Dynamic Bayesian Network for Stock Record at Tesco

Armando Vieira and Olmo Martinez

February 24, 2019

Abstract

In this document we detailed the model based on Dynamic Bayesian Network to accurately estimate Tesco stocks. This approach is very powerful as it takes into account most uncertainties and hidden variables in complex sales and replenishing process. Is uses the sparse, but accurate information, from MST to constrain all the variables in the model. Once properly validated, this model could be used to autocorrect the stock record.

1 Introduction

Stock records are a vital part of the calculation to establish store orders. Nevertheless, stock records are often inaccurate, as shown during MSTs

Inventory management systems based on Re-Order Point Policies, rely on the assumption that all decreases in product inventory levels result from product sales. Unfortunately, it usually happens that small but random quantities of the product get lost, stolen or broken without record as time passes, e.g., as a consequence of shoplifting. This is usual for retailers handling large varieties of inexpensive products, e.g., grocery stores. In turn, over time these discrepancies lead to stock freezing problems, i.e., situations where the system believes the stock is above the re-order point but the actual stock is at zero, and so no replenishments or sales occur. Motivated by these issues, we model the interaction between sales, losses, replenishments and inventory levels as a Dynamic Bayesian Network (DBN), where the inventory levels are unobserved (i.e., hidden) variables we wish to estimate. We present an Expectation-Maximization (EM) algorithm to estimate the parameters of the sale and loss distributions, which relies on solving a one-dimensional dynamic program for the E-step and on solving two separate one-dimensional nonlinear programs for the M-step

2 Model Variables and Their Dynamics

We have been provided with data regarding a set of products $\mathcal{P}$. For each product $p \in \mathcal{P}$ and each day $t \in \{1, \ldots, T\}$ we observe the values of the two following variables:

In addition, for each product $p \in \mathcal{P}$ and each day $t \in \{1, \ldots, T\}$ we seek to estimate the unobserved values of the following two variables:

Furthermore, for each product p and day t the following events occur (in order):

1. The store opens with nominal inventory level $\widetilde{I}^p_{t-1} \in \mathbb{Z}_{\geq 0}$ and physical inventory level $I^p_{t-1} \in \mathbb{Z}_{\geq 0}$. The nominal inventory $\widetilde{I}^p_{t-1}$ is observable but the physical inventory is not.

Table 1: Observed variables

Variable	Description
S_t^p	Sales of product p on day t
R_t^p	Replenishment (delivery) of product p on day t
$\widetilde{I}_t^p$	Nominal inventory level of product p at the end of day t

Table 2: Unobserved variables

Variable	Description
L_t^p	Losses of product p during day t
I_t^p	Physical inventory level of product p at the end of day t

2. $S_t^p \in \mathbb{Z}_{\geq 0}$ units are sold. This quantity is random, observable, and upper-bounded by the physical inventory level at the beginning of the day I_{t-1}^p. *I.e.:*

$$0 \;\leq\; S_t^p \;\leq\; I_{t-1}^p$$

3. $L_t^p \in \mathbb{Z}_{\geq 0}$ units are accidentally destroyed, lost or stolen. This quantity is random, unobservable, and upper-bounded either by maximum allowable loss, denoted M_p, or by the difference between the inventory level I_{t-1}^p and the number of units sold S_t^p. *I.e.:*

$$0 \;\leq\; L_t^p \;\leq\; M_{p,t} \triangleq \min\{\, M_p,\, I_{t-1}^p - S_t^p \,\}$$

4. $R_t^p \in \mathbb{Z}_{\geq 0}$ units are replenished (delivered). This quantity obeys a deterministic decision rule which is a function of time only, and its value is instantiated and fixed on day $t = 0$ before the sale and loss variables are instantiated. This quantity is observable.

   Note: In reality, the variable $R_t^p \in \mathbb{Z}_{\geq 0}$ follows a deterministic policy that is a function of the inventory level at the beginning of the day $\widetilde{I}_{t-1}^p$. However, since we do not know the form of the policy, we assume that the entire sequence of replenishments $(R_t^p)_{t=1}^T$ is instantiated arbitrarily at day $t = 0$ before the sequences of sales and losses, $(S_t^p)_{t=1}^T$ and $(L_t^p)_{t=1}^T$, respectively, are instantiated.

Table 3: Inventory Level Update Rules

Inventory Level	Update Rule
Nominal	$\widetilde{I}_t^p \;=\; \widetilde{I}_{t-1}^p - S_t^p + R_t^p$
Physical	$I_t^p \;=\; I_{t-1}^p - S_t^p - L_t^p + R_t^p$

5. The inventory levels are updated as shown in Table 3.

Regarding the initial inventory levels, we shall choose one of the following two assumptions, depending on the situation:

- If a physical inspection of the product has been carried on $t = 0$ then we shall assume that the nominal and physical inventory levels match. *I.e.:*

$$I_0^p \;=\; \widetilde{I}_0^p$$

  Hence, in this case we shall assume that I_0^p is a deterministic constant of the problem; *i.e.,* it does not follow a probability distribution.

- If no inspection data is available at $t = 0$, then we shall assume that the initial nominal inventory level is an *over-estimate* of the initial physical inventory level. *I.e.:*

$$I_0^p \;\leq\; \widetilde{I}_0^p$$

  In this case we shall assumme that the initial physical inventory level I_0^p is a non-negative integer random variable uniformly distributed on the interval $[I_{p,0}^{\min}, I_{p,0}^{\max}]$, where the values of $I_{p,0}^{\min}$ and $I_{p,0}^{\max}$ are the lower and upper bounds explained next.

