Journal of the Operational Research Society

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tjor20

Prescriptive analytics for inventory management in health care

Leonardo Galli, Tommaso Levato, Fabio Schoen & Luca Tigli

To cite this article: Leonardo Galli, Tommaso Levato, Fabio Schoen & Luca Tigli (2021) Prescriptive analytics for inventory management in health care, Journal of the Operational Research Society, 72:10, 2211-2224, DOI: 10.1080/01605682.2020.1776167

To link to this article: https://doi.org/10.1080/01605682.2020.1776167

JOURNAL OF THE OPERATIONAL RESEARCH SOCIETY
2021, VOL. 72, NO. 10, 2211–2224
https://doi.org/10.1080/01605682.2020.1776167

ORIGINAL ARTICLE

Prescriptive analytics for inventory management in health care

Leonardo Galli, Tommaso Levato, Fabio Schoen and Luca Tigli

Università degli Studi di Firenze, Firenze, Italy

ABSTRACT

In this paper we propose an innovative optimization method for drug replenishment in a hospital ward. Although hospital logistics has in general shifted towards a zero local inventory policy, still most wards keep an inventory of drugs necessary for short time periods (between one and three days). The model we propose makes use of machine learning coupled with stochastic optimization in order to take into consideration both the historical usage patterns of drugs and the ward's current situation to minimize inventory levels as well as the necessity for emergency replenishments. Differently from generic inventory models, drugs are associated with patients who tend to consume a similar amount of drugs every day during their stay. Our aim in this paper was to show that, with suitably defined ward features, a machine learning method is capable of selecting a set of scenarios over which a stochastic optimization approach is capable of delivering good decisions, both in terms of small order quantities and of service quality (avoiding emergency replenishment orders). An ad hoc feature engineering procedure has been designed to exploit ward features, easily collected every day in the ward. Moreover, we introduce a new validation metric tailored for selecting hyperparameters to balance between optimality and robustness. Our results, based on a dataset composed of one year of drug administrations in an Italian ward, support the superiority of the proposed approach versus traditional ones.

ARTICLE HISTORY
Received 11 July 2019
Accepted 25 May 2020

KEYWORDS
Prescriptive analytics; machine learning; inventory optimization; stochastic optimization; *K*-nearest neighbors; decision trees; random forest; XGBoost

1. Introduction

Inventory models have been proposed in the literature for several different environments, depending on characteristics of the demand, of the inventory review process, of lead and setup times, etc. In Darling and Wise (2010) it is stated that good inventory management practices can generate huge savings in a hospital's operating budget, improving the quality of patient care and services. Although one of the aims of a hospital is to ensure that drug supplies are always available and accessible for clinical staff, inventory optimization did not seem to be a priority in healthcare decision making for a long time. In recent years, some of the healthcare systems have realized that there is a definite need to improve their inventory policies.

A recent survey on research in the field of quantitative methods for hospital logistics can be found in Volland et al. (2017).

In this paper we deal with the situation of drug inventories inside a hospital ward. It is a general trend, in modern hospitals, to reduce local inventories inside the wards and to replenish, based on the demand, from centralized facilities. It is however a common practice, (see e.g. Volland et al. 2017), to keep a relatively small inventory of drugs inside each ward and to issue replenishment orders on a regular, periodic basis, to cope with predicted demand; in case of a possible stock-out, an emergency order is issued and the required drug is delivered within a few hours of the request. It is somewhat surprising that so little attention has been given in the literature to the management of internal wards, with few exceptions (e.g. in Landry and Philippe 2004 and Volland et al. 2017: "the internal supply chain of hospitals remains the sore spot or weak link in supply chain integration"). It is also quite surprising that machine learning models have not yet entered the internal logistics in hospitals.

Recently, the availability of huge amounts of data is creating new opportunities for decision making under uncertainty, and the renewed attention towards machine learning provides new tools that can be used to generate predictions of quantities that may be of interest for inventory management.

Our aim in this paper is to propose efficient methods for the automatic replenishment suggestion of drugs with the two conflicting objectives of reducing emergency replenishment requests as much as possible and to keep inventory levels at the lowest possible value. This multi-criteria framework is quite common in hospital inventory optimization (see the recent survey on hospital performance measures in Moons et al. 2019). In Bijvank and Vis (2012), in the context of disposal, this conflict is

CONTACT Fabio Schoen fabio.schoen@unifi.it DINFO - Università degli Studi di Firenze, 50139 Firenze, Italy

addressed by proposing two different types of models, the first one focused on minimizing the total storage capacity required with a service level constraint, while in the second one capacity is constrained and the service level maximized. What makes real-world drug inventory management different from standard replenishment systems in production environments is the presence of highly non stationary demand. A common practice to account for such nonstationarities is to consider potentially meaningful auxiliary data to explain part of the variation of the demand, data such as the current situation of the ward, for example. In fact, patients usually spend quite a few days in the same ward and it is likely that a drug which has been administered to a patient in a day, will likely be necessary also in the next days, until the patient is discharged. So, a method which takes into account some statistics on the ward situation might be more accurate in predictions and more efficient inventory decisions might be taken.

It is also to be observed that much literature on general inventory systems (see, e.g. Syntetos et al. 2009) separates the predictive from the optimization phase: statistical tools are used to forecast future drug demand, and, based on this forecasted demand, inventory policies are built. In this paper we choose a model in which these phases are addressed jointly. Following a standard practice in modern prescriptive approaches, no point prediction is proposed. Instead, we select a set of scenarios which are in some sense similar to the current one, and the optimal decision is based on this subset of scenarios. Thus we have no single demand prediction for a specific planning horizon, but a set of forecasted demand values based on past history. Without a robust amount of different scenarios it would not be possible to avoid emergency replenishment that arises every time demand is underestimated by the machine learning model. In fact, inaccurate predictions are inherent in every regression task and for this reason point prediction is generally outperformed when we care about robustness. For this reason, the whole prescriptive procedure is under exam, while accurate demand prediction is out from the scope of this paper: our aim is not to predict correctly, but to efficiently manage inventories.

A central contribution of this work is also a newly designed validation metric to find a balance between robustness and optimality. Hyperparameters need to be selected properly in every machine learning application, but in the prescriptive framework this is even more important, since the number of scenarios directly depends on this choice. A new function that takes into account also the variance of the selected scenarios has been designed to replace classical performance measures.

The structure of this paper is the following. In the next section we will briefly recall some basic inventory optimization models; in section 3 we introduce the idea of prescriptive analytics methods based on machine learning coupled with stochastic optimization. In section 4 we describe the experimental settings and the data used in the numerical experiments, whose results are summarized next. Detailed numerical results are also reported in an appendix.

2. Inventory optimization

In this section we will introduce a basic inventory model and define the notation which will be used throughout the paper. As in this paper we will not consider joint replenishment decisions (i.e. we will consider each drug separately), in what follows we will avoid reference to the drug label and consider single item inventory models only.

