Transportation Research Part B 149 (2021) 52–78

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

Shipping Domain Knowledge Informed Prediction and Optimization in Port State Control

Ran Yan[a], Shuaian Wang[a,*], Jiannong Cao[b], Defeng Sun[c]

[a] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[b] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[c] Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 12 January 2021
Revised 24 March 2021
Accepted 2 May 2021

Keywords:
Domain knowledge informed machine learning
Domain knowledge informed artificial intelligence
Optimization in port state control (PSC)
PSCO scheduling model
Inspection template

ABSTRACT

Maritime transportation is the backbone of global supply chain. To improve maritime safety, protect the marine environment, and set out seafarers' rights, port state control (PSC) empowers ports to inspect foreign visiting ships to verify them comply with various international conventions. One critical issue faced by the port states is how to optimally allocate the limited inspection resources for inspecting the visiting ships. To address this issue, this study first develops a state-of-the-art XGBoost model to accurately predict ship deficiency number considering ship generic factors, dynamic factors, and inspection historical factors. Particularly, the XGBoost model takes shipping domain knowledge regarding ship flag, recognized organization, and company performance into account to improve model performance and prediction fairness (e.g., for two ships that are different only in their flag performances, the one with a better flag performance should be predicted to have a better condition than the other). Based on the predictions, a PSC officer (PSCO) scheduling model is proposed to help the maritime authorities optimally allocate inspection resources. Considering that a PSCO can inspect at most four ships in a day, we further propose and incorporate the concepts of *inspection template* and *un-dominated inspection template* in the optimization models to reduce problem size as well as improve computation efficiency and model flexibility. Numerical experiments show that the proposed PSCO scheduling model with the predictions of XGBoost as the input is more than 20% better than the current inspection scheme at ports regarding the number of deficiencies detected. In addition, the gap between the proposed model and the model under perfect-forecast policy is only about 8% regarding the number of deficiencies detected. Extensive sensitivity experiments show that the proposed PSCO scheduling model has stable performance and is always better than the current model adopted at ports.

1. Introduction

Maritime transport is the backbone linking global supply chains, supporting trade and enabling participation in global value chains (Chang et al., 2020; Zhen et al., 2020; Wu et al., 2021). Maritime safety is one of the most essential prerequisites

* Corresponding author.
E-mail addresses: angel-ran.yan@connect.polyu.hk (R. Yan), wangshuaian@gmail.com (S. Wang), csjcao@comp.polyu.edu.hk (J. Cao), defeng.sun@polyu.edu.hk (D. Sun).

https://doi.org/10.1016/j.trb.2021.05.003
0191-2615/

for running a successful business. It is a broad term ranging from ship construction to operation and management, and to how professional the crews are. In recent years, shipping pollution has been receiving wide attention as the greenhouse gas and pollutants produced by ships significantly contribute to global climate change and acidification (Christodoulou et al., 2019; Bell et al., 2020). To enhance maritime safety and protect the marine environment, global and regional conventions are implemented by the International Maritime Organization (IMO) and local governments (Zhang et al., 2020).

Generally, a ship is regarded as substandard if its hull, machinery, equipment, or operational safety is substantially below the standards required by the relevant conventions or if the crew is not in conformance with the safe manning document (IMO, 2017). Effective identification and rectification of substandard ships is essential, as it guarantees efficient implementation of various conventions. As a complement of flag state control, which is the first line of defense against substandard shipping, port state control (PSC) renders port authorities the right to inspect foreign visiting ships. During PSC inspections, a condition found not to be in compliance with the requirements of the relevant convention is called a ship deficiency. If fatal deficiencies are found onboard, an intervention action, which is also called detention, can be taken by the port state (IMO, 2017). Ship deficiency and detention are seemed as the most crucial outcomes of PSC inspection (Akpinar and Sahin, 2020).