For each product p we can calculate time histories of upper and lower bounds on the physical inventory levels as follows:

- For the time history of upper bounds $(I_{p,t}^{\max})_{t=1}^{T}$ we simply use the nominal inventory level history $(\widetilde{I}_t^p)_{t=1}^{T}$. This is the case because $I_0^p \leq \widetilde{I}_0^p$ and for all days $t \in \{1, \ldots, T\}$:

$$I_t^p \;=\; I_{t-1}^p - S_t^p - L_t^p + R_t^p \;\leq\; I_{t-1}^p - S_t^p + R_t^p \;=\; \widetilde{I}_t^p$$

- For the time history of lower bounds $(I_{p,t}^{\min})_{t=1}^{T}$ we first let $I_{p,T}^{\min} \triangleq 0$ and then recursively compute $I_{p,t}^{\min}$ for $t \in \{T-1, \ldots, 0\}$ as follows:

$$I_{p,t-1}^{\min} \;=\; I_{p,t}^{\min} + S_t - R_t$$

3 Probabilistic Assumptions

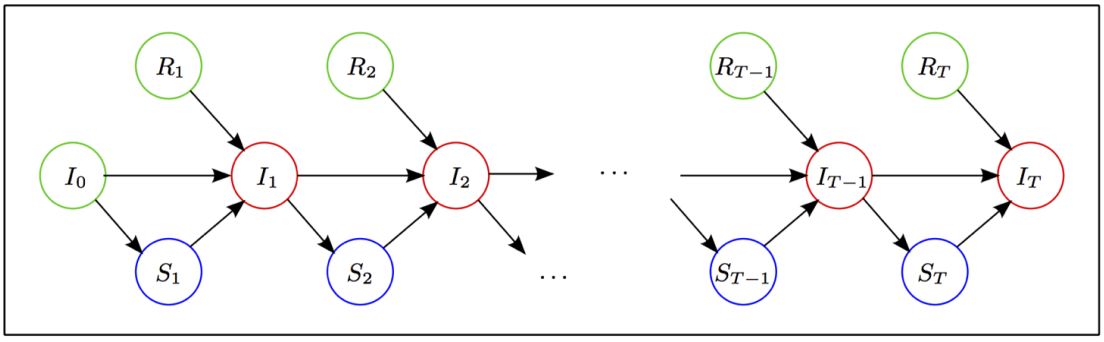

Figure 1: The Dynamic Bayesian Network (DBN) model.

3.1 Conditional Distributions

For each product p and day t we assume the following:

1. Conditional on I^p_{t-1}, the random variable S^p_t follows a truncated Poisson distribution with parameter $\sigma_{p,t}$ and truncation level I^p_{t-1}. *I.e.:*

$$S^p_t \mid I^p_{t-1} \;=\; \min\{\,\text{Poisson}(\sigma_{p,t}),\, I^p_{t-1}\,\} \tag{1}$$

Therefore:

$$\mathbb{P}(\, S^p_t = s \mid I^p_{t-1} = i\,) \;=\; \begin{cases} 1 & \text{if } 0 = s = i \\ \dfrac{\sigma^s_{p,t}\, e^{-\sigma_{p,t}}}{s!}, & \text{if } 0 \le s < i \\ 1 - \displaystyle\sum_{k=0}^{s-1} \frac{\sigma^k_{p,t}\, e^{-\sigma_{p,t}}}{k!}, & \text{if } 0 < s = i \\ 0 & \text{if } s > i \end{cases} \tag{2}$$

2. Conditional on I^p_{t-1} and S^p_t, the random variable L^p_t follows a truncated Poisson distribution with parameter $\lambda_{p,t}$ and truncation level $M_{p,t}(I^p_{t-1}, S^p_t)$, where:

$$M_{p,t}(I^p_{t-1}, S^p_t) \;\triangleq\; \min\{\, M_p,\, I^p_{t-1} - S^p_t\,\} \tag{3}$$

More precisely:

$$L^p_t \mid I^p_{t-1},\, S^p_t \;=\; \min\{\,\text{Poisson}(\lambda_{p,t}),\, M_{p,t}(I^p_{t-1}, S^p_t)\,\} \tag{4}$$

Thus:

$$\mathbb{P}(\, L^p_t = \ell \mid I^p_{t-1} = i,\, S^p_t = s\,) \;=\; \begin{cases} 1 & \text{if } 0 = \ell = M_{p,t}(i,s) \\ \dfrac{\lambda^\ell_{p,t}\, e^{-\lambda_{p,t}}}{\ell!}, & \text{if } 0 \le \ell < M_{p,t}(i,s) \\ 1 - \displaystyle\sum_{k=0}^{\ell-1} \frac{\lambda^k_{p,t}\, e^{-\lambda_{p,t}}}{k!}, & \text{if } 0 < \ell = M_{p,t}(i,s) \\ 0 & \text{if } \ell > M_{p,t}(i,s) \end{cases} \tag{5}$$

Finally, recalling the update rule of the physical inventory levels (Table 3), we recognize that for product p and day t the probability of the physical inventory level I^p_t conditional on the previous inventory level I^p_{t-1}, on the sales S^p_t and on the replenishment R^p_t can be calculated as follows:

$$\begin{aligned} &\mathbb{P}(\, I^p_t = i_t \mid I^p_{t-1} = i_{t-1},\, S^p_t = s,\, R^p_t = r\,) \\ &\;=\; \mathbb{P}(\, L^p_t = i_{t-1} - i - s + r \mid I^p_{t-1} = i_{t-1},\, S^p_t = s,\, R^p_t = r\,) \end{aligned} \tag{6}$$

3.2 Joint Distribution

In light of the conditional distributions previously described, it is possible to realize that the joint probability distribution of all the variables con be factored as the Dynamic Bayesian Network (DBN) shown in Figure 1, where deterministic variable nodes are shown in green, observed random variable nodes are shown in blue, and unobserved random variable nodes are shown in red. Moreover, for each day t :

- The distribution of node S_t^p conditional on node I_{t-1}^p is given in Equation 2.
- The distribution of node I_t^p conditional on nodes I_{t-1}^p, S_t^p and R_t^p can be computed using Equations 6 and 5.