Let us initially consider the case in which future demand $\{d_t\}$ for a drug is perfectly known in advance for a specific time interval. This case in unrealistic, but the corresponding model forms the basis for the following stochastic ones. Let z_t be the amount ordered in period (day) t, and let m_t be the inventory level at the beginning of period t: drugs in the inventory at time t are ready to be used to satisfy the demand of period t. We assume to operate with no capacity restrictions and zero-lead time of supply and thus, an order placed at the beginning of a period is immediately filled. Assuming a cost $K_t + C_t(z)$ is incurred when placing an order of z units in day t, where $K_t > 0$ is a fixed ordering cost which needs to be paid if and only if $z > 0$, and assuming that the daily inventory cost, depending on the inventory level m, is given by $H_t(m)$ in day t, the optimization model can be formulated as:

$$\begin{aligned}
&\min_{z,m,\delta} \quad \sum_{t=1}^{T} C_t(z_t) + H_t(m_t) + K_t\delta_t \\
&m_t + z_t = d_t + m_{t+1} && \forall\ t = 1, \ldots, T \\
&\delta_t \in \{0,1\} && \forall\ t = 1, \ldots, T \\
&z_t \le M_t\delta_t && \forall\ t = 1, \ldots, T \\
&z_t \in \mathbb{Z}^+ && \forall\ t = 1, \ldots, T \\
&m_t \in \mathbb{Z}^+ && \forall\ t = 1, \ldots, T.
\end{aligned} \tag{1}$$

Here the first constraint is the *inventory-balance* constraint; the initial inventory m_1 is assumed to be known in advance. The model is based on the optimization of the overall cost during a planning horizon T; although the cost functions C_t and H_t might be assumed to be concave, here we will consider only linear cost functions, for the sake of simplicity. Constraints $z_t \le M_t\delta_t$, with $M_t > 0$ large enough constants, link binary decision variables δ_t with the order quantity z_t and correspond to the logical implication

$$z_t > 0 \Rightarrow \delta_t = 1.$$

It is well known that if the cost functions are concave, the initial inventory is 0 (or small enough) and the upper bound constants are large enough to allow for any reasonable order level, then the model can be efficiently solved in polynomial time, e.g. by the Wagner–Whitin algorithm.

In the situation considered in this paper, the planning horizon T is usually very short (from one to three days) and in general it is assumed that a single regular order is issued at the beginning of the period. Although a more reactive ordering policy (e.g. one based on safety stock and reorder level) might be preferable from the point of view of cost, many wards operate in a periodic way to reduce the overhead associated to non periodic order management; the fact that we consider only very short horizons is due to the fact that in any case local inventories in the ward are discouraged, as replenishment is done from a central and large warehouse with efficient deliveries to all the region. In particular, it is frequently the case that regular orders are issued on Monday, Wednesday, Friday mornings (with planning horizons of 2, 2, 3 days, respectively). In this case the fixed purchase costs for days $t > 1$ can be set to a much higher value than that of day 1 to represent the inconvenience of emergency orders. In a deterministic situation like this one, it is very unlikely that an emergency order will ever be issued, unless the penalty for this kind of orders is very low compared, e.g. with inventory costs.

If, as it is the more natural setting, the demand is uncertain, the above model cannot be applied anymore. However, when the probability distribution of future daily demand is known, a stochastic optimization model can be defined. Let us assume that possible demand outcomes during the planning horizon can be represented as a finite set of scenarios S and that the probability of each scenario is known in advance.

We can then formulate a stochastic optimization problem with recourse in which a first stage decision z_0 needs to be taken now, without knowing anything about future demand outcomes, while, as days pass and future demand becomes known, possible recourse actions $z_1, z_2, \ldots$ (emergency orders) might be considered. We also assume that orders, either regular or emergency, produce an instantaneous replenishment; this is indeed the practice in the hospitals we have observed, and corresponds to the situation in which a central warehouse maintains a sufficient inventory of drugs which are delivered to hospitals either in the early morning for regular orders, or immediately for emergency ones. In any case, it is assumed that no shortages are allowed (and thus no shortage cost is added to the objective function). In this scenario, our decisions are non anticipative of future demand realization. This choice is mainly due to the desire not to add too many modeling assumptions to the ordering process; it is evident that, in case of an emergency order, if the remaining planning horizon is more than just the current day, an emergency order including also the expected future demand might (and typically will) be preferable. However this would have increased the complexity of the model, possibly hiding the main message of this paper. The idea here is that the first decision is affected by the possibility of future emergency orders required to ensure supplies are always available and, considering the particular choice of the coefficients, we find a value for z_0 that minimizes the expected negative effects of recourse actions. For what concerns emergency orders, the common practice in the hospital wards we analyzed is that, given a fixed schedule for placing orders, whenever the demand happens to be in excess with respect to the available stock, an emergency order is placed for a quantity which exactly matches the current unsatisfied demand. Thus, at the end of a day in which an emergency order has been placed, the remaining inventory is zero. We are aware of the fact that a second-level decision might be taken in these situations, but we preferred to maintain the current practice in Italian hospitals as a reference model, not to add, as we said before, too many modeling assumptions.

Differently from the deterministic model above in which, thanks to the perfectly known future demand, emergency orders can always be avoided, here we need to introduce this possibility. In practice we consider, as before, variables associated to drug replenishment, but now these variables correspond to emergency orders, issued as soon as the demand of the current day is known; the starting inventory, at the beginning of the planning period, is the quantity that needs to be chosen by the model without an exact knowledge of future demand. Let z_0 denote this main variable. Of course z_0 might take into account inventory already on hand, but we will assume, without loss of generality, that the starting inventory is zero; for a non null initial inventory this value might be considered as an up-to level. A formulation of the optimal decision problem is the following:

$$\begin{aligned}
&\min_{z,m,\delta} \; C_0 z_0 + \textstyle\sum_{s\in S} p_s \left(\sum_{t=1}^{T} C_t\left(z_t^s\right) + H_t\left(m_t^s\right) + K_t \delta_t^s \right) \\
&m_t^s + z_t^s = d_t^s + m_{t+1}^s \quad \forall\, t,s \\
&\delta_t^s \in \{0,1\} \quad \forall\, t,s \\
&z_t^s \le M_t \delta_t^s \quad \forall\, t,s \\
&z_t^s \in \mathbb{Z}^+ \quad \forall\, t,s \\
&m_t^s \in \mathbb{Z}^+ \quad \forall\, t,s \\
&m_1^s = z_0 \quad \forall\, s.
\end{aligned} \tag{2}$$

In this model, all relevant variables (emergency orders, inventory levels, binary decisions associated

to emergency order emission) have a further index s associated to a possible demand scenario and p_s is the probability that scenario s will occur. The last equation in the model is meant to represent the fact that the first purchase decision needs to be taken disregarding the scenario. This equation is needed in order to keep the notation simpler – we might easily reformulate the problem without variables m_1^s and without this equation, but in this case the inventory balance equation should be formulated in a different way in the current period with respect to future ones.

In the above model it can be seen that, after the decision z_0 has been taken, depending on the scenario s we might need to correct with an emergency order z_t^s which depends on the demand d_t^s which became known during time period t. In order to implement the policy of placing the minimum emergency order sufficient to avoid stockout at the end of the current day, we also add the logical constraint:

$$m_t^s \leq M_t(1-\delta_t^s) \ \forall \ t, s$$

so that whenever an emergency order is issued the value of the logical variable δ_t^s forces the end of day inventory to go to zero.

When a prior probability distribution cannot be assumed for possible scenarios, it is common practice, as we will see in the next section, to consider the historical frequency of the scenarios as the probability value of their occurrence.

It is well known (see, e.g. Ross 1983) that under mild assumptions the optimal solution to the above model is an (s, S) policy, which consists in ordering up to S when the inventory level falls below s. Much research has been devoted towards finding computationally feasible methods to compute optimal levels (Federgruen and Zipkin 1984), or to expand the models to include, e.g. stochastic lead times (Tsai and Ho 2019).

In this paper we prefer to directly use the above model, as it will form the basis for our new proposal; one of the main shortcomings of standard stochastic optimization models is the fact the no other information, except historical data, is used to reasonably estimate the expected cost. It is our aim in this paper to show that, taking into account the availability of contextual information, advanced prescriptive methods can be better suited to suggest optimal replenishment quantities, even without assuming stationarity of the demand process.

3. Prescriptive analytics

In the context of inventory management, demand is the key uncertain factor affecting decisions. Traditionally, as exemplified in the model just introduced, decision-making under uncertainty in Operations Research has largely focused on formulating and solving the stochastic optimization problem

$$z^* \in \arg\min_{z \in \mathcal{Z}} \mathbb{E}[c(z, Y)], \tag{3}$$

where a decision z in a *feasible* set $\mathcal{Z}$ is chosen to minimize the expected value of an uncertain cost $c(z, Y)$, that depends on the chosen decision but also on a random variable Y. The stochastic drug replenishment model (2) belongs to this class of optimization problems.