The general process of PSC inspections is as follows. In the morning of a working day, the port state authority selects the foreign visiting ships with high risk regarding ship safety and marine pollution for inspection based on the selection criteria adopted in the region. Then, available qualified inspectors, who are also called PSC officers (PSCOs), are assigned and scheduled to inspect the selected high risk ships. It is generally believed that accurate identification of high-risk visiting ships is a pre-requirement while effective assignment and scheduling of available PSCOs is a foundation for effective PSC inspections. The reasons are as follows. First, it is not possible to inspect all visiting ships as port inspection resources, especially the number of available PSCOs, are quite limited. Second, among all visiting ships, only a small portion of ships need to be inspected. The annual report of Tokyo Memorandum of Understanding (MoU) in Asia-Pacific region shows that only 60% of the inspections conducted between 2009 and 2019 identified deficiencies, and no more than 6% inspections were with detention (Tokyo MoU, 2020). Third, the proportion of ships inspected is crucial in port management. If too few substandard ships are inspected at a port, ship owners may lack the motivation to intensively maintain ship conditions, which in return attracts more substandard ships to the port. On the contrary, if too many qualified ships are inspected, the competitiveness of the port may be reduced and consequently leads ship owners to turn to other destinations with relaxed inspection policy (Yang et al. 2018b, Yan et al., 2021c). Therefore, accurate identification of high-risk ships and rational allocation of inspection resources guarantee effective PSC inspections by picking out which ships are most worthy of inspection and finishing the inspection tasks efficiently without putting too much delay in shipment. They also help the port states to find a balance between stringent inspections of substandard ships and reducing un-necessary inspections of qualified ships and thus to better fulfill their responsibilities and enhance their competitiveness.

One of the widely adopted ship selection scheme at the port states is the new inspection regime (NIR). It calculates ship risk profile (SRP) based on ship generic parameters including ship type, ship age, flag performance, recognized organization (RO) performance, and company performance, and inspection historical factors including deficiency and detention conditions (Paris MoU, 2010; Tokyo MoU, 2014). It is noted that all the parameters are objective except for flag, RO, and company performance, which is calculated by the MoUs. More specifically, ship flag performance is established annually by taking its ships' inspection and detention conditions over the preceding three calendar years into account. Black-grey-white ship flag lists are published in an MoU's annual report, where flag performance gets worse from white to grey and to black. RO is a qualified organization which has been assessed and authorized by the flag state to provide necessary statutory services and certification of ships entitled to fly its flag (IMO 2017). The performance of all ROs is established annually considering their ships' inspection and detention history over the preceding three calendar years. The RO performance list is published in an MoU's annual report, where the performance of ROs gets worse from high, to medium, to low, and to very low. Ship company is the International Safety Management (ISM) company of a ship, and its performance is determined by their ships' detention and deficiency history calculated daily on the basis of a running 36-month period. Similar to ship RO performance, company performance gets worse from high, to medium, to low, and to very low.

As ship flag, RO, and company play an important role in ship management, operation, and maintenance, they are taken into account in the popular SRP ship selection scheme applied at ports. In return, a ship's performance in PSC inspection can influence the reputation of its flag, RO, and company and their performance evaluated by PSC MoUs. Under this condition, it is justifiable to conclude that given all other conditions being equal, a ship should be estimated to have worse performance in PSC inspection (e.g. more deficiencies and higher probability of detention) if the performance of its flag/RO/company gets worse. However, such domain knowledge is seldom considered in current literature of high-risk ship selection mainly because combining domain knowledge with machine learning models is not a trivial task as it requires modifications of the prediction models or finding good properties of them. Besides, PSCO assignment and scheduling models, which require allocating the available and scarce inspection resources as well as arranging the starting and ending time of the required activities, are also rarely proposed in current research. This study aims to bridge this gap with the contributions summarized as follows.

First, from a theoretical point of view, we first develop a machine learning prediction model considering proper and adequate domain knowledge to solve problems in maritime transportation. Specifically, a state-of-the-art tree-based model called XGBoost is developed to predict ship deficiency number in PSC inspection. In the XGBoost model, we combine the shipping domain knowledge regarding ship flag, RO, and company performance in a natural way. Based on the predictions, a

PSCO scheduling model for ship inspection is then proposed considering a PSCO's work and rest time to guarantee inspection effectiveness. By taking the properties of the optimization model for PSCO scheduling into account, we propose the concepts of *inspection template, un-dominated inspection template*, and strengthened constraints to reduce problem size as well as improve model flexibility and solving efficiency.