Now, suppose that for some product p we have physical inspection data for the zeroth day, *i.e.,* $I_0^p = \widetilde{I}_0^p$. In addition, suppose we also have a particular instance of all problem parameters and variables; *i.e.,* suppose we also have:

- Fixed sale and loss parameters, denoted $\boldsymbol{\sigma^p} = (\sigma_1^p, \sigma_2^p, \ldots, \sigma_T^p)$ and $\boldsymbol{\lambda^p} = (\lambda_1^p, \lambda_2^p, \ldots, \lambda_T^p)$, respectively.
- An instance of the physical inventory level history $(I_t^p)_{t=0}^T$, denoted $\boldsymbol{i^p} = (i_t)_{t=0}^T$.
- An instance of the sales history $(S_t^p)_{t=1}^T$, denoted $\boldsymbol{s^p} = (s_t)_{t=1}^T$.
- The replenishments history $(R_t^p)_{t=1}^T$, denoted $\boldsymbol{r^p} = (r_t)_{t=1}^T$.

Then the joint likelihood (probability) of the instance described above, denoted here as

$$L(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p}),$$

is given by the following expression:

$$\begin{aligned} & L(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p}) \\ & \triangleq \prod_{t=1}^{T} \mathbb{P}(\, S_t^p = s_t \mid I_{t-1}^p = i_{t-1} \,)\, \mathbb{P}(\, I_t^p = i_t \mid I_{t-1}^p = i_{t-1},\, S_t^p = s_t,\, R_t^p = r_t \,) \end{aligned} \tag{7}$$

This joint likelihood L is the quantity we seek to maximize as a function of the parameters $\boldsymbol{\sigma^p}$ and $\boldsymbol{\lambda^p}$ and of the physical inventory levels $\boldsymbol{i^p} = (i_t)_{t=0}^T$, as will be explained later. However, for numerical reasons it is much more convenient to maximize the log-likelihood, denoted here as $\ell(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p})$. *I.e.:*

$$\ell(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p}) \triangleq \log L(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p})$$

Hence, taking the logarithm of Equation 7 we can see that:

$$\begin{aligned} & \ell(\boldsymbol{\sigma^p}, \boldsymbol{\lambda^p}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p}) \\ & = \sum_{t=1}^{T} \left[\underbrace{\log \mathbb{P}(\, S_t^p = s_t \mid I_{t-1}^p = i_{t-1} \,)}_{\triangleq\, \phi_t(i_{t-1}, s_t)} + \underbrace{\log \mathbb{P}(\, I_t^p = i_t \mid I_{t-1}^p = i_{t-1},\, S_t^p = s_t,\, R_t^p = r_t \,)}_{\triangleq\, \psi_t(i_{t-1}, i_t, s_t, r_t)} \right] \end{aligned} \tag{8}$$

Furthermore, for each day t:

- The term $\phi_t(i_{t-1}, s_t)$ can be calculated as the logarithm of Equation 2.
- The term $\psi_t(i_{t-1}, i_t, s_t, r_t)$ can be calculated by first taking the logarithm of Equation 6 and then using Equation 5.

4 Learnable Parameters

In Luis's paper, he assumed that each product $p \in \mathcal{P}$ has constant sale and loss parameters; *i.e.,* that $\boldsymbol{\sigma^p} = (\sigma^p, \sigma^p, \ldots, \sigma^p)$ and that $\boldsymbol{\lambda^p} = (\lambda^p, \lambda^p, \ldots, \lambda^p)$. Now, since for this work we have been asked to extend the original model, we will suppose there existe a rule for computing a feature vector for each product p and day t, denoted $\boldsymbol{x_{p,t}} \in \mathbb{R}^n$, and then assume a functional relationship between those feature vectors and the product's parameters.

4.1 Computation of Feature Vectors

To decide on a rule for computing feature vectors in a way that prevents extrapolation, as well as possible overfitting issues, we must base the decision on the availability of the data. In our case, we have been given, for each product $p \in \mathcal{P}$, the time series of sales, replenishments and nominal inventory levels from May 10 of 2018. (Today is, say, December 27, 2018.) Hence for each product we have data for up to $T = 231$ days, or equivalently 33 weeks.

We propose the following rule for computing feature vectors $\boldsymbol{x_{p,t}}$ of dimention n, where for any $i \in \{1, \ldots, n\}$ de i^{th} entry of the vector is denoted $\boldsymbol{x_{p,t}}[i]$.

- $\boldsymbol{x_{p,t}}[1:7]$ is a one-hot encoding of the weekday associated with day t.
- $\boldsymbol{x_{p,t}}[8:13]$ is a one-hot encoding of the week of the month associated with with day t.
- $\boldsymbol{x_{p,t}}[14:15] = (\widetilde{I}^p_{t-1}, S^p_{t-1})$, *i.e.,* it is the nominal inventory level at the beginning of day t and the sales on day $t-1$.
- $\boldsymbol{x_{p,t}}[16]$ is the average nominal intentory level of the week previous to day t.
- $\boldsymbol{x_{p,t}}[17]$ is the average sales of the week previous to day t.
- $\boldsymbol{x_{p,t}}[18:n]$ may be any other function of the date or of the nominal inventory levels, sales or replenishments from the first day up to day t.

4.2 Functional Relationships of the Learnable Parameters

Since we require all parameters σ^p_t to be strictly positive, we shall assume that there exists a constant $\sigma^p_0 \in \mathbb{R}$ and a vector $\boldsymbol{w_{\sigma,p}} \in \mathbb{R}^n$ such that:

$$\sigma^p_t \;\approx\; \exp\left(\sigma^p_0 + \boldsymbol{w'_{\sigma,p}}\, \boldsymbol{x_{p,t}}\right)$$

Hence in this case the actual sale parameters that we seek to learn (estimate) are no longer the values $\boldsymbol{\sigma^p} = (\sigma^p_1, \sigma^p_2, \ldots, \sigma^p_T)$, but rather the constant σ^p and the vector $\boldsymbol{w_{\sigma,p}}$.