Generally, the techniques employed address the problem under a priori assumptions about the distribution p_y or, at times, assuming the availability of *independent* observations $(y^1, \ldots, y^m)$ drawn from the distribution.

In particular, the traditional approach for solving (3), when no a priori assumptions can be made on the probability distribution of the uncertain data, leads to the *Sample Average Approximation* (SAA) method. Here, the unknown true distribution is replaced with the empirical one, S_m, that assigns a probability $\frac{1}{m}$ to each available data point. SAA, then, solves the following optimization problem:

$$z^{SAA} \in \arg\min_{z \in \mathcal{Z}} \frac{1}{m} \sum_{i=1}^{m} c(z, y^i), \tag{4}$$

where the objective approximates $\mathbb{E}[c(z, Y)]$.

In contrast, when the decision maker has access to more information than demand history alone, the estimation problem can be solved more accurately. For example, information such as recent disease profile of patients, seasonal diseases, specific patients needs, or fast-moving pharmaceuticals are, intuitively, relevant predictors of the demand. Unfortunately, most data-driven inventory approaches do not consider auxiliary variables (features, covariates) that may improve the quality of the decision. Here, instead, we assume that an unknown joint probability distribution $p_{x,y}$ exists between the demand y and the features x used for predicting it, and that we have a sample of observations of size m drawn from this distribution.

More formally, we assume that we have access to observations

$$S_m = \{(x^1, y^1), \ldots, (x^m, y^m)\},$$

where x^i are features related to historical demand data y^i. Then, the main problem of interest becomes a *conditional stochastic optimization problem*, given imperfect observations, where the joint probability distribution that specifies the problem is unknown. In this context, an optimal decision z^* depending on the available covariate observation x can be obtained by solving the following optimization problem:

$$z^*(x) \in \arg\min_{z\in\mathcal{Z}} \mathbb{E}[c(z,Y)|X=x]. \quad (5)$$

The intuition is that taking into account covariates $x^1,\ldots,x^m$ our final decision will improve. To this end, we can use Machine Learning (ML) models to try to learn the relationship between the covariates and the demand. In traditional supervised learning algorithms, a regression model is selected based on the criterion of minimizing the empirical expected loss. This technique leverages the available data on X in predicting Y given the observation $X=x$. Given this prediction, one possibility might be to consider an approximation of the expected value for the random variable Y and then solving the corresponding optimization problem in a certainty-equivalent framework. However this approach suffers from strong limitations, as it does not take into account the remaining uncertainty after observing $X=x$ and also the fact that the conditional distribution tends to be degenerate. Moreover, most of the available information is lost when considering only the empirical conditional distribution.

For these reasons, we consider a set of non-parametric ML methods based on local learning, where predictions are based on the averaging, of some kind, of past observations that are in some way similar to the one at hand. This approach is based on rigorous theoretical justifications that support its effectiveness – see, in particular, Bertsimas and Kallus (2020). This kind of methodology is referred to as the *prescriptive* approach. A *prescriptive decision framework* has the general form:

$$z^{presc}(x) \in \arg\min_{z\in Z} \sum_{i=1}^{m} w(x^i)c(z,y^i) \quad (6)$$

where $w(\cdot)$ is a weight function derived from the data and x is the current observation of the feature vector. The values for the weights w are "learned" through a ML approach by analyzing the correlation between past observations of the uncertain quantity to be estimated along with the associated values of the covariates. As a simple example, $w(x^i)$ might be chosen as a constant when x^i belongs to a specific neighborhood of the current observed covariate x and zero otherwise. The most common ML approaches used in this context are inspired by *K*-nearest neighbors (*K*NN), Decision Trees and Random Forests.

It is important to remark that the approach we are following is based on the use of ML not as a predictive tool, but more as a selection tool which is capable of identifying a set of scenarios, in the past data, which are close to the current one. The stochastic model to be solved becomes, thus, a stochastic optimization model based on a conditional probability distribution whose support consists of a subset of carefully selected samples, instead of the whole data history.

We now very briefly describe the core ideas behind these machine learning models – for a more detailed discussion about these methods, see, e.g. Hastie et al. (2009).

We would like to stress the fact that all of these methods are "instance-based", meaning that each prediction is constructed taking into account only a limited portion of the training data. In other words, these methods naturally fit into the prescriptive approach described earlier. While other machine learning models can be adapted to this specific methodology, the possibility to use "off-the-shelf" instance-based models is surely a bonus. In what follows, besides a short introduction to a few ML methods, we also declare the hyperparameters that we decided to tune for each of the proposed methods. Our choice was to select a tiny subset of possible hyperparameters, in order to reduce the computational effort needed; other hyperparameters, not cited in the paper, were kept at their default value.

3.1. KNN

The *K*-nearest neighbors (*K*NN) model is arguably one of the simplest machine learning algorithms, at least from a conceptual point of view. This learning algorithm is memory-based: the training phase simply amounts to memorizing the training set. Given a test sample x, the predicted value y is obtained by looking at the K training samples x^i that are closest, in feature space, to x, and by averaging – in some way – their associated labels y^i. Usually, the Euclidean distance metric is used to find the K training examples to consider, and their labels y_i are combined by taking their arithmetic mean. While it is an extremely simple method to describe, *K*-nearest neighbors is actually a powerful non-parametric model, that can produce complex decision functions. In the context of prescriptive methods, *K*NN is used to derive weights in (6): in particular the weight $w(x^i)$ associated to each of the *K*-nearest neighbors of x is equal to $1/K$, while it is 0 for each other data point

$$w(x^i) = \frac{1}{K}\mathbb{1}(x^i \text{is a } K\text{NN of } x). \quad (7)$$

In this way, the prescriptive approach takes into account a sample average approximation restricted to those samples which are in some sense as close as possible to the current one. If K is chosen very large, the approach progressively tends to the Sample Average Approximations method (4). The only hyperparameter in *K*NN is K, the cardinality of the neighborhood considered by the algorithm.

3.2. Decision trees

Decision trees build regression models in the form of a tree structure. The dataset is broken down into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. In simple terms, each node of the tree partitions the training set, based on the value of a single attribute. For example, given a dataset of patients and their clinical measurements, a node might divide patients with high blood pressure from those with normal or low blood pressure. Terminal nodes, or leaves, are then labeled with an average of the labels of the training examples associated to that leaf. For regression tasks, usually, the arithmetic mean of the labels is used. Given a test example, it is then possible to "navigate" the tree, choosing the correct branch at each stage, and output the label associated to the terminal leaf node.

Building an optimal decision tree (according to some optimality criterion) is an NP-Hard problem (Hyafil and Rivest (1976). However, greedy methods, such as the well known CART algorithm (Breiman et al. (2017), usually perform reasonably well in practice. Moreover, when compared to other machine learning methods, decision trees are easily interpretable, which in certain contexts is certainly desirable.

In the context of prescriptive methods, a decision tree can be used to select a constant weight associated to each observation x^i which belongs to the same leaf node as the current observed one x:

$$w(x^i) = \frac{\mathbb{1}(\mathcal{R}(x^i) = \mathcal{R}(x))}{|\{j : \mathcal{R}(x^j) = \mathcal{R}(x)\}|} \tag{8}$$

where $\mathcal{R} : X \to \{1, ..., L\}$ is the partition rule that provides L disjoint regions. In a sense this is not very different from the previous approach, the main difference being the definition of a neighborhood: here the neighborhood is induced by the decision tree. While training our decision tree models we considered the following hyperparameters, whose names refer to those of the `scikit-learn` Python library (see Pedregosa et al. (2011)): `min_samples_leaf`, which corresponds to the minimum number of sample points which can be associated to a leaf of the tree; in other words, a split which leaves less than this number of samples in one of the two generated descendants is not allowed. In a sense, this is quite related to the neighborhood size. The other, but less relevant, parameter we varied was `max_depth`, the maximum allowed depth of the tree.

