CASE REPORT

Predicting Packaging Sizes Using Machine Learning

Michael Heininger1 · Ronald Ortner2

Received: 17 February 2022 / Accepted: 19 July 2022 © The Author(s) 2022

Abstract
The increasing rate of e-commerce orders necessitates a faster packaging process, challenging warehouse employees to correctly choose the size of the package needed to pack each order. To speed up the packing process in the Austrian e-commerce company niceshops GmbH, we propose a machine learning approach that uses historical data from past deliveries to predict suitable package sizes for new orders. Although for most products no information regarding the volume is available, using an approximate volume computed from the chosen packages of previous orders can be shown to significantly increase the performance of a random forest algorithm. The respective learned model has been implemented into the e-commerce company’s software to make it easier for human employees to choose the correct packaging size, making it quicker and easier to fulfill orders.

Keywords Packaging · Machine learning

1 Introduction
Packaging is an integral part of the delivery process. With orders consisting of various items of different sizes, choosing a suitable packet among several available sizes is an ongoing challenge for the operators. Choosing a packet that is too small implicates time consuming repackaging while packets that are too large lead to a larger volume and hence higher transportation costs.

1 In general, an order may be split up into several packets for several reasons like different delivery times or diverging sizes of the ordered items. However, in the following we keep using the term order for items put into one packet for delivery.

* Ronald Ortner
rortner@unileoben.ac.at

Michael Heininger
michael.heininger@niceshops.com

1 niceshops GmbH, Steiermark, Austria
2 Lehrstuhl für Informationstechnologie, Montanuniversität Leoben, Leoben, Austria

Published online: 22 August 2022 © Springer
In practice, the choice of the packaging size is often left to the warehouse operator who selects the appropriate packaging size based on experience. This paper presents an approach that based on machine learning techniques applied to historical data of past deliveries is able to predict a suitable packaging size for each order. With a reliable recommendation for a suitable packet available in advance, the operator can speed up the packaging process. Moreover, with predictions of the required packaging sizes available the transport volume can be computed in advance, which can be forwarded to the delivery service for further optimization in the next step of the delivery process.

Overview The following Sect. 2 gives a detailed problem description. Sect. 3 then presents our machine learning approach. The subsequent Sects. 4 and 5 describe the experiments that were conducted and the respective results we could achieve. The final two sections discuss the implementation process as well as advantages over a classic Operations Research approach.

2 Problem Description

In the following we give a brief description of the starting point of the project that was developed and implemented by the Austrian e-commerce company niceshops GmbH. Niceshops develops online shops in various niche markets and is operating in several European countries. The company has a strong focus on logistics and information technology. The software for the online shops, processing master data and orders as well as managing the warehouse is developed in-house. The packaging process in the warehouse is also supported by this software.

2.1 Current Packaging Process

In the company a two-stage picking process is implemented, in which the gathering of the products and the assembly for a specific customer order are separated. In the first stage, several customer orders are grouped into a so-called picklist at product level. This picklist contains the products with the total number required for customer orders and defines the sequence in which the products are to be accessed. In the next step, the picked products are brought to the packing table where the operator starts the packing process.

Usually, an operator first piles the items of an order on this packing table to be able to estimate the size of a suitable packet. In the next step, a corresponding packet is taken from the stack of packaging (placed above the packing table) and the products are put into the packet together with the invoice. The whole process is accompanied by the company’s software where the operator also has to specify which packaging has been used for the respective order.

In the final step, the package is sealed, assigned a package label and taken to the outbound delivery. From there, the packets leave the warehouse in the vehicle of the delivery service.
2.2 Possible Approaches for Obtaining Packaging Size Recommendations

The goal of the project was to obtain reliable recommendations for the packaging size of each order based on the items to be packed and to check whether the prediction accuracy is sufficient for the use in the productive system.

