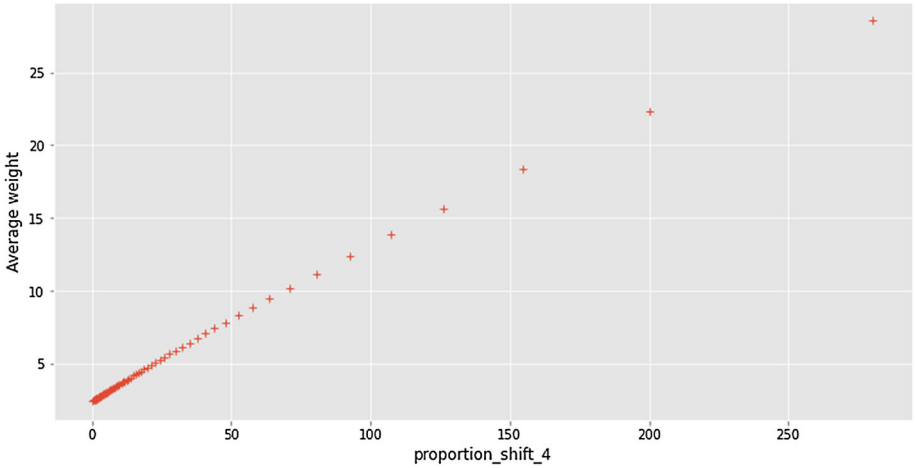

Fig. 3 Partial dependence with respect to the past proportion variables

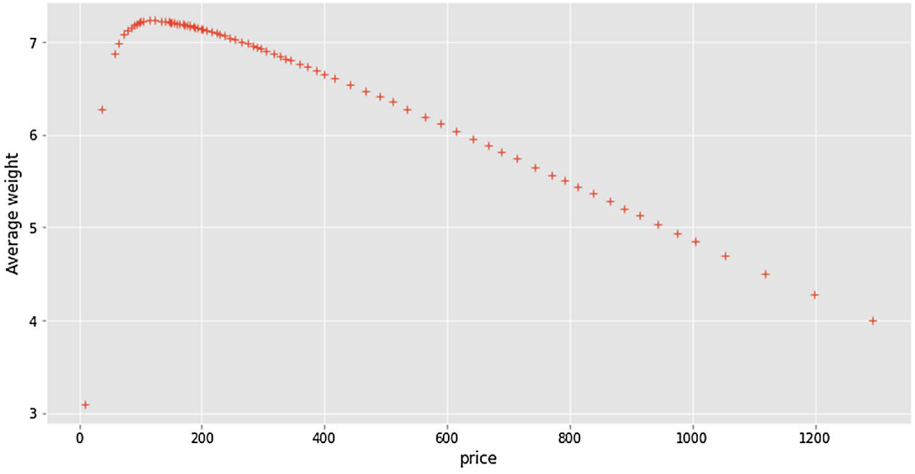

Fig. 4 Partial dependence with respect to the price

it takes into account the quality of the product, but suffers from other factors. Even so, let us note that the overall weight variation is largely smaller than in the last graph . The role of the raw margin seem less important than other factors.

Here, when margin is positive, the weight seems to increase with the margin. It is also counter-intuitive, but can also be explained by pricing behavior of the company. When a product receive a great interest, the company can easily increase its margin. The company may also want to push forward products with higher margin.

When margin are negative however, i.e. when products are on sales, we observe a strong increase on its competitiveness.