3.3. Random forests & XGBoost

The main principle behind ensemble methods, like random forests, is that a group of "weak learners" can come together to form a "strong learner". A single regressor, individually, can be seen as a "weak learner", while more regressors taken together are a "strong learner". Random forest (RF) is an ensemble approach where the weak learner is a decision tree, so a set of trees is what composes the forest. In order to have diversity among the trees of the forest, for each of them, only a subset of the whole data and only a subset of the features are randomly chosen. These techniques, called *tree bagging* and *feature bagging*, select a random sample with replacement of the training set and fit trees to these samples. After training, predictions for an unseen sample x can be made by averaging the predictions obtained from each of the individual regression trees on x. The key idea is that each tree is slightly different from the others and the variance is reduced without affecting the bias of the model too much, so that when averaging the results a reasonably good answer is produced. In the context of our prescriptive approach, given the current scenario x, the weight associated to a sample x^i is computed on the basis of the percentage of all trees T in which x belongs to the same leaf as x^i.

$$w(x^i) = \frac{1}{T}\sum_{t=1}^{T} \frac{\mathbb{1}(\mathcal{R}(x^i) = \mathcal{R}(x))}{|\{j : \mathcal{R}(x^j) = \mathcal{R}(x)\}|} \tag{9}$$

Training a random forest usually requires setting quite a few of hyperparameters; in order to cope with the most relevant ones and not to unnecessarily augment the required training time, we choose to vary a single hyperparameter, `min_samples_leaf`, already defined above. All other parameters were kept fixed, including the maximum depth of each tree as well as the total number of trees.

XGBoost is another ensemble method based on decision trees. This method is an implementation of the gradient tree boosting algorithm, originally proposed in Friedman (2001). Essentially, after training an initial tree in the usual way, we train new trees fitting each one on the errors committed by the "forest of trees" built right before it. For instance, when fitting, say, the fifth tree, the label for each training example is the difference between the actual training label of that particular example and the prediction that the ensemble of the first four trees associates to that example. Note that only the labels change, while we can exploit the usual training algorithms (e.g. CART) for fitting each decision tree. For more details, see Chen and Guestrin (2016).

The usage in the context of a prescriptive approach is similar to that of weights based on random forests. For what concerns hyperparameters, we choose to consider during training a single one, identified by `min_child_weight` in the official XGBoost documentation; this parameter in our

context has the same meaning as the `min_samples_leaf` defined above; other parameters, like, e.g. the maximum depth, the number of trees, as well as the learning rate, were kept fixed. Summarizing, our choice of the hyperparameters to be varied during the training phase was guided from one side by the desire of keeping training time at reasonable levels and on the other to select those parameters which were most relevant for the task at hand; in all of the methods the most important one, if not the unique, has been the definition of the cardinality of the neighborhood.

4. Experimental setting

4.1. Data and tasks

In the approach proposed in this paper, the available data consist of historical records of drug administrations. The data on which we tested our approach was based on one year of records from the sub-intensive ward of an Italian hospital in the Tuscany region. Although the total number of drugs in the records is quite high, a large amount of them is administered only a few times during the observed year. Thus, we decided to focus our analysis on the ten most frequently consumed drugs. The following simplifying assumptions are considered:

- in order to work with a larger amount of administrations, we assume that it is directly possible to order and administer the active ingredient, which is indeed shared among several drugs with different commercial names;
- the administered quantity is often a standard drug-specific amount of mg or ml (or other units of measurement); for this reason we assume that each administration is an integer number of "doses", where a dose is a standard unit for the active ingredient.

Each record is characterized by the following information:

- name/id of the (active ingredient of the) drug,
- id of the patient,
- date (and hour) of the administration,
- gender of the patient,
- age of the patient,
- administered quantity.

We do not have, nor we tried to estimate, any information regarding correlations between different drugs and we don't have access to the therapies of the patients. For this reason each drug is treated independently from the others. We chose to base our models on these data as they are readily and unambiguously available from administrative records, which we assume can be easily accessed at the ward level; inclusion of therapeutic programs or diagnoses or other treatment–specific information might possibly improve the quality of the model, but at the expense of much more restricted possibility of usage. This data might even be collected within the ward, without any necessity to rely on external data. We built a dataset for each drug. Machine learning was employed in the form of regression tasks, as the quantity to be predicted is the daily amount of administrations of the selected drug.

We took into account three different planning horizons for replenishment, from one to three days; one day-ahead prediction is indeed not particularly useful in our context as orders are issued on Mondays, Wednesdays, Fridays, with 2, 2, 3 days ahead, respectively – the planning horizon of one day has been included for the sake of completeness. In the prescriptive approach, several scenarios are considered, each one characterized by a demand pattern and a probability, or weight. The demand for a planning horizon composed of more than one period is an array; thus we associated to each scenario a label consisting of a vector of the same length of the considered horizon ($h = 1, 2, 3$) that contains the administration quantities to be required for each one of the next h days. We had also tested an alternative model, which we called *Aggregated*, in which the label represented the total quantity of drugs to be administered in the next h days. However the experiments we performed did not lead to significant differences with respect to the original, day by day, model; thus in this paper we will not present results on the *Aggregated* model.

In the experiments, with reference to model (2), we used the following penalties: for all $t \geq 1$ ordering costs C_t were defined as 0.1 per unit and the fixed cost has been set to 5, in order to penalize emergency orders; the cost associated to the initial order quantity z_0 was set to 0. Thus a high penalty is associated to emergency orders, to be issued in future periods within the planning horizon. Holding costs were kept fixed at 0.01 per unit per day, measured at the end of each day. These values were chosen slightly taking into account observed situations in Italian hospitals; however they can be considered as somewhat arbitrary, apart from the fact that we wanted to stress the fact that an emergency order needs to be penalized orders of magnitude more than the cost of storage. We performed a limited set of experiments with different values for these constants, but we could not see any significant difference in the relative performance of the different approaches.

4.2. Feature engineering

In order to face the above tasks we need to characterize each day with a collection of features. In fact, the approach we followed in this paper is to observe, every day, the situation of the ward (and not of any specific patient) and to summarize this situation in a number of features which are likely to be correlated with future drug consumption at the ward level.

The first feature, extracted daily from past historical data, is a count of the total number of doses administered for the drug under consideration on that day (`num_adm`). Note that this information is correlated with the number of drug-specific patients which are hospitalized in the ward. The second characteristic is the percentage of administrations given to women with respect to the total amount of administrations given that day (`gender_perc`). From the id of the patient it was possible to compute a drug-specific length of stay (LOS) for each of them and the total amount of drugs given to him/her up to the current day. This value was then combined with the computed LOS in order to find the dosage per day for each patient. The LOS, the dosage and the age have been exploited to characterize the situation in the ward. The procedure employed is the same for all the three informations above: an overall average was first computed on the whole dataset; then, for each day, the percentage of administrations given to patients with higher age/dosage/LOS with respect to the computed mean (`age_perc`, `severity_perc`, `long_los_perc`) was computed. The idea behind these three features is that of trying to characterize days in which the ward has to deal with patients which have higher needs than normal. Thus we defined a total of five features associated to each drug for every day. We did not try to experiment with different aggregation of data like, e.g. frequency histograms, or with other ways, possibly connected to causal factors. The approach we followed here was that of having a simple, small, readily available set of features to guide the optimization process.