Similarly, since all parameters λ^p_t must be strictly positive, we shall assume that there exists a constant $\lambda^p_0 \in \mathbb{R}$ and a vector $\boldsymbol{w_{\lambda,p}} \in \mathbb{R}^n$ such that:

$$\lambda^p_t \;\approx\; \exp\left(\lambda^p_0 + \boldsymbol{w'_{\lambda,p}}\, \boldsymbol{x_{p,t}}\right)$$

5 Objective Function

The goal of this model is estimating, for each product $p \in \mathcal{P}$, the model parameters σ^p_0, λ^p_0, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$. In particular, we would like to obtain maximum likelihood estimates (MLEs) for

these parameters; *i.e.,* the values they take when they maximize Equation 7. Unfortunately, the model that we have previously defined is non-linear in its parameters and requires that the history of physical inventory levels $(I_t^p)_{t=1}^T$ be simultaneously estimated. This is an issue because the model parameters are continuous, the physical inventory levels are discrete, and there are there are no reliable non-linear programming algorithms for maximizing functions with mixed variables.

To work around this shortcoming, we select initial values for the model parameters σ_0^p, λ_0^p, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$ and then repeat apply the following EM Algorithm:

1. **E-Step:** We compute a sequence of physical inventory levels $(I_t^p)_{t=1}^T$ of maximum log-likelihood. This is achieved by using Dijkstra's shortest-path algorithm on a specially constructed graph.
2. **M-Step:** Using the sequence of physical inventory levels $(I_t^p)_{t=1}^T$ from the previous step, we improve the log-likelihood of the values of the model parameters σ_0^p, λ_0^p, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$ using a few tens or hundreds of iterations of a gradient ascent method.

Finally:

- Ideally, the EM-Algorithm is executed until the optimal values of both the model parameters and the physical inventory levels are obtained. In reality, we could set a fixed number of iterations and run the algorithm for that number of times.
- The details of the implementation of the E and M steps are shown in the technical document titled "Doc-02: Model Implementation".

6 M-Step

In this step we seek to maximize the likelihood of the model parameters σ_b^p, λ_b^p, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$ using a gradient ascent given:

- An instance of the physical inventory level history $(I_t^p)_{t=0}^T$, denoted $\boldsymbol{i^p} = (i_t)_{t=0}^T$.
- An instance of the sales history $(S_t^p)_{t=1}^T$, denoted $\boldsymbol{s^p} = (s_t)_{t=0}^T$.
- The replenishments history $(R_t^p)_{t=1}^T$, denoted $\boldsymbol{r^p} = (r_t)_{t=1}^T$.

The very first time the M-Step is exectuted we must select initial values of the model parameters. Every other time the M-Step is executed, the parameters are initialized to their last values in the previous call to the M-step.

6.1 Initial Values of the Model Parameters

Recall the functional relationships of the learnable parameters:

$$\begin{aligned} \sigma_{p,t} &\approx \exp\left(\sigma_b^p + \boldsymbol{w'_{\sigma,p}}\, \boldsymbol{x_{p,t}}\right) \\ \lambda_{p,t} &\approx \exp\left(\lambda_b^p + \boldsymbol{w'_{\lambda,p}}\, \boldsymbol{x_{p,t}}\right) \end{aligned}$$

Based on these relationships, we propose the following initialization procedure:

1. Set the bias σ_b^p to be equal to the logarithm of the average sales. *I.e.,*

$$\sigma_b^p \;=\; \log\left(\frac{1}{T}\sum_{t=1}^{T} S_t^p\right)$$

2. Set the bias λ_b^p to be the logarithm of the average loss. For this purpose, we will use the data on manual stock corrections available. Ideally, the average loss should be a small number, so that its logarithm should be a very negative number.

3. Let ε_σ be a small positive real number; say $\varepsilon_\sigma \in [0.005, 0.100]$. Then sample a sequence of n i.i.d. normal random variables with mean zero and variance ε_σ, and set the vector $\boldsymbol{w_{\sigma,p}}$ to be equal to the sampled sequence.

4. Let ε_λ be a small positive real number; say $\varepsilon_\lambda \in [0.005, 0.100]$. Then sample a sequence of n i.i.d. normal random variables with mean zero and variance ε_λ, and set the vector $\boldsymbol{w_{\lambda,p}}$ to be equal to the sampled sequence.

6.2 Forward Pass

Here we explain how to calculate the log-likelihood of the model parameters given the sequence of input feature vectors. *I.e.,* for each product p, we take as input a sequence of feature vectors $\boldsymbol{x_p} \triangleq (\boldsymbol{x_{p,t}})_{t=1}^T$ and compute the log-likelihood.

1. For each time step t we compute $\sigma_{p,t}$ and $\lambda_{p,t}$.

$$\sigma_{p,t} \;=\; \exp\left(\sigma_b^p + \boldsymbol{w'_{\sigma,p}}\,\boldsymbol{x_{p,t}}\right) \tag{9}$$

$$\lambda_{p,t} \;=\; \exp\left(\lambda_b^p + \boldsymbol{w'_{\lambda,p}}\,\boldsymbol{x_{p,t}}\right) \tag{10}$$

2. For each time step t we compute the term $\phi_t(i_{t-1}, s_t)$.

$$\begin{aligned}\phi_t(i_{t-1}, s_t) \;&=\; \log \mathbb{P}(\, S_t^p = s_t \mid I_{t-1}^p = i_{t-1}\,) \\ &= \begin{cases} 0, & \text{if } 0 = s_t = i_{t-1} \\ s_t\, \log(\sigma_{p,t}) - \sigma_{p,t} - \log(s_t!), & \text{if } 0 \le s_t < i_{t-1} \\ \log\left(1 - e^{-\sigma_{p,t}} \displaystyle\sum_{k=0}^{s_t-1} \frac{\sigma_{p,t}^k}{k!}\right), & \text{if } 0 < s_t = i_{t-1} \\ -\infty & \text{if } s_t > i_{t-1} \end{cases}\end{aligned} \tag{11}$$

Note: Since in this step we optimize with respect to parameters and not variables, in the case $0 \le s_t < i_{t-1}$ we may drop the constant term $-\log(s_t!)$.