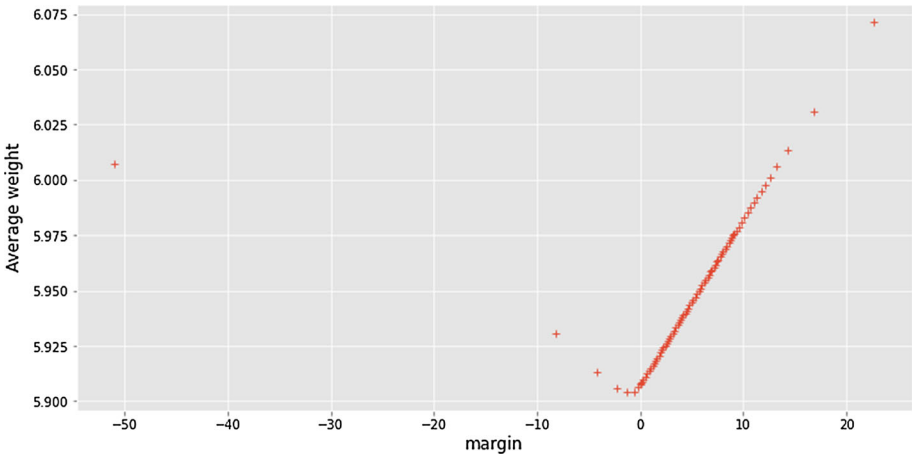

Fig. 5 Partial dependence with respect to the margin percentages

6 Conclusions

A model for concurrent time series is proposed in this article. It is based on the creation of unknown "competitiveness" quantity that depends on the characteristic of a product and previous information on its popularity. Under a relatively common condition, we establish a bound on the risk of our model. This bound follows the usual decay in $\mathcal{O}(\sqrt{\frac{\log(\frac{1}{\delta})}{n}})$ observed in Machine Learning.

We use this model on real-world data, using a Neural Network to compute the competitiveness using past sales values and other external features. This approach outperforms classical ML and times series estimators, especially for short and medium terms predictions. It also improves classical Neural Network estimators, especially when they are numerous products and when the competition between time series is high. It also partially avoid under-prediction that tends to affect other predictors.It is possible to explain the weight function. However, the behavior of this function may be counter-intuitive due to the feedback loop.

Further work may include the extension of this model to non-differentiable sub-model such as Random Forests or Boosting trees. Indeed, practitioners often use a mix of categorical and numerical features to predict sales, and this could only be done by a non-differentiable model. Therefore, extending concurrent model could be of great use for practionners.

Appendix: Moment bound for Poisson distribution

Lemma 3 *Let X be a random variable following a Poisson distribution of parameter λ. Denote $M = \max(1, \lambda e)$, then :*

$$\mathbb{E}\left[|X|^k\right] \leq k! M^k \,.$$

Proof The moment generating function of the X is $g(x) = \exp(\lambda(e^x - 1))$. We denote by $m_k = g^{(k)}(0)$ the k-th moment of the distribution. The first derivative of g satisfies

$g'(x) = \lambda \exp(x) g(x)$. Using Leibniz formula, we have :

$$g^{(k+1)}(x) = \lambda \sum_{i=0}^{k} \binom{n}{i} g^{(i)}(x) \exp(x).$$

For all k, we have the following recurrent relation

$$m_{k+1} \leq \lambda \sum_{i=0}^{k} \binom{n}{i} m_i.$$

We will prove the hypothesis $H_k : m_k \leq k! M^k$ by induction. We have $m_0 = 1$ and $m_1 = \lambda$, so H_0 and H_1 are verified. For $k > 1$, if we suppose (H_i) verified for all $i \leq K$:

$$\begin{aligned} m_{k+1} &\leq \lambda \sum_{i=0}^{k} \binom{k}{i} i! M^i \leq \lambda \sum_{i=0}^{k} \frac{k!}{(k-i)!} M^i \leq \lambda M^k k! \sum_{i=0}^{k} \frac{1}{i!} M^{-i} \\ &\leq \lambda M^k (k+1)! e^{\frac{1}{M}} \end{aligned}$$

As $M \geq 1 : m_{k+1} \leq \mu e M^k (k+1)! \leq M^{k+1}(k+1)!.$ □

Ahle (2021) gives slightly better moment bound for Poisson distribution. However, this article has not been published yet.

References

Agosto, A., Cavaliere, G., Kristensen, D., & Rahbek, A. (2016). Modeling corporate defaults: Poisson autoregressions with exogenous covariates (parx). *Journal of Empirical Finance, 38,* 640–663.

Alquier, P., Doukhan, P., & Fan, X. (2019). Exponential inequalities for nonstationary markov chains. *Dependence Modeling, 7*(1), 150–168.