Second, from a practical point of view, a practical problem in PSC inspection, which is one of the most important shipping policies, is addressed in this study. Numerical experiments show that the proposed combined model for ship deficiency number prediction and PSCO scheduling is more than 20% better than the current PSCO scheduling strategy at ports regarding the number of deficiencies identified. Meanwhile, the gap between the proposed model and the perfect-forecast policy is only about 8% regarding the number of deficiencies identified. The proposed model can help port state authorities to identify higher risk ships and schedule inspection resources more efficiently. Especially, it contributes to assisting the port states to achieve a balance between effectively identifying and inspecting substandard ships and reducing un-necessary inspections of qualified ships and consequently frightening them from choosing this port in future shipment. Therefore, the main objectives of PSC to eliminate substandard shipping, to promote maritime safety and security, to protect the marine environment, and to safeguard seafarers' working and living conditions on board ships can be enhanced.

2. Literature review

As PSC is critical for promoting maritime safety, guaranteeing the marine environment, and protecting seafarers' rights, there has been a large body of literature on PSC inspection. Yan and Wang (2019) classified the studies on PSC into four main categories, namely studies on exploring factors influencing PSC inspection results, studies on developing ship selection models, studies on exploring the effects of PSC, and studies on proposing suggestions for MoU management. In this study, we focus on the studies on improving PSC efficiency, which develop models for ship selection and onboard inspection efficiency improvement.

Both abstract ship risk level and concrete ship detention or deficiency condition have been used as the prediction targets in models for high-risk ship identification. For prediction models with abstract targets, Li (1999) proposed the concept of ship risk score and identified ship risk level considering several factors including ship age, flag, insurer, classification, and operator. Degré (2007) also adopted the concept of ship risk to select high-risk vessels for inspection. The risk concept in this study was evaluated by the probability of the occurrence of casualties and its potential consequences. Based on the method to generate black-grey-white flag performance list in Paris MoU, Degré (2008) derived the black-grey-white ship category risk list with regard to the observed casualties over a given period. Information of past incidents and accidents was also used by Heij and Knapp (2019) and Knapp and Heij (2020), where combined models considering such information together with historical detentions were used to target high-risk vessels and prioritize inspection areas. Dinis et al. (2020) adopted parameters from the NIR used in Paris MoU as risk variables to assess individual ship and maritime traffic risk level using Bayesian network (BN) models.

Ship detention or deficiency serves as the concrete prediction target in more current studies, as both of them are the main outcomes of PSC inspections. Xu et al. (2007) proposed one of the pioneer machine learning-based models to predict ship detention by developing support vector machine (SVM). In recent years, more advanced models are developed for accurate selection of high-risk ships. BN is a type of popular model for ship condition prediction. Yang et al. (2018a) developed a BN for bulk carrier detention prediction at the ports in Paris MoU. Key factors influencing ship detention, such as the number of deficiencies, inspection type, RO, and ship age were identified. Based on the output of the BN, a game model was developed to determine the optimal inspection rate at the ports to improve inspection efficiency. BN model was also developed by Wang et al. (2019) to predict ship deficiency number using the real inspection records at the Hong Kong port. To match ship deficiency condition with the expertise of PSCOs, Yan et al. (2020a) proposed three two-step models for ship condition prediction and PSCO assignment. The structure of the PSCO assignment model was partially considered in some ship condition predictions models, which were based on random forest consisting of several multi-target regression trees. It is noted that the PSCO assignment model is quite different from the one proposed in this study as it aimed to match various ship conditions with the expertise of the PSCOs. Yan et al. (2020b) developed a balanced random forest model to predict ship detention probability, which addressed the problem of highly unbalanced distribution of inspection records with and without detention. A combined ship risk prediction model considering ship deficiency and detention simultaneously was developed by Yan et al. (2021a) to address different needs of the port states.

To improve the efficiency of onboard inspection, association rule mining technologies are adopted to figure out the relationship between various factors. The generated rules can offer meaningful insights to onboard deficiency and detention identification. Tsou (2019) explored the detention database of Tokyo MoU using association rule mining techniques. The author identified the correlations between detention deficiencies and the correlations between deficiencies and ship-/inspection-related factors. Chung et al. (2020) analyzed the historical PSC inspection records in Taiwan Provence of China using Apriori algorithm. The correlations between ship characteristics and PSC deficiencies were identified. Yan et al. (2021b) also adopted Apriori algorithm to identify the relationship between ship deficiencies based on the inspection records at the Hong Kong port. Onboard inspection schemes were then proposed according to the rules identified. Fu et al. (2020) analyzed the correlations between ship generic properties and ship deficiency and detention conditions using Apriori algorithm based on the inspection records in Tokyo MoU.