3. For each time step t we calculate the loss and the maximum allowable loss.

$$\ell_t \;= i_{t-1} - i - s_t + r_t \tag{12}$$

$$M_{p,t}(i_{t-1}, s_t) \;=\; \min\{\, M_p,\; i_{t-1} - s_t \,\} \tag{13}$$

Then, we compute the term $\psi_t(i_{t-1}, i_t, s_t, r_t)$.

$$\psi_t(i_{t-1}, i_t, s_t, r_t) \;=\; \log \mathbb{P}(\, I_t^p = i_t \mid I_{t-1}^p = i_{t-1},\; S_t^p = s_t,\; R_t^p = r_t\,)$$

$$
\begin{aligned}
&= \log \mathbb{P}(\, L_t^p = \ell_t \mid I_{t-1}^p = i_{t-1},\, S_t^p = s,\, R_t^p = r\,) \\
&= \begin{cases}
0, & \text{if } 0 = \ell_t = M_{p,t}(i_{t-1}, s_t) \\
\ell_t \, \log(\lambda_{p,t}) - \lambda_{p,t} - \log(\ell_t!), & \text{if } 0 \leq \ell_t < M_{p,t}(i_{t-1}, s_t) \\
\log\left(1 - e^{-\lambda_{p,t}} \displaystyle\sum_{k=0}^{\ell_t - 1} \frac{\lambda_{p,t}^k}{k!}\right), & \text{if } 0 < \ell_t = M_{p,t}(i_{t-1}, s_t) \\
-\infty & \text{if } \ell_t > M_{p,t}(i_{t-1}, s_t)
\end{cases}
\end{aligned} \tag{14}
$$

Note: As in the previous step, in the case $0 \leq \ell_t < M_{p,t}(i_{t-1}, s_t)$ we may drop the constant term $-\log(\ell_t!)$.

4. We return the summation in time of the two terms calculated.

$$
\ell(\sigma_b^p, \lambda_b^p, \boldsymbol{w_{\sigma,p}}, \boldsymbol{w_{\lambda,p}}, \boldsymbol{i^p}, \boldsymbol{s^p}, \boldsymbol{r^p}) \;=\; \sum_{t=1}^{T} [\, \phi_t(i_{t-1}, s_t) + \psi_t(i_{t-1}, i_t, s_t, r_t)\,] \tag{15}
$$

6.3 Backward Pass

The backward pass is the calculation of the partial derivative of the log-likelihood with respect to each of the model parameters σ_b^p, λ_b^p, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$ using the Chain Rule. In this work we implement the forward pass using `pytorch`, which means we can readily obtain the partial derivative of the objective with respect to any tensor of parameters by calling the `autograd` function.

6.4 Gradient Ascent

During each M-step many of iterations of the gradient ascent algorithm are executed. Here we employ the simplest variant of the gradient ascent method, which is has only one hyper-parameter; namely, the learning rate $\alpha \in \mathbb{R}_{>0}$. In particular, for any parameter

$$
\rho \;\in\; \{\, \sigma_b^p,\, \lambda_b^p,\, \boldsymbol{w_{\sigma,p}},\, \boldsymbol{w_{\lambda,p}} \,\}
$$

a gradient ascent iteration i proceeds as follows:

1. Begin with current parameter value $\rho(i)$ and compute the partial derivative of the log-likelihood with respect to parameter ρ at $\rho = \rho(i)$, denoted here as $\nabla_\rho \, \ell(\rho(i))$.

2. Compute the next parameter value $\rho(i+1)$.

$$
\rho(i+1) \;=\; \rho(i) + \alpha \, \nabla_\rho \, \ell(\rho(i))
$$

Finally, please note that for the method to be stable we must select a small value of the hyper-parameter α. However, if the value is too small, then the method will require large numbers of iterations to learn the parameters. Therefore, we will need to try different values of the learning rate and then select the best one. Here we start with a value of $\alpha = 0.01$.

7 E-Step

In this step we seek to maximize the likelihood of the physical inventory level history $(I_t^p)_{t=0}^T$ using a graph search given:

- Fixed model parameters σ_b^p, λ_b^p, $\boldsymbol{w_{\sigma,p}}$ and $\boldsymbol{w_{\lambda,p}}$.
- The initial physical inventory level history I_0^p.
- An instance of the sales history $(S_t^p)_{t=1}^T$, denoted $\boldsymbol{s^p} = (s_t)_{t=0}^T$.
- The replenishments history $(R_t^p)_{t=1}^T$, denoted $\boldsymbol{r^p} = (r_t)_{t=1}^T$.

For this purpose, we use the time histories of lower and upper bounds on the physical inventory levels, defined in the previous document and denoted $(I_{p,t}^{\min})_{t=1}^T$ and $(I_{p,t}^{\max})_{t=1}^T$, respectively, to construct a directed weighted graph as follows:

1. We introduce starting node $i_0 \triangleq I_0^p$.
2. For each time step $t \in \{1, \ldots, T\}$ and each inventory level
$$I_t^p \in \{\, I_{p,t}^{\min}, \ldots, I_{p,t}^{\max}\}$$
we introduce a node i_t.
3. For each time step $t \in \{1, \ldots, T\}$ and each pair of nodes i_{t-1} and i_t we compute the loss and maximum allowable loss, denoted ℓ_t and $M_{p,t}(i_{t-1}, s_t)$, using Equations 12 and 13, respectively. Then we check the condition:
$$0 \;\leq\; l_t \;\leq\; M_{p,t}(i_{t-1}, s_t)$$
If the condition evaluates to true, we compute the terms $\phi_t(i_{t-1}, s_t)$ and $\psi_t(i_{t-1}, i_t, s_t, r_t)$ using Equations 11 and 14, introduce the directed edge (i_{t-1}, i_t), and set its weight to:
$$-\phi_t(i_{t-1}, s_t) - \psi_t(i_{t-1}, i_t, s_t, r_t)$$
Note: Unlike the case of the M-step, here we cannot drop the terms $-\log(s_t!)$ and $-\log(\ell_t!)$. Nevertheless, in the rare cases where s_t or ℓ_t are large we can make use of Sterling's Approximation to the log-factorial to avoid number overflow issues.