The problem of finding a suitable packet for a given set of items in principle could be handled as a geometric bin packing problem [4, 5]. Solving such problems is usually computationally hard, which sometimes can be overcome by heuristic approaches [6]. However, often there are further constraints that need to be taken into account (like certain properties of the items to be packed like fragility) which are not easy to formalize. Most importantly, for items sold by niceshops the necessary data concerning the item dimensions are usually not available making such an approach not feasible.

Instead, it seems more promising to try to predict packaging sizes using machine learning algorithms, as plenty of historical data of past orders is available not only to train but also to validate models. We note that such an approach is suitable to pursue the mentioned goal of saving packaging time by suggesting packaging sizes to the human operators. However, machine learning on past data will not be able to improve, e.g., fill rates of packets, for which a completely different approach would be necessary, cf. also the discussion in Sect. 7.

The following sections describe in more detail, which data is available, which features are selected for learning, which machine learning algorithms are used, and what results are achieved in the end.

In general, there is little literature available documenting the use of machine learning to predict packaging sizes. The only relevant reference we are aware of is [2], which presents a case study for the prediction of packaging and fill rate of individual parts in automotive industry based on historical data. Note that packaging of single parts is an easier problem, as feature selection and learning approach can be chosen in a much more straightforward way. For the packaging of several items one either has to use features aggregated from the items to be packed or consider a kind of multiple-instance learning setting, cf. Sect. 3.2

3 Packaging Size Prediction by Machine Learning

3.1 Data Analysis

The historical data available for all past deliveries includes the information which items were put into which packet. In the following we focus on four different shops out of more than 20 that in the course of the project turned out to be representative. For these four selected shops historical data of a total of 896,598 orders is available. 673,485 of these orders are from the shop Ecco Verde, 145,317 from VitalAbo, 58,713 from Piccantino and 19,083 from 9Weine.

According to the available data, Ecco Verde is the biggest shop selling natural cosmetics products. VitalAbo is a webshop for nutritional supplements and 9Weine sells wine. These three shops, especially 9Weine, have a pretty narrow product
range. On the other hand, the webshop *Piccantino* is selling a wide range of different products consisting of spices, oils, sauces, and fresh products like meat.

In total there are up to about 40 different packaging sizes, so that the problem at hand is a multi-class learning problem with quite a few different labels to choose from. Closer examination of the available data however shows that across the selected shops a small number of the most frequently used packaging sizes are able to cover a majority of all orders, see Fig. 1. While for *Ecco Verde*, *VitalAbo*, and *9Weine* the three most frequent packaging sizes cover more than 75% of all orders, for *Piccantino* it is around a half.

Accordingly, we use the (three) most frequent packaging size(s) as a baseline which the goal was to improve upon.

### 3.2 Learning Setting

Formally, denoting the set of all items by $I$, each available training example is of the form $(X, \ell)$, where $X$ is the set of items contained in one order with $X \subseteq I$ and the label (or class) $\ell$ is the packaging size that was used for packing the items in $X$. With the instances $X$ being sets, this formally constitutes a multiple-instance learning setting [1], where each order corresponds to what is usually called a *bag*.

However, in classical multiple-instance classification problems it is usually assumed that the label of the bag is determined in a certain way by the labels of the instances contained in the bag. Thus, the *standard assumption* in the setting with binary labels is that a bag is labeled negative if all its instances have a negative label. Accordingly, a single positive instance makes the label of the bag positive. In our case the packaging size of an order will not be determined by the packaging size of a single item contained in the order. Moreover, in the multi-class case (i.e., having more than two labels) the typical setting considered in multiple-instance learning is to assign in general more than one label to each bag [3], which in our setting does not make sense.

---

Fig. 1 Percentage of packaging sizes per shop, three most common sizes
As the structure of our problem is different from classical multiple-instance problems, an application of respective learning approaches available in the literature (cf. [1] for an overview) seems not promising. Moreover, covering the power set \(2^I\) (corresponding to all possible orders) in general would require many more training examples than when working with a suitable set of features that give an aggregated description of the items in each order, which is the approach we take in the following.