Although there are various studies proposing models for ship condition prediction and inspection efficiency improvement, there are several limitations in current research. First, current studies of high-risk ship selection have failed to consider shipping domain knowledge in ship risk prediction, including the monotonicity regarding ship flag/RO/company performance in ship risk prediction. It is likely that the prediction models ignoring such domain knowledge give opposite prediction results due to model inaccuracy and noises in training data (Sill, 1997; Duivesteijn and Feelders, 2008; Daniels and Velikova, 2010; Pei et al., 2016). This indicates that only by taking such shipping domain knowledge into account in ship risk prediction models can fair and reasonable prediction results be generated. Here regarding such prediction results to be "fair" is because for ship flag/RO/company which adopt more effective management measures on their ships, it can be expected that their ships' performance in PSC inspection should be better than other flags/ROs/companies adopting worse management strategies. In return, reducing the inspection frequency of their ships can promote them to better fulfill their maintenance and operational duties and attract more shippers to choose their services. The reason to regard such prediction results to be "reasonable" is that considering monotonicity into a machine learning model "can be an important model requirement with a view toward explaining and justifying decisions" (Duivesteijn and Feelders, 2008) to the decision makers. It is also reported by Pazzani et al. (2001) that the learned rules with monotonicity constraints were significantly more acceptable to experts than rules learned without the monotonicity restrictions when experts expect certain monotonicity based on their experience.

Second, there is little literature aiming to design tailored PSCO assignment or scheduling schemes for ship inspection, and thus to validate the superiority of the proposed ship risk prediction models over the current schemes at ports. Indeed, prediction model accuracy is figured out in many current studies, and their superiority over current ship selection scheme is also presented. However, port inspection resources (e.g. the number of available PSCOs) are scarce and the arrival and departure time of ships are not fixed. This indicates that not all ships can be inspected in practice, and the gap between the proposed and current schemes in practice remains to be validated. Formulation and solution techniques for assignment and scheduling models are proposed in current literature, and typical modeling approaches include column generation (Van Den Akker et al., 2005; Huisman, 2007; Janacek et al., 2017; Kulkarni et al., 2018) and Dantzig-Wolfe decomposition (Janacek et al., 2017; Kulkarni et al., 2018; Muñoz et al., 2018). Nevertheless, there is no tailored modelling approach considering the problem structure and the corresponding properties of PSC inspection as well as proposing intuitive solving strategies that are comprehensible to the decision makers at port authorities. Therefore, it is of vital importance to develop tailored and easy-to-understand PSCO assignment and scheduling modeling approach based on ship risk prediction models to figure out their superiority in practice and improve the efficiency of PSC inspection.

To address these issues, this study develops a highly accurate XGBoost model for ship deficiency number prediction considering shipping domain knowledge. It then proposes PSCO scheduling models based on the predictions which are consistent with the actual situation at the ports. Extensive computational experiments and sensitivity analysis are conducted to validate the model performance.

3. Data and model validation metrics

3.1. Data description

The case dataset of this study contains 1,974 PSC initial inspection records and the corresponding ship related factors at the Hong Kong port from January 2016 to December 2018. Especially, PSC inspection records are downloaded from the public database provided by Tokyo MoU[1], and ship related factors are searched from World Shipping Register database. The prediction target is the number of deficiencies detected in the current PSC inspection. We consider 14 features that are regarded to be highly related to ship deficiency number in the current literature and by domain knowledge, namely ship age, gross tonnage (GT), length, depth, beam, type, flag performance, RO performance, and company performance in Tokyo MoU, last PSC inspection date in Tokyo MoU, the number of deficiencies in last inspection in Tokyo MoU, the number of total detentions in all historical PSC inspections, the number of flag changes, and whether the ship has a casualty in last 5 years. Moreover, as required by the Tokyo MoU, from the best to the worst, the states of ship flag performance are white, grey, and black, the states of ship RO and company performance are high, medium, low, and very low, respectively. After data preprocessing, the whole dataset contains 1,926 samples. The explanation, method of feature encoding, and the descriptive statistics of the prediction target and the 14 features in the whole dataset are shown in Appendix A.