Ahle Thomas, D. (2021). Sharp and simple bounds for the raw moments of the binomial and poisson distributions. arXiv preprint arXiv:2103.17027,

Alon, I., Min, Q., & Sadowski, R. J. (2001). Forecasting aggregate retail sales: A comparison of artificial neural networks and traditional methods. *Journal of retailing and consumer services, 8*(3), 147–156.

Box, G. E.P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control*. John Wiley & Sons.

Berry, B., James, L., & Ariel, P. (1995). Automobile prices in market equilibrium. *Econometrica: Journal of the Econometric Society*, 841–890.

Berry, S., Levinsohn, J., & Pakes, A. (2004). Differentiated products demand systems from a combination of micro and macro data: The new car market. *Journal of political Economy, 112*(1), 68–105.

Basuroy, S., Mantrala, M. K., & Walters Rockney, G. (2001). The impact of category management on retailer prices and performance: Theory and evidence. *Journal of Marketing, 65*(4), 16–32,

Bandara, K., Peibei, S., Christoph, B., Hansika, H., Quoc, T., & Brian, S. (2019). Sales demand forecast in e-commerce using a long short-term memory neural network methodology. In *Neural information processing* (pp. 462–474). Springer International Publishing.

Dedecker, J., & Fan, X. (2015). Deviation inequalities for separately lipschitz functionals of iterated random functions. *Stochastic Processes and their Applications, 125*(1), 60–90.

Doukhan, P., Fokianos, K., & Tjøstheim, D. (2012). On weak dependence conditions for poisson autoregressions. *Statistics & Probability Letters, 82*(5), 942–948.

Doukhan, P., Neumann, M. H., & Truquet, L. (2020). Stationarity and ergodic properties for some observation-driven models in random environments.

Debaly Max Z., & Truquet, L. (2020). Iterations of dependent random maps and exogeneity in nonlinear dynamics.

Davis Richard, A., Pengfei, Z., & Tian, Z. (2016). Sparse vector autoregressive modeling. *Journal of Computational and Graphical Statistics, 25*(4), 1077–1096.

Fokianos, K. (2012). Count time series models. In *Handbook of statistics* (Vol. 30, pp. 315–347). Elsevier.
Fokianos, K., Støve, B., Tjøstheim, D., & Doukhan, P. (2020). Multivariate count autoregression. *Bernoulli, 26*(1), 471–499.
Fokianos, K., & Truquet, L. (2017). On categorical time series models with covariates. *Stochastic Processes and their Applications, 129,* 09.
Garnier, R., & Arnaud, B. (2019). Une approche multi-séries pour la prévision de la demande sur des données d'E-Commerce. In *Proceedings of APIA 2019*.
Kumar, M., & Patel, N. R.(2010). Using clustering to improve sales forecasts in retail merchandising. *Annals of Operations Research, 174*(1), 33–46.
Lin, W. T. (1989). Modeling and forecasting hospital patient movements: Univariate and multiple time series approaches. *International Journal of Forecasting, 5*(2), 195–208.
Lutkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.
Miller, J., & Moritz, H. (2019). Stable recurrent models. In *Proceedings of ICLR 2019*,
Pedersen, R. S., & Rahbek, A. (2019). Testing garch-x type models. *Econometric Theory, 35*(5), 1012–1047.
Ren, S., Chan, H.-L., & Siqin, T. (2020). Demand forecasting in retail operations for fashionable products: methods, practices, and real case study. *Annals of Operations Research, 291*(1), 761–777.
Srinivasan, S. R., Sreeram, R., & Grasman, S. E. (2005). Identifying the effects of cannibalization on the product portfolio. *Marketing Intelligence & Planning*.
Tay, F. E. H., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega, 29*(4), 309–317.
Vandeput, N. (2021). *Data science for supply chain forecasting*. De Gruyter.
Van Heerde, H. J., Srinivasan, S., & Dekimpe, M. G. (2010). Estimating cannibalization rates for pioneering innovations. *Marketing Science,29*(6), 1024–1039.
Zhang, B., Jhen-Long, W., & Chang, P.-C. (2018). A multiple time series-based recurrent neural network for short-term load forecasting. *Soft Computing, 22*(12), 4099–4112.