### 3.3 Feature Selection and Engineering

As already mentioned, even basic features such as the item dimensions are not available for all items. The only relevant property that is known of all items is the *item weight*. As we do not use a multiple-instance approach, the weight of all items in the order has been summed up to obtain a total weight of the order. The other aggregated feature is the *number of items* in the order. Further, some shops provide fresh food that is usually put in special packets, so that a *fresh food* flag has been added to the features. Another flag is used for items that are picked up by the customers themselves and are usually not put in a packet. In the software, there is an explicit option of choosing *no packet* in this case.

**Approximate Item Volume** As the dimensions of items are not generally available, it is not possible to compute a volume of the items to be packed. However, we try to estimate an *approximate volume* for each item from previous deliveries, using the fact that the weight of each item as well as the volume of each chosen packet are known.

Given the data \((X_n, \mathcal{C}_n)_{n \geq 0}\) of past deliveries with items \(X_n = \{i_{k_1}, \ldots, i_{k_n}\} \subset I\) packed in packet \(\mathcal{C}_n\), we use the volume \(\text{vol}(\mathcal{C}_n)\) of the chosen packet \(\mathcal{C}_n\) as well as the weights \(w(i)\) of the items \(i \in X_n\) to first compute an approximate volume \(\text{vol}_n(i)\) of each item \(i \in X_n\) of delivery \(n\) by

\[
\text{vol}_n(i) := \text{vol}(\mathcal{C}_n) \cdot \frac{w(i)}{\sum w(i_k)},
\]

simply assuming that each item takes up a volume proportional to its weight. In a second step we define the approximate volume for an item \(i \in I\) to be the median value of the approximate volumes \(\text{vol}_n(i)\) of all orders \(n\) that contain item \(i\), that is,

\[
\text{vol}(i) := \text{median}\{\text{vol}(\mathcal{C}_n) \mid i \in X_n\}.
\]

With an approximate volume available for each item\(^3\) for a new order with items \(X \subset I\) the approximate volume is then defined as

\(^2\) These orders could also be treated separately. However, conceptually it was easier to let all orders be dealt with by the learned machine learning model.

\(^3\) An item that has not been ordered before is assigned approximate volume 0. More sophisticated values could be used instead. However, this simple approach works reasonably well.
We will see that while the approximate item volume itself is not sufficient to obtain a satisfactory recommendation of packaging size, the use of it as additional feature is able to significantly improve the prediction of the random forest algorithm (cf. Sect. 4).

To sum up, the following features of each order are used:

- number of items to be packed
- total weight of items to be packed
- approximate volume
- fresh food flag
- self-service flag

That way, each original training example \((X, \ell)\) with \(X \subset I\) and label \(\ell\) is transformed into \((f(X), \ell)\) where \(f(X)\) denotes the 5-dimensional feature vector computed from the items in \(X\). The transformed samples are then fed to various machine learning algorithms as described in the following section.

4 Experiments

In this section we report experiments for the four shops we have selected in Sect. 3. Experiments were also conducted for other shops with similar results.

4.1 Setup

The model development is carried out in Python using a Google Colab IPython notebook, where for each of the selected shops an individual model is learned. We use the open source library Scikit-learn for feature engineering as well as for training and evaluating these shop-specific models. Scikit-learn offers state-of-the-art implementations of most machine learning algorithms so that different models can be learned and evaluated relatively quickly and easily. While we also did a few experiments with other algorithms, here we report the experimental setup and the results for the following algorithms for which we briefly describe the experimental setup and parameter setting that was used. We also point out how each of the used algorithms can determine probabilities for predicted labels that are used in order to determine the three best labels in the experiments.

**k-Nearest Neighbor (kNN)** kNN is the simplest algorithm that we used. Since kNN for each new instance \(X\) simply predicts a majority vote of the \(k\) nearest neighbors to \(X\) among the available labeled data, there is actually no training to be done. For the parameter setting, we have to choose the value of \(k\) as well as a proper scaling for the numerical features that we used. The scaling is carried out with the help of the
MinMaxScaler from Scikit-learn, while for the choice of $k$ we test values between 5 and 1000 (see Sect. 4). The implementation of the algorithm in Scikit-learn also enables the output of the label probabilities using the proportion of a label among the labels of the $k$ nearest neighbors.