Consequently, in the graph described above a path $(i_t)_{t=0}^T$ has weight equal to the negative log-likelihood of that history of physical inventory levels. Therefore, searching for a path of minimum weight from the initial node at $t = 0$ to any of the nodes at $t = T$ is equivalent to computing the history of physical inventory levels of maximum log-likelihood.

The search for a path of minimum cost in the graph can be carried using Dijkstra's shortest path algorithm. Furthermore, in order to make the E-step as fast as possible, we can generate the nodes and load them into memory only if they are explored during the graph search, meaning that nodes along paths of very low log-likelihood will never be instantiated. The algorithm is as follows:

1. We initialize the ordered set of explored nodes V to the empty set. This set will need to the repeatedly queried for the node with least cost, so in practice this data structure is implemented using both a set and a min-priority queue.
2. We generate initial node i_0, set its cost to zero, and set its parent node to none. Then we insert node i_0 into set V.

3. While the set V is non-empty:

   (a) Let node i_{t-1} be the node in set V with the least cost.

   (b) If node i_{t-1} is associated with the final time step, *i.e.,* if $t-1=T$, then terminate and return the node. Otherwise:

      i. Remove node i_{t-1} from set V and enumerate all its descendant nodes.

      ii. For each node i_t that descends from node i_{t-1} evaluate its tentative cost as the cost of node i_{t-1} plus the weight of edge (i_{t-1}, i_t). Then:

         - If the node i_t was not previously in the set V, then set its cost to the tentative cost, set its parent to i_{t-1}, and insert the node into set V.

         - If the node i_t was already in set V, compare the tentative cost to the actual cost of the node. If the tentative cost of node i_t is lower than its actual cost, replace its actual cost with the tentative cost, and replace its parent with node i_t; otherwise, do nothing.

Finally, notice that once the algorithm returns a node and terminates, the path of least cost (maximum log-likelihood) can be obtained by first backtracking the nodes' parents, starting from the returned node, and then reversing the obtained sequence.

8 Some use cases

First we consider the weekly variations of parameters λ and σ - see Fig. 2 and 3. We can see that average loss goes from 0.03 to .06 items a day while not having a very clear weekly pattern. The sales parameter σ, on the other hand, has a more well pronounced weekly pattern with a peak between Friday and Saturday.

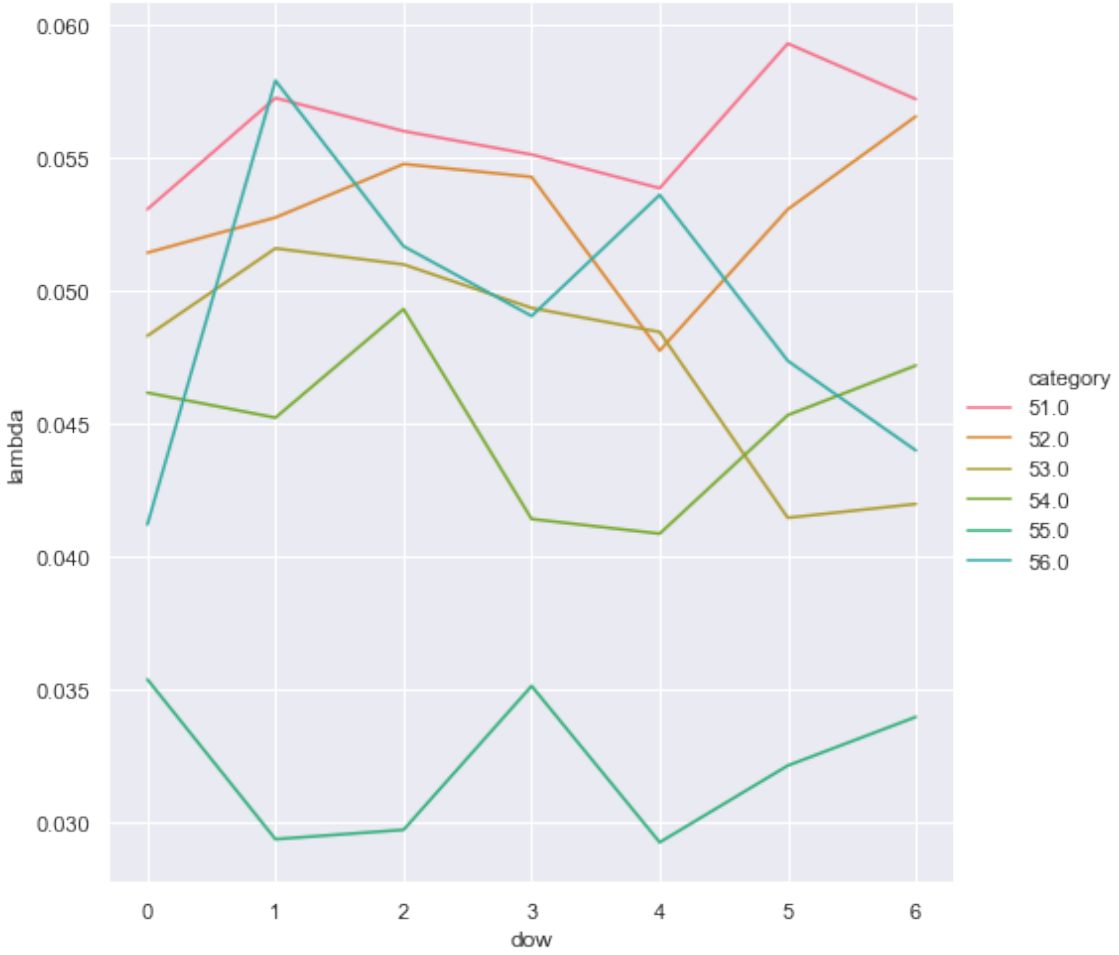

Figure 2: Weekly variation of λ parameter for several product categories considered.