4.3. Validation

Once the drug and the time horizon h are chosen, the 365 entries of the dataset are fully defined by a feature matrix $X \in \mathbb{R}^{365\times 5}$ and a label matrix $Y \in \mathbb{R}^{365\times h}$. As it will be explained in the following, the last portion of the dataset (*test_set*) is used for testing, while the first part (*training_set*) is used for training the models and validating their hyperparameters. As the dataset contains only 365 examples, we decided not to skip any rows when the horizon is $h = 2$ or 3; by this we mean that, for $h = 2$, we used day 1 and day 2 for a first sample, days 2 and 3 for the second, days 3 and 4 for the third and so on. This leads to a dataset in which examples are not stochastically independent from each other. In general, this might be a disadvantage for machine learning models that do not exploit the sequential structure of data, as those employed here, but the loss of observations would otherwise have been too drastic (half of the dataset with $h = 2$, 66% when $h = 3$). Such a setting is pretty common in the machine learning community, since it is assumed that performances obtained on each test example does not depend on the outcome of a previous application of the model (as in the opposite case of evaluations based on simulations). Another consequence of this choice is that we could perform only very short-term simulation runs both in training and in testing, as opposed to what might have been preferable if we had much longer time series. Given the overlapping between consecutive periods, we also had to assume that the initial inventory for every period was 0. Of course, in practice, when using a method like the one proposed here, if a non zero inventory is observed when a new order is issued, the order quantity will be reduced accordingly.

Accurately choosing hyperparameters is often a crucial step in the machine learning process. In this context, hyperparameters are even more important, since they select the dimension of the subset of the scenarios to be used in the optimization procedure. The validation procedure we employed is a 5-fold cross validation. In general, this is not the most indicated choice for data with a time structure, but again, the machine learning models do not exploit the sequential structure of data. This validation setting enables us to exploit the whole *training_set* and obtain a more robust validation score for the hyperparameter selection.

A standard metric employed for regression tasks is the Mean Absolute Error (MAE):

$$MAE(Y_{pred}, Y_{true}) = \frac{1}{M}\sum_{i=0}^{M} |y^i_{pred} - y^i_{true}|,$$

where M is the cardinality of the *training_set*. The machine learning models employed in this study have the peculiarity that the value y^i_{pred} is obtained through the (weighted) mean of the labels associated to the points belonging to the same subset of the considered testing point: a Voronoi cell for *KNN*, a leaf for DT or various leaves for RF or XGBoost. Moreover, this point prediction y^i_{pred} is not used by the optimization procedure. On the contrary, the optimization model directly uses the labels associated to the points that fall in the same subset of the considered testing point. For this reason we were led to consider a different metric, namely Neighbor MAE (NMAE), that does not address y^i_{pred} but the original values from which it is obtained: for each x^i this set ($\mathcal{N}^i$) contains all the

labels associated to the points that belong to the same subset of x^i. Note that since we are using a K-fold cross validation, those points are only obtained from the K-1 folds used for the actual training. The metric NMAE is defined as follows

$$NMAE(\mathcal{N}, Y) = \frac{1}{M}\sum_{i=0}^{M}\frac{1}{|\mathcal{N}^i|}\sum_{y^j\in\mathcal{N}^i}|y^j - y^i_{true}|.$$

In a classical machine learning application, a measure of the error made by the model is often enough to understand how well is the model behaving. In this study the situation is different, because the final decision is taken by an optimization model which needs to take into account various possible scenarios in order to make a robust choice. For this reason, selecting the hyperparameters that minimize the error is not enough, as this would often lead to choose small neighborhoods. In order to select more robust neighborhoods we decided to subtract a measure of the standard deviation from each term in the summation. This produces a situation in which the best hyperparameter is the one which produces the best trade-off between a neighborhood with accurate labels or one with high variance. In a certain sense, the metric below reflects the main concept behind this study: find a trade-off between optimality and robustness. With this in mind, the two elements, error and standard deviation, have been scaled using the probability of over ($\mathbb{P}_o$) and under ($\mathbb{P}_u$) estimating the demand, respectively. The idea is that when the probability of underestimating is high, it is better to pick a larger neighborhood and make a safer choice, while in the opposite case it is better to reduce uncertainty and make a choice which is closer to optimality. The actual values of $\mathbb{P}_o$ and $\mathbb{P}_u$ are computed by taking the ratio of the number of scenarios that over/under-estimate the real demand y_{true} with respect to the total number of neighboring scenarios:

$$\mathbb{P}^i_o = \frac{\left|\{y\in\mathcal{N}^i | e^\top y > e^\top y^i_{true}\}\right|}{\left|\mathcal{N}^i\right|},$$

$$\mathbb{P}^i_u = \frac{\left|\{y\in\mathcal{N}^i | e^\top y \leq e^\top y^i_{true}\}\right|}{\left|\mathcal{N}^i\right|},$$

where $e\in\mathbb{R}^h$ is the vector of components all equal to 1.

The resulting metric, namely Weighted Error-Variance (WEV), is

$$WEV(\mathcal{N}, Y) = \frac{1}{M}\sum_{i=0}^{M} -\mathbb{P}^i_u\cdot std(\mathcal{N}^i) + \mathbb{P}^i_o\cdot\frac{1}{|\mathcal{N}^i|}\cdot\sum_{y^j\in\mathcal{N}^i}|y^j - y^i_{true}|.$$

It is important to remind that this definition was only used in order to guide the method towards a specific and reasonable choice of some hyperparameters, during the training phase. The actual performance of the prescriptive approach will be measured in terms of standard statistics on the order quantities and on the service quality measured on a test set which is not used in the training phase. One may argue if this non standard metric is indeed producing the expected results. We performed a large set of numerical experiments: those based on the usage of our proposed WEV metric are reported in the following sections. We chose not to present analogous results for optimization models based on the traditional metric in order to keep the number of numerical results at a reasonable level. In all of our experiments we indeed observed that the use of the new metric indeed had the desired effect. By this we mean that, while the optimization based on the traditional metric usually produced a more precise demand estimate and, thus, a lower inventory level, for all the methods we tried, our new metric produced much lower numbers of emergency orders which, in our models, is the most relevant component of the cost.

5. Numerical experiments and results

To test the prescriptive framework, we compared it against the Sample Average Approximation (SAA) approach. In fact, our main focus is that of showing that an accurate choice of scenarios can improve performances. In particular, we show results for all the machine learning methods we described in the previous section, namely: *K*NN, Decision Trees, Random Forests, XGBoost.

The experiments we show here have been performed according to the following scheme:

1. partition the dataset into a *train_set* and a *test_set*.
2. while *test_set* $\neq\emptyset$:
3. Select a set of *scenarios* from the *train_set* using a stochastic or a prescriptive approach.
4. Use the above set of *scenarios* as input for the stochastic optimization model. Solve it and find the optimal initial order quantity z_0
5. Remove the first *true_scenario* (actual demand for *h* periods) from the *test_set* and evaluate the policy associated to the choice of z_0 with the actual demand.
6. Add the *true_scenario* to the *train_set*.
7. end while

Note that, in step 3, we train a new model for each example in the test set. While this is, in general, not

standard for testing machine learning models, in the case of forecasting it is quite realistic.

The quantity z_0 represents the final output of the whole procedure and the actual element to be tested in this phase: in a real application this value would tell us how much to order at the beginning of the planning horizon. In the model, this value comes from the decision variable z_0 and it is chosen by the optimizer. There's a practical reason behind this choice: since multiple scenarios take part in the final decision, different order policies z_t^s are returned by the optimizer, and picking a single one is a nontrivial task. Instead, we force the different scenarios to agree on how much to order on the first day, before any real demand is revealed. Thus, the outcome that is tested is the one obtained by ordering a suitably chosen amount on the first day of the horizon.