**Neural Network (NN)** For learning a multi-layered perceptron classifier Scikit-learn offers an implementation of the backpropagation algorithm, which however does not offer GPU support (cf. Sect. 4.2), so that we only use a simple model with three intermediate layers of ten neurons each and rectified linear activation function (ReLU). In order to determine a label probability in our multi-class classification problem, an output neuron and a so-called softmax activation function are required in the output layer for each label. The softmax activation function ensures that all output probabilities of the neurons are between 0 and 1 and add up to 1.

**Random Forest (RF)** For the random forest classifier that combines suitable decision trees, we use a grid search with the help of Scikit-learn’s GridSearchCV in order to set the parameters of the algorithm. Also for random forest one can obtain label probabilities by computing the mean of the probabilities of the labels for each tree in the forest. The probability of each tree is calculated by the fraction of the instances with the same label in a leaf.

### 4.2 Hardware and Computation Time

We used hardware resources from Google such as GPUs, TPUs, and 35 GB RAM for training the models. While the computations in principle could be done on more modest hardware, the used resources allow the models to be trained quickly. The training of a model from the biggest shop Ecco Verde only takes about three minutes for the random forest algorithm and over an hour for a very simple neural network.

### 5 Results

In this practical use of the prediction results, also in accordance with the used baseline, not only the best predicted packaging size but also the three best predicted packaging sizes were considered. Therefore the classifiers were set up to determine both the corresponding class and the probability with which an instance belongs to a particular class. The classes corresponding to the three highest class probabilities then were chosen as the three best labels. When reporting accuracy with respect to these three best predicted packaging sizes, the prediction is considered to be correct if the packaging size actually chosen for the respective order (i.e., the true label) is among these three.

The following tables show the prediction accuracy determined by performing a 5-fold cross validation on the available data, where all algorithms use the same folds. In general, we observed that the results hardly deviated from each other over different folds. The percentages in the following tables are the average values of the five folds.
5.1 Results without Using the Approximate Volume

Tables 1 and 2 show the results that we got without using the approximate volume. The prediction of the single best packaging size is most accurate using the simple kNN algorithm. However, when considering the best three packaging sizes the other two algorithms turn out to be superior. For kNN we also tested various different values for $k$. While the prediction accuracy of the best label does not change much with increasing $k$, when considering the three best labels accuracy is increasing with $k$. Only for $k$ approaching $k = 1000$, accuracy starts to decrease slightly. Of course, this improvement comes at the cost of increased time needed for label computation. Tables 1 and 2 show the prediction accuracy of kNN for $k = 50$, which provides a good payoff. The difference to the best achieved accuracy of kNN in each shop is below 1%. In any case, all algorithms already provide a significant improvement over the baseline algorithm.

5.2 Improvements Using the Approximate Volume

In a second step we tried to improve prediction accuracy by also using the approximate volume as a feature.

5.2.1 Using the Approximate Volume for Prediction

An obvious idea would be to predict the packaging size using only the approximate volume. However, as Table 3 shows this gives worse accuracy than the baseline. For the predictions used in Table 3 we also computed an approximate volume for each packaging size. That is, for each order with a given packaging size $P$ we consider the approximate volume as given in Eq. (3). Then the approximate volume for packaging size $P$ is defined as the median value of all orders for which packaging $P$

| Shop     | Baseline | kNN     | NN      | RF     |
|----------|----------|---------|---------|--------|
| 9Weine   | 35.38%   | 92.09%  | 92.30%  | 93.16% |
| Ecco Verde | 32.31%   | **63.00%** | 61.92%  | 61.22% |
| Piccantino | 17.15%   | 44.19%  | 42.24%  | **45.87%** |
| VitalAbo | 49.40%   | **73.16%** | 70.01%  | 72.94% |