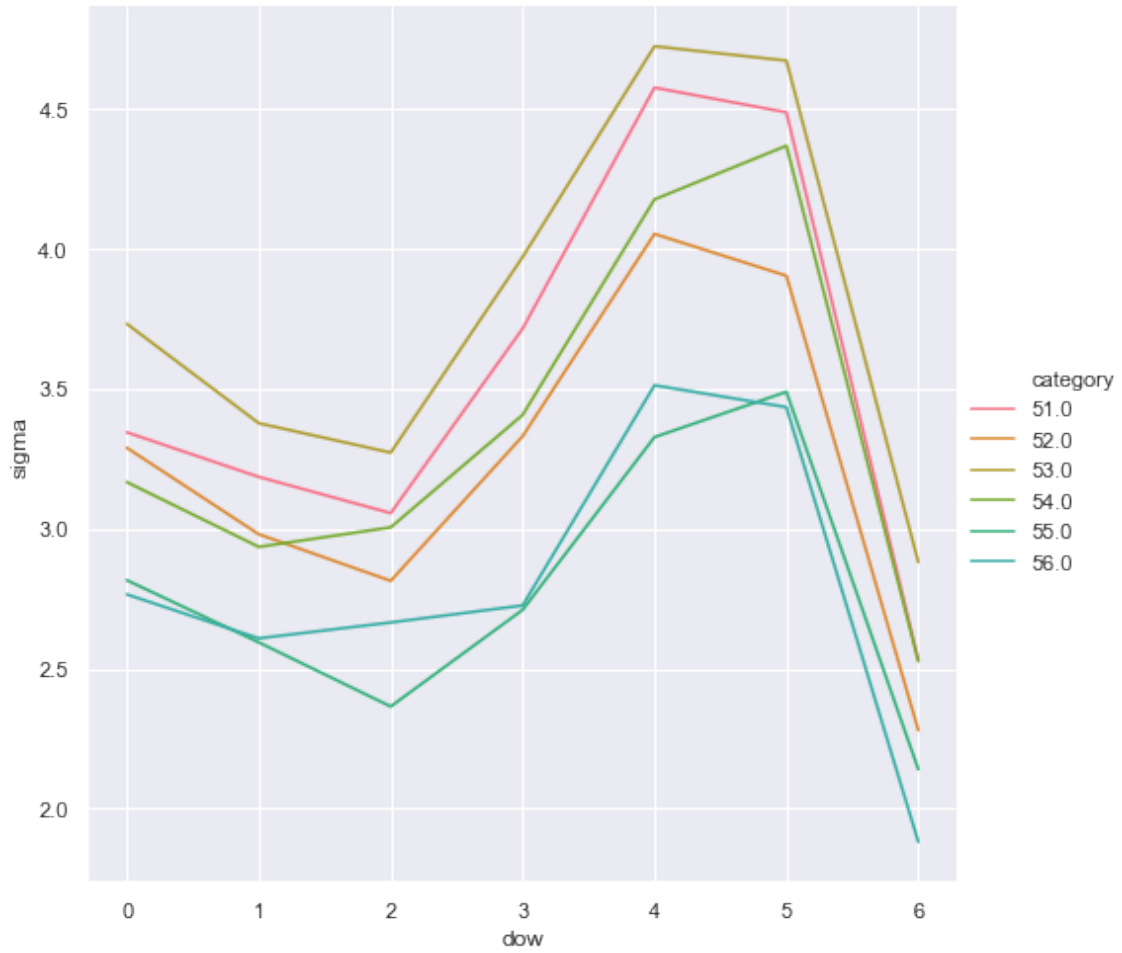

Figure 3: Weekly variation of σ parameter for several product categories considered.

Next Figures 4 and 5 plots the evolution of several quantities of interest for a specific product (id = 52786303). It uses the cycle between two Modern Stock Takes (MST) to train and makes predictions for the next 10 days. Notice the big discrepancy between nominal and estimated stock mean.