The initial split assigns the last 20% of the dataset to the *test_set* and the initial 80% to the *train_set*, but, as explained above, the cardinality of the training set increases until it reaches the dimension of the entire dataset[1]. Also in this phase, no testing days were skipped when $h \neq 1$ and each optimization is performed independently from the previous one. Thanks to this setting, it is possible to test the whole procedure on a robust section of data and, in addition, compare different horizons on the same amount of testing days.

In evaluating the different approaches, we consider two metrics: the total initial order quantities and the number of emergency orders. In fact, in the testing phase, we don't use the objective function of the optimization phase as a result metric, in order to stress the bi-objective nature of the decision problem. In other words, while during the training phase we optimized an objective function corresponding to a compromise between the two objectives, in the testing phase we choose to show the two components separately. This way, even if we optimized an aggregate objective, we can measure if and to what extent the resulting policies give good results for each of the two objectives. The order quantity z_0 for the first day and the necessity for emergency replenishment during the planning horizon are the two relevant metrics. The overall objective function strongly depends on these two quantities – we preferred to show these two components separately in order to have a better way of appreciating the relative strength of different methods.

We compared the methods using performance profiles (Dolan and Moré (2002).Without any claim of completeness, for which we refer the reader to the cited paper, we would like to recall here that performance profiles are a standard tool, in the optimization literature, to compare the behavior of different algorithms. In each plot, given a value $\bar{x}$ on the x–axis, the corresponding y–value on the vertical one associated to the curve representing a specific method corresponds to the fraction of problems that the algorithm is capable of solving with a performance which is within $\bar{x}$ times with respect to the best performing method for each problem. In other words, if for a method m the line at $\bar{x}$ has the value $\bar{y}$, this means that for a percentage $\bar{y}$ of all test cases, the performance considered in the graph (e.g. the number of emergency orders) obtained by this method was either the minimum with respect to that obtained by all other methods, or, if worse, in any case the ratio with respect to the best was no more than $\bar{x}$ times worse. Thus, at $x = 1$ we can read for each method the percentage of test cases in which the method had the best performance (these percentages do not sum to 1 as there might be ties among different methods). At, say, $x = 1.5$ we can read the percentage of test cases in which each method displayed a performance which was within a factor 1.5 with respect to the best one, and so on – thus, in this case, we can observe the percentage of runs in which a single method was at most 50% worse than the best one in a case by case comparison. As it is quite common in the literature, performance profiles will be displayed using a logarithmic scale for the x axis. When looking at performance profiles we can easily obtain information on the efficacy of the methods (at $x = 0$), but also on their "robustness": sometimes a method is almost never the best one, but it might perform sufficiently well in most cases, while another one, which frequently is the best one, might display a very bad behavior in other cases. From performance profiles we can get this kind of information; as a general rule, the overall best algorithms are those for which the graph is "top left".

Performance profiles have been computed on both the order quantities and the number of emergency orders in the out-of-sample experiments. In other words, the data summarized in the figures is obtained by measuring the performance of the trained methods on "future" test data, which was not used to train and calibrate the models. Figure 1 reports the results obtained considering every planning horizon with the model. Single performance profiles computed separately for each horizon are reported in Figure 2.

First of all, it is worth noting that the results obtained for what concerns the order quantities (associated to the inventory) and emergency orders are, generally, in contrast. This was expected, as our problem is indeed a bi-objective one in which we would like to minimize both emergency orders and inventories, and these objectives are usually in contrast to

[1]To be more precise, for the last test example the training set consists of all the other examples in the dataset.

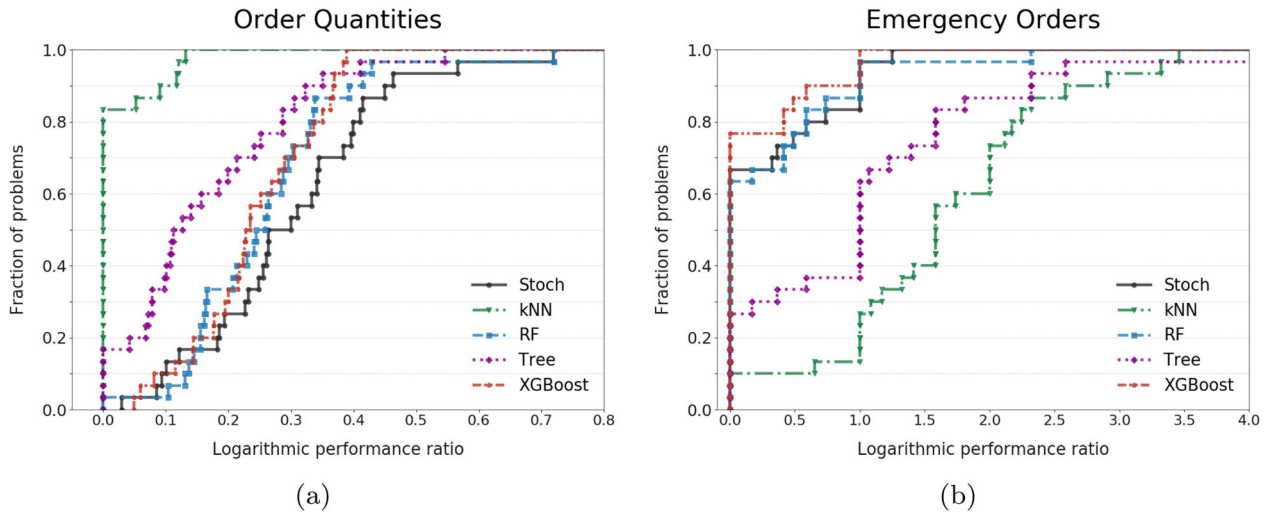

Figure 1. Performance profiles obtained considering every planning horizon.

one another. An aggressive policy which tries to minimize the order levels might induce more emergency orders than a more conservative strategy. The stochastic method based on Sample Average Approximation (4) is conservative by design, since it considers every observed past historical data in its decision. For this reason it is expected to incur in fewer emergency orders with respect to more focused strategies like the prescriptive approach. Nonetheless, the cost savings provided by ordering fewer products, in terms of ordering and holding costs, can outweigh the cost of a few emergency orders. This balance has to be carefully considered and evaluated when choosing the methodology of ordering for the task at hand. Considering the conservative property of the stochastic approach, it was surprising to see that the performance of some of the prescriptive approaches is better than that of the stochastic one even for what concerns emergency orders (see Figures 1(b), 2(e) and 2(f)). On the other hand, the stochastic approach is uniformly worse than all of the prescriptive ones for what concerns the order quantities.

Looking at Figure 1 specifically, the results generally agree with the previous discussion. For example, the *K*-nearest neighbors strategy minimizes order quantities, by a tangible margin, which results in a significantly higher number of emergency orders. Also decision trees show the same pattern of performance. However, both Random Forests and XGBoost dominate the stochastic method, being able to reduce both the order quantities as well as the number of emergency orders. Moreover, the results obtained show the flexibility of the prescriptive framework, which can be an advantage in light of the previous discussion. Here, by "flexibility" we mean the availability of different methods whose application has a different, yet quite predictable, effect on the quality of the two conflicting objectives – depending on the context, one user might be able to choose the method which best fits their needs. In other words, the emphasis on smaller orders versus fewer emergency orders depends on the specific context in which these tools are used.

The former argument is generally confirmed also when looking at the individual planning horizons, in Figure 2. When comparing results from the point of view of Pareto dominance, the stochastic approach is almost always dominated by methods based on the prescriptive approach. In particular, for longer planning horizons $h = 2, 3$, performance profiles suggest that random forests and XGBoost outperform the stochastic approach. However, when looking also at the detailed numerical results in the Appendix it is readily seen that all methods based on machine learning are usually preferable to the standard stochastic one. By this we mean that in most cases prescriptive approaches gave better results both in terms of the total ordered quantities, and in terms of emergency orders issued, as well as in the objective function value. However, given the bi-objective nature of the problem, there is no single winner: *K*NN-based methods are very good in generating low order quantities (and thus low inventories) at the expense of high emergency order events; random forests and decision trees are usually quite good in having a very low emergency order rate coupled with a reasonably low order quantity. XGBoost, although not being in general the best performing one for neither of the two objectives, is quite robust and not far from optimality in both criteria, as it can be seen from the performance profiles. As a further possible way to compare all of the proposed methods, in Figure 3 we report the performance profiles of all methods for what concerns the value of the objective function. Here a single profile is used to compare all methods on all drugs for all horizons.