Table 1: Accuracy of best predicted packaging size (without using approximate volume)

| Shop     | Baseline | kNN     | NN      | RF     |
|----------|----------|---------|---------|--------|
| 9Weine   | 80.95%   | 98.40%  | **99.06%** | 98.59% |
| Ecco Verde | 74.25%   | 92.57%  | **96.31%** | 95.76% |
| Piccantino | 47.19%   | **81.50%** | 80.09%  | 81.21% |
| VitalAbo | 83.33%   | 91.94%  | 96.16%  | **96.54%** |

Table 2: Accuracy of three best predicted packaging sizes (without using approximate volume)

Bold values indicate the highest achieved accuracy of the respective shop.
has been used. That is, denoting the sequence of the past orders used for training by 
\((X_n, \ell_n)_{n \geq 0}\) with \(X_n \subseteq I\) being the items ordered and \(\ell_n\) the used packaging, we set

\[
\text{vol}(P) := \text{median}\{\text{vol}(X_n) \mid \ell_n = P\}. \quad (4)
\]

For a new order \(X\), the predicted label \(\ell\) used in Table 3 is then the packaging size whose approximate volume is closest to the approximate volume of the order, that is,

\[
\ell := \arg \min_P \mid \text{vol}(X) - \text{vol}(P) \mid. \quad (5)
\]

We also tried more straightforward predictions using the true instead of the approximate volume. However, predictions turned out to be even less accurate.

### 5.2.2 Using the Approximate Volume as Additional Feature

Using the approximate volume as additional feature for the random forest algorithm boosts the prediction performance as the results in Table 4 show. For most shops the achieved accuracy values are the best achieved over all tested algorithms. Interestingly, unlike for the random forest algorithm using the approximate volume as additional feature for kNN and the neural network does not give improved performance. Quite to the contrary, in some cases the accuracy decreases significantly for both algorithms. Possibly, the use of dependent features confuses both kNN and the neural network.

To give a more detailed picture of the individual predictions of the random forest algorithm, we present in Figs. 2 and 3 the confusion matrices for the most common package sizes of the shops Ecco Verde and Piccantino, respectively. For Ecco Verde these make up 99.5% of all orders, while for Piccantino it is 91.59%. The package sizes shown in the confusion matrix for Ecco Verde are ordered by volume.

| Shop       | AV1  | AV3  |
|------------|------|------|
| 9Weine     | 47.58% | 72.80% |
| Ecco Verde | 10.08% | 44.15% |
| Piccantino | 16.22% | 41.44% |
| VitalAbo   | 24.52% | 61.64% |

### Table 3 Accuracy of best (AV1) and three best (AV3) predicted packaging size(s) using only the approximate volume for prediction

| Shop       | AV1  | AV3  |
|------------|------|------|
| 9Weine     | +0.41% | +0.39% |
| Ecco Verde | +7.75% | +2.35% |
| Piccantino | +14.16% | +10.53% |
| VitalAbo   | +7.48% | +2.05% |

### Table 4 Improvement in accuracy when adding approximate volume as feature for the random forest algorithm. The accuracy is shown for best (RF-AV1) and three best (RF-AV3) predicted packaging sizes. Bold values show best achieved performance over all tested algorithms and setups
and it can be seen that most errors happen close to the diagonal. That is, if a wrong packaging size is predicted, the volume is still close to the one actually chosen for the respective order. The same pattern can be seen for Piccantino. Here the packaging sizes shown in the confusion matrix are first grouped into three different categories (packaging for bottles, packaging for food, and general packaging) and

![Confusion matrix for first fold of Ecco Verde for most common packaging sizes ordered by volume. Matrix entries give the number of orders with true label (picked packaging) shown in the respective column and in the respective row the label predicted by the model learned by the random forest algorithm (using the approximate volume) on the data of the other four folds.](image1)

![Confusion matrix for first fold of Piccantino for most common packaging sizes (grouped by category and ordered by volume within category) when classified by the model learned by the random forest algorithm (using approximate volume) on the other four folds.](image2)
then again ordered by volume within each category. As before, most errors happen within the same packaging category along the diagonal. The only exception is that the bottle packages are often confused with general packages of sizes h–k (and vice versa), which have a similar volume. Such errors could be easily avoided by adding more information such as a bottle flag to the order data, which at the moment is not available.