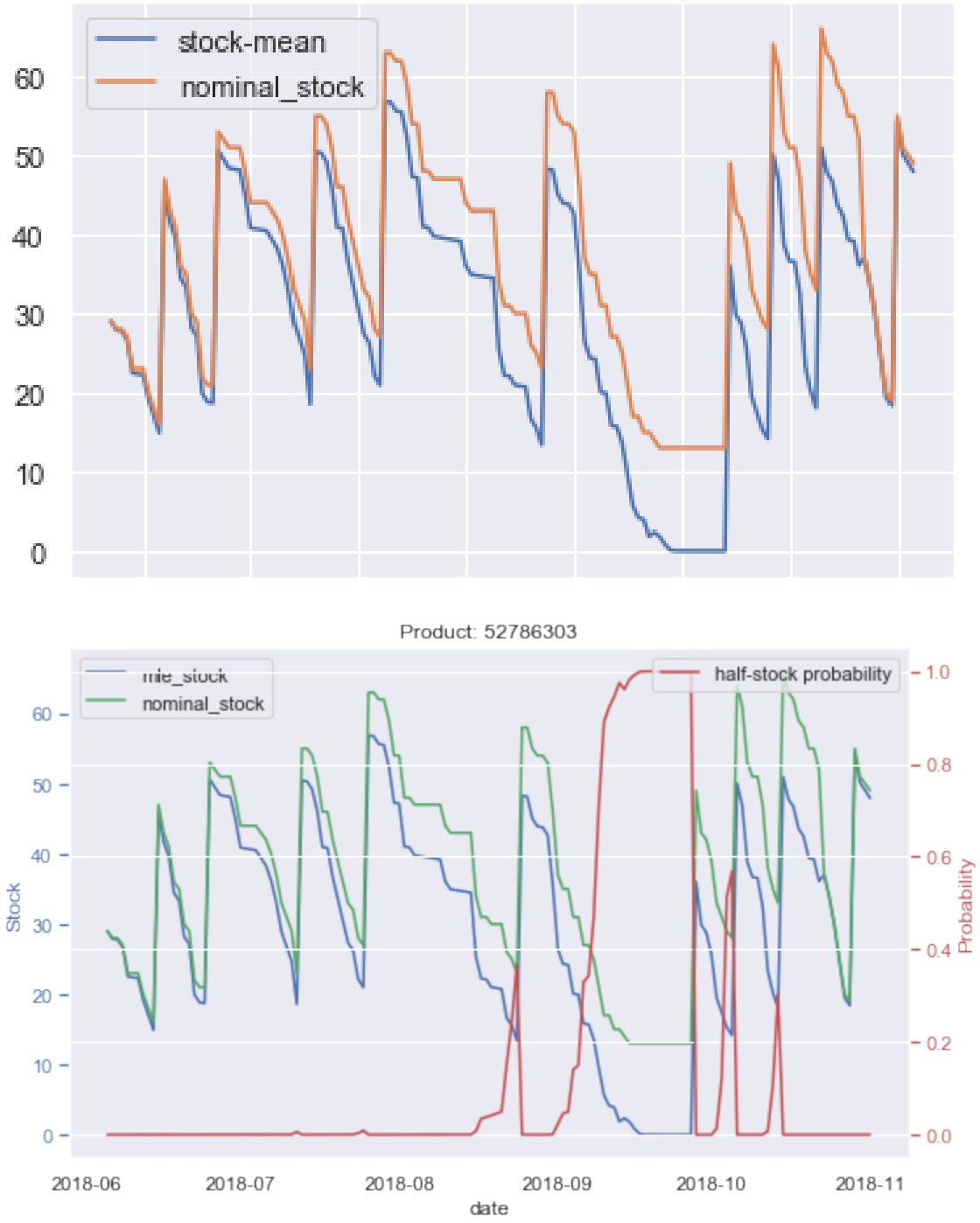

Figure 4: Daily variation of several quantities of interest for product number 52786303.

Figure 5 show an increase in uncertainty in stock levels near mid August. The reason for this increase is atypical sales and replenish pattern in this period.

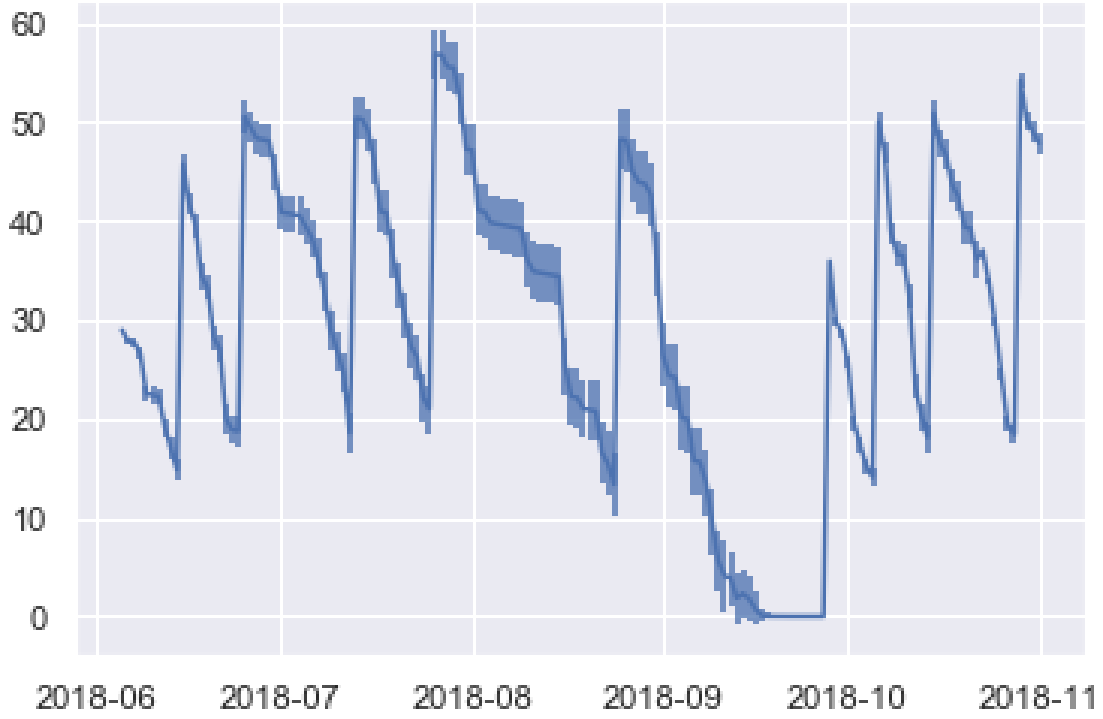

Figure 5: Daily variation stock mean and standard deviation (as error bars) for product id = 52786303.

9 Next steps

There are several lines of future research which can stem from the work presented. Another avenue of research is the extension of our model to one that can optimally decide on product re-orders and inventory inspections.

Below is a list of next steps that are suggested to increase the quality of the model.

- include product description for better clustering
- remove assumptions weekly behaviour
- Complete the Bayesian network with other nodes, like the unexpected waste.
- include uncertainty on the replenishment (either on quantities and time).

10 Glossary

- **mle_stock**: stock expected at the end of each the E-step.
- **stock-mean**: stock mean obtained from the distribution
- **stock-std**: stock standard deviation obtained from the distribution
- **Nominal_stock**: stock that would exist if we don't account for losses and corrections

- **half-stock-prob**: Probability that the stock is less than half.
- **stock-entropy**: how diverse in the stock distribution at a given moment in time.

References

[1] A Dynamic Bayesian Network Model for Inventory Level Estimation in Retail Marketing, https://arxiv.org/abs/1604.01075, (2016)

[2] Yun Kang and Stanley B Gershwin. Information Inaccuracy in Inventory Systems: Stock Loss and Stockout. IIE Transactions, 37(9):843859, 2005.

[3] Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques. MIT press, 2009.