In summary the results show that the prescriptive framework is a suitable approach for inventory

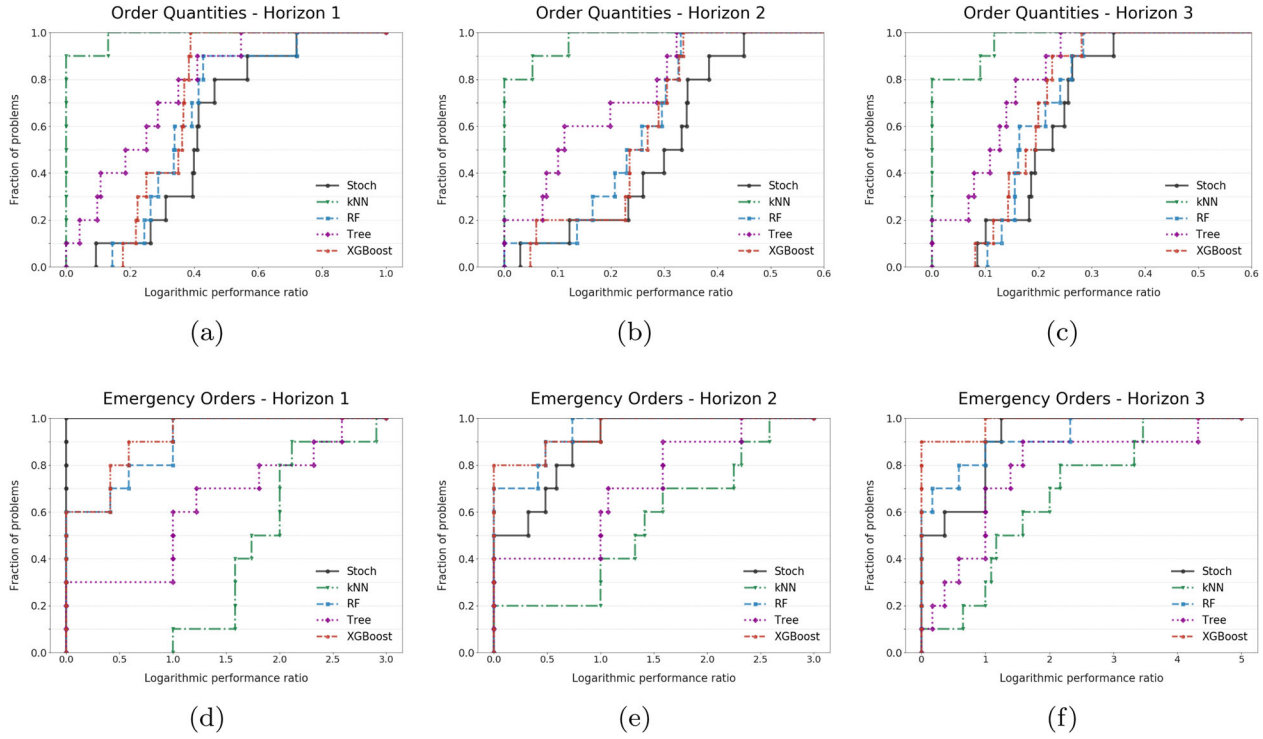

Figure 2. Performance profiles computed separately for each horizon.

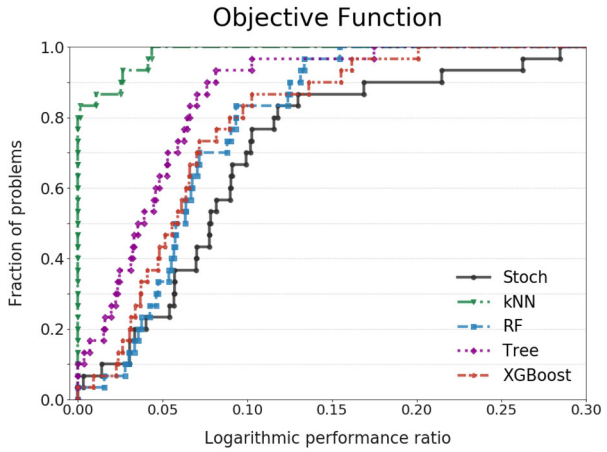

Figure 3. Performance profiles obtained considering every planning horizon.

management, both in terms of absolute performance and flexibility, giving the possibility to adopt different strategies depending on the task at hand.

6. Conclusions

We have proposed a method in which a stochastic optimization model is adopted for replenishment decisions in a hospital ward. What makes drug inventory management different from other contexts is the fact that the demand for drugs is correlated with the number, type and length of stay of patients in the ward. Thus our aim has been to check whether including these features was indeed beneficial with respect to a classical data-driven stochastic model. The evidence we could collect from our experimentation on real data supports the fact that by properly defining features associated to the daily ward situation it is possible to significantly improve the quality of replenishment decisions, both in keeping a low level of inventory as well as in reducing the probability of emergency orders. From our experiments, it seems that Random Forest and XGBoost-based approaches dominate the others we have tested and found a good compromise between the two conflicting objectives of reducing emergency replenishment orders as well as order quantities. It is important to observe that we could improve over the standard inventory approaches by simply exploiting quite a small quantity of information on the ward. In particular, we did not take into account prescriptions, specific therapeutic information, severity indices, seasonality, or any detailed information on each patient. Simply taking into account aggregated information on the whole ward we could obtain improvements in the quality of drug management, which, in turn, corresponds both to lower costs for the hospital as well as improved service to the patient.

We would like to stress the novelties of the approach proposed in this paper:

- we applied the prescriptive approach to the context of drug replenishment in wards, a critical and delicate point of the health care supply chain;
- our proposal does not separate the (point) prediction of demand from the optimization of

order quantities. Instead prediction and replenishment decision are integrated;
- inventories in a hospital ward need to follow and forecast a demand which is in general nonstationary, which makes standard policies less effective;
- differently from published prescriptive approaches, we used an innovative scoring function to train our machine learning models, taking into account in some way the multiobjective nature of the problem at hand;
- an ad hoc feature engineering technique has been developed to extract suitable observations to feed the machine learning model and obtain scenarios which are actually based on the current ward situation. Information required in this step is easily found in ward records;
- differently from many approaches in the literature on inventories, we do not rely on point prediction, nor in the overall probability distribution function of the demand. We do not use point prediction, as it does not take into account variability in demand. We also don't use the complete empirical distribution, as it cannot capture the nonstationarity of the demand process and its correlation with the ward situation (number of patients, severity indices, gender, …). Future developments might include taking into account obsolescence effects on the available drugs (see,e.g. Baron (2010); Coelho and Laporte (2014); Dobson et al. (2017); Nahmias (2011)); however, it is the authors' opinion that by suitably reducing inventory levels the problem of expiry dates on frequently used drugs becomes less and less relevant. Another line of research might possibly consider other advanced decision techniques like, e.g. approaches based on reinforcement learning.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

The research presented in this paper was partially supported by "Project LINFA - Logistica Intelligente del Farmaco" - Tuscany Region - PAR FAS 2007-2013, Linea d'Azione 1.1 - Bando FAR FAS 2014.