5.2.3 Learning Across Shops

It is an interesting question whether models learned on one shop could be used for classification in another shop. Such models could be, e.g., employed for starting up a new shop when no data is available yet. We tested the two different shop models of Ecco Verde and Piccantino in order to classify the orders of VitalAbo. As the portfolio of Ecco Verde and VitalAbo is similar with respect to the dimensions of the available items, we expected the model learned for Ecco Verde to work better than that of Piccantino, which has a different range of products and less data available.

We performed experiments for the random forest classifiers learned with and without approximated volume. Table 5 shows that the accuracy for the Piccantino model without using the approximated volume is a bit above our baseline, while the respective Ecco Verde model is significantly better, although not nearly as good as the numbers for cross validation on the orders from VitalAbo itself, cf. Tables 1 and 2.

Surprisingly, the Ecco Verde model that uses approximate volume gives lower accuracy than the respective model learned without approximate volume. Obviously, here the volume information learned on Ecco Verde is misleading for the orders of VitalAbo, maybe due to different weight distributions of common items. Unlike that, for the Piccantino model there is a boost in accuracy when adding approximate volume resulting in numbers that are even better than for the respective Ecco Verde model. For the three best models prediction using the approximate volume also the accuracy of the Ecco Verde model improves at least a bit.

To sum up, the results for learning across shops are not as convincing as for models that are learned from orders of the same shop. However, at least with respect to the best three predicted sizes one obtains a clear advantage over the baseline. Moreover, for both learned models the accuracy is large enough so that they could be used to preselect three package sizes as suggestion for the human operator, cf. next section.

| Model   | RF1   | RF-AV1 | RF3   | RF-AV3 |
|---------|-------|--------|-------|--------|
| Ecco Verde | 61.82% | 54.76% | 91.65% | 92.72% |
| Piccantino | 51.03% | 59.44% | 86.88% | 94.45% |

Table 5 Accuracy for orders of VitalAbo when classified by the random forest models learned from Ecco Verde and Piccantino, respectively. Shown are numbers for models learned with approximated volume (RF-AV) and without (RF), which we report for best (RF1 and RF-AV1) as well as best three (RF3 and RF-AV3) predicted sizes.
6 Implementation

Since the random forest algorithm has the highest forecast accuracy for almost all shops, it was decided to integrate a model of this algorithm into the production system. Currently, a model of the random forest algorithm is in use in the shop VitalAbo to suggest the size of the packaging of each order. The trained models are hosted on the Google Cloud AI Platform. The prediction of the packaging sizes is done for a large number of deliveries at once shortly after these have been created. A request with the values of the previously defined features of these deliveries is sent to the model in the cloud. The model processes this request and returns the three most likely packaging sizes for each delivery. These predicted packaging sizes are stored in the database for the respective deliveries. When the packaging process starts, the suggested packaging sizes for the delivery to be packed are loaded from the database and proposed to the warehouse operator.

Due to the fact that the prediction is carried out in advance and not directly at the start of the packaging process, it is also possible to use models for which the classification takes a little longer. In general, depending on the accuracy of the forecast, it can be defined separately for each shop whether the most likely packaging size is preselected in the packaging process or whether only the three most likely sizes are suggested.

7 Conclusion

We have shown that packaging sizes can be predicted with surprisingly high accuracy training off-the-shelf machine learning algorithms on historical data, even when using a very small number of aggregated features and without having access to proper volume information. While even a very simple algorithm like kNN provides good results, using an approximation of the volume boosted the performance of the random forest algorithm, providing a sufficiently high prediction accuracy for employing the prediction in practice to speed up the packaging in the shop VitalAbo.

The possibility of using the forecasting systems is not limited to the packaging process. As already mentioned, the total transport volume per delivery service can be estimated using the packaging size predictions. In this way, it can be determined how many vehicles are required from each delivery service, which in future could be further used for optimizing the transports. Another possible utilization of the packaging size predictions is an optimization with respect to the packaging inventory. As the models are now being successively integrated into the packaging process for all shops of the company, these and other possible uses of the forecast models are being examined. In the course of this process, it can be considered to use shop-specific features in order to increase the predictive accuracy of the respective models.
Our approach differs from a classical Operations Research perspective, which would consider the packaging problem rather within an optimization setting (e.g., trying to maximize the fill rates of packets) for whose solution however more data would be needed. One could argue that collecting the missing data and employing suitable packing algorithms in the end would give additional benefit by providing the user with a single choice of packaging size that in addition has (near-)optimal fill rate.

Indeed, the collection of the product dimensions is useful for a number of reasons like determining the warehouse utilization or in order to comply with delivery service restrictions. However, using it as the sole measure for determining the packaging size might be insufficient due to a number of reasons:

- First of all, data quality is crucial but usually quite costly to achieve. This holds even more considering the turnover of articles, which are sometimes available only for a few months so that there is not much time neither for data acquisition nor correction.
- Second, for such an approach knowing just the volume will be not enough. Rather it has to be contemplated which data of the items to be packed (which are often not simple rectangular objects) shall be acquired with which accuracy. Products like bottles or fresh food have characteristics apart from dimensions that result in different packaging. Thus, fragile items need stuffing resulting in a larger packaging size. Further, the amount of additional stuffing might correlate with the total number of (fragile) items or the weight of the package. On the other hand, stackable products or items like clothes that can be folded may result in a smaller packaging.
- Last but not least, a near-optimal packing algorithm in general will give solutions that may be difficult to realize for a human operator. Thus, in general there is a payoff between packaging speed of a human operator and the fill rate of the respective packet.

Accordingly, we think that for achieving our main goal of speeding up the packaging process our approach of getting a solution that is good enough and simple to implement given the available data is a good compromise to start with. In practice, this gives some gain already after a short implementation phase and with no additional investments necessary. Further, possible refinements and improvements can be easier assessed given some evaluation of the initial solution.

Acknowledgements The authors would like to thank the reviewer for valuable comments on the original submission, which greatly helped to improve the paper with respect to content and positioning.

Funding Open access funding provided by Montanuniversität Leoben.

Data Availability In order to protect commercial interests of niceshops GmbH as well as the privacy of its customers the datasets analyzed for this study cannot be made available.
Declarations

Conflict of Interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Carbonneau MA, Cheplygina V, Granger E, Gagnon G (2018) Multiple instance learning: A survey of problem characteristics and applications. Pattern Recogn 77:329–353. https://doi.org/10.1016/j.patcog.2017.10.009
2. Knoll D, Neumeier D, Prüglmeier M, Reinhart G (2019) An automated packaging planning approach using machine learning. In 52nd CIRP Conference on Manufacturing Systems (CMS) 81:576–581. https://doi.org/10.1016/j.procir.2019.03.158
3. Zhou ZH, Zhang ML, Huang SJ, Li YF (2012) Multi-instance multi-label learning. Artif Intell 176(1):2291–2320. https://doi.org/10.1016/j.artint.2011.10.002
4. Martello S, Pisinger D, Vigo D (1997) The Three-Dimensional Bin Packing Problem. Oper Res 48(2):256–267. https://doi.org/10.1287/opre.48.2.256.12386
5. Wäschcher G, Hausner H, Schumann H (2007) An improved typology of cutting and packing problems. Eur J Oper Res 183(3):1109–1130. https://doi.org/10.1016/j.ejor.2005.12.047
6. Lodi A, Martello S, Vigo D (2002) Heuristic algorithms for the three-dimensional bin packing problem. Eur J Oper Res 141(2):410–420. https://doi.org/10.1016/S0377-2217(02)00134-0

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.