ORCID

Leonardo Galli http://orcid.org/0000-0002-8045-7101
Tommaso Levato http://orcid.org/0000-0002-4616-8277
Fabio Schoen http://orcid.org/0000-0003-1160-7572
Luca Tigli http://orcid.org/0000-0001-9913-9843

References

Baron, O. (2010). *Managing perishable inventory*. Wiley Encyclopedia of Operations Research and Management Science.

Bertsimas, D., & Kallus, N. (2020). From predictive to prescriptive analytics. *Management Science*, *66*(3), 1025–1044. https://doi.org/10.1287/mnsc.2018.3253

Bijvank, M., & Vis, I. F. A. (2012). Inventory control for point-of-use locations in hospitals. *Journal of the Operational Research Society*, *63*(4), 497–510. https://doi.org/10.1057/jors.2011.52

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (2017). *Classification and regression trees*. Routledge.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794). ACM. https://doi.org/10.1145/2939672.2939785

Coelho, L. C., & Laporte, G. (2014). Optimal joint replenishment, delivery and inventory management policies for perishable products. *Computers & Operations Research*, *47*, 42–52. https://doi.org/10.1016/j.cor.2014.01.013

Darling, M., & Wise, S. (2010). Not your father's supply chain. following best practices to manage inventory can help you save big. *Materials Management in Health Care*, *19*, 30–33.

Dobson, G. a., Pinker, E. J. b., & Yildiz, O. a. (2017). An eoq model for perishable goods with age-dependent demand rate. *European Journal of Operational Research*, *257*(1), 84–88. https://doi.org/10.1016/j.ejor.2016.06.073

Dolan, E. D., & Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, *91*(2), 201–213. https://doi.org/10.1007/s101070100263

Federgruen, A., & Zipkin, P. (1984). An efficient algorithm for computing optimal (s, s) policies. *Operations Research*, *32*(6), 1268–1285. https://doi.org/10.1287/opre.32.6.1268

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/aos/1013203451

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Hyafil, L., & Rivest, R. (1976). Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, *5*(1), 15–17. https://doi.org/10.1016/0020-0190(76)90095-8

Landry, S., & Philippe, R. (2004). How logistics can service healthcare. *Supply Chain Forum: An International Journal*, *5*(2), 24–30. https://doi.org/10.1080/16258312.2004.11517130

Moons, K., Waeyenbergh, G., & Pintelon, L. (2019). Measuring the logistics performance of internal hospital supply chains – A literature study. *Omega*, *82*, 205–217. https://doi.org/10.1016/j.omega.2018.01.007

Nahmias, S. (2011). *Perishable inventory systems* (Vol. 160). Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *JMLR*, *12*, 2825–2830.

Ross, S. M. (1983). *Introduction to stochastic dynamic programming: probability and mathematical*. Academic Press, Inc.

Syntetos, A. A., Boylan, J., & Disney, S. M. (2009). Forecasting for inventory planning: A 50-year review. *Journal of the Operational Research Society*, *60*(sup1), S149–S160. May). https://doi.org/10.1057/jors.2008.173

Tsai, S. C., & Ho, I.-Y. (2019). Sample average approximation for a two-echelon inventory system with service-level constraints. *Journal of the Operational Research Society*, *70*(4), 675–688. https://doi.org/10.1080/01605682.2018.1457479

Volland, J., Fügener, A., Schoenfelder, J., & Brunner, J. O. (2017). Material logistics in hospitals: A literature review. *Omega*, *69*, 82–101. https://doi.org/10.1016/j.omega.2016.08.004

Appendix

In this appendix we report some detailed data on the results obtained with the ten most frequently used drugs in the ward used for our experiments.

In Tables A1–A10, for the out-of-sample tests, we report the total order quantities (Ord) and the number of emergency orders (E.O.) observed when applying each order policy. In the columns, under the caption Deterministic we report the theoretical lower bound obtained by running a Wagner–Whitin optimization model with perfectly known future demand. As expected, in this case there are no emergency orders. Columns associated to "Stochastic" report the results obtained by solving a standard stochastic optimization model based on the whole past history (equivalent to using the empirical probability distribution of future demands). This can be considered as the benchmark for comparison. The remaining columns report, respectively, the results obtained by using the *K*-nearest neighbors, Random Forest, Decision Trees, XGBoost approaches. Every row of the tables corresponds to a different planning horizon *h*. In each table the non-dominated solutions are emphasized.

Table A1. Amlodipina.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	514	0	*1856*	*0*	1906	2	1924	0	*1740*	*1*	1968	0
2	1032	0	3248	0	*2765*	*0*	3102	0	2906	1	2882	1
3	1549	0	3944	1	*3719*	*1*	4166	0	4011	0	*3934*	*0*

Table A2. Atorvastatina.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	3000	0	*5852*	*1*	*4438*	*14*	5973	2	*5412*	*6*	*5809*	*2*
2	6002	0	10219	4	*8531*	*18*	*9851*	*3*	*9147*	*11*	9987	3
3	9010	0	13537	18	*12623*	*17*	13566	8	*13234*	*8*	*13672*	*7*

Table A3. Bisoprololo.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	1406	0	*2923*	*0*	*2200*	*2*	2605	1	2264	5	*2558*	*1*
2	2818	0	*4254*	*5*	4321	7	*4581*	*4*	*4167*	*14*	*4905*	*2*
3	4210	0	*7130*	*0*	*6489*	*10*	7420	4	*6094*	*19*	7403	0

Table A4. Clopidogrel.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	307	0	1160	0	*966*	*1*	1160	0	*1098*	*0*	1128	0
2	613	0	1907	0	*1549*	*1*	1911	0	1889	0	*1823*	*0*
3	919	0	2606	0	*2183*	*0*	2617	0	2581	2	2466	1

Table A5. Enoxaparina.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	2266	0	6084	0	*4108*	*2*	*5016*	*1*	4396	4	*5282*	*0*
2	4562	0	10132	2	*7991*	*4*	9370	1	*8438*	*1*	10033	1
3	6878	0	13659	3	*11498*	*8*	*12858*	*1*	*12556*	*2*	13353	1

Table A6. Fondaparinux.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	1683	0	4712	0	*3799*	*3*	4803	0	5048	1	*4522*	*0*
2	3364	0	9804	0	*7179*	*2*	*8586*	*0*	*7761*	*1*	8648	0
3	5012	0	11452	0	*10069*	*2*	*11022*	*0*	*10631*	*1*	11116	0

Table A7. Lansoprazolo.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	465	0	1130	0	*859*	*3*	1128	0	1095	1	*1095*	*0*
2	932	0	2034	0	*1615*	*4*	2032	0	1996	0	*1974*	*0*
3	1401	0	*2782*	*1*	*2452*	*9*	2842	1	2843	1	*2806*	*0*

Table A8. Pantoprazolo.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	1488	0	*3042*	*2*	*2207*	*9*	*2787*	*3*	*2377*	*6*	2850	3
2	2979	0	4373	13	*4369*	*9*	*4019*	*13*	4019	20	4157	13
3	4477	0	7020	7	*6340*	*16*	*6510*	*11*	*5847*	*20*	*6836*	*7*

Table A9. Ramipril.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	1358	0	2990	2	*2244*	*12*	2990	2	*2671*	*2*	2894	3
2	2694	0	5214	4	*4107*	*17*	*5041*	*3*	*4714*	*8*	*5074*	*2*
3	4040	0	6905	8	*6039*	*10*	6726	6	6733	8	*6675*	*6*

Table A10. Ticagrelor.

	Deterministic		Stochastic		*K*NN		RF		DT		XGBoost	
h	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.	Ord	E.O.
1	267	0	963	0	*585*	*3*	964	0	854	0	*763*	*0*
2	532	0	1596	0	*1223*	*1*	1534	0	*1530*	*0*	1544	0
3	796	0	2280	0	*1800*	*3*	2127	0	*1983*	*1*	*2067*	*0*