3.2. Model validation metrics

The deficiency number prediction models are validated using two common metrics for regression problems in machine learning: mean squared error (MSE) and mean absolute error (MAE). Given a total of n samples in the dataset, the real output y_i and the predicted output $\hat{y}_i$ for sample i, $i = 1, ..., n$, the definitions of MSE and MAE are as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2, \tag{1}$$

[1] http://www.tokyo-mou.org/inspections_detentions/psc_database.php

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{y}_i - y_i\right|. \tag{2}$$

4. Introduction and construction of XGBoost model

4.1. The structure of XGBoost model

Ensemble models in machine learning combine the predictions of multiple simpler base models to improve the overall model prediction performance (Friedman et al., 2001). Two main ensemble models are bagging (bootstrap aggregating) and boosting. Bagging builds several base models independently and then average their predictions. Boosting builds sequential and dependent base models in the way that one base model is built considering the errors of the base models built so far and then produces a powerful ensemble (Friedman et al., 2001). In boosting models, a base model is also called a weak learner which may be only slightly better than random guessing. Meanwhile, the main idea of boosting is to add new weak learners to the ensemble sequentially, and in each iteration, the weak learner is trained with respect to the error of the whole ensemble learned so far (Natekin and Knoll, 2013). As boosting is purely algorithm-driven, a gradient-descent based formulation of boosting methods is derived which is called gradient boosting machine (GBM) (Freund and Schapire, 1997; Friedman et al., 2000). The principal idea of GBM is to construct new weak learners to be maximally correlated with the negative gradient of the loss function associated with the whole ensemble.

XGBoost (short for eXtreme Gradient Boosting) is an implementation of GBM that uses tree-structured weak learners (Chen and Guestrin, 2016). It is highly effective (which allows parallel and distributed computing) and scalable (which is able to handle datasets containing billions of examples in distributed or memory-limited settings). Due to limited space, the detailed process to construct XGBoost model is presented in Appendix B.

4.2. Feature monotonic constraints in XGBoost

Apart from the state-of-the-art prediction performance, XGBoost also has the nice property to enforce monotonic constraint on the feature(s) regarding the prediction target (Chen, 2016). Suppose we have a total of m features and the feature vector is denoted by $\mathbf{x} = (x^1,\ldots, x^{m'},\ldots, x^m)$.We put a monotonically increasing constraint on feature $\bar{m}$, which means that for two samples i_1 and i_2 that have the same feature values except for $x^{\bar{m}}$, i.e. $x_{i_1}^{m'} = x_{i_2}^{m'}$, $m' = 1,\ldots,\bar{m}-1,\ \bar{m}+1,\ldots,m$ and $x_{i_1}^{\bar{m}} < x_{i_2}^{\bar{m}}$, the predicted target for i_1 should be no more than that for i_2, i.e. $\hat{y}_{i_1} \le \hat{y}_{i_2}$. As the monotonic constraint works in the context that all the features are equal in the samples except for the feature which is enforced to be monotonic (denote the set of samples by I'), the prediction process of the samples in I' in a tree can be simplified to only contain the splits on the monotonic feature (as using all other features and values will always lead the samples to the same tree nodes and thus have the same output). In this context, the working process to impose monotonic constraint on a feature can be illustrated as follows.

We still use feature $\bar{m}$ which we put a monotonically increasing constraint on as an example. An illustration of the tree structure is shown in Figure 1. The output of all samples in the root node is W_0. From splitting the root node, we would expect the weight assigned to the right child not to be lower than the weight assigned to the left child while using the monotonic feature for splitting. When feature $\bar{m}$ is picked to split the root node, if a candidate value of $\bar{m}$ leads to a higher weight in the left child than that in the right child, this candidate value will be abandoned for the current node splitting. That is, when enumerating all possible values of feature $\bar{m}$ to split the root node, only the values leading to no lower weights in right child than in left child will be retained for further comparison. If all possible splits lead to higher output in the left child than in the right child, the node would not be split any more. If feasible splits exist and the optimal splitting point is found, we could have $W_L \le W_R$, where W_L is the output of the left child node while W_R is the output of the right child node. When splitting node L to left child LL and right child LR, only the splits that lead to $W_{LL} \le W_{LR}$ will be considered, where W_{LL} is the output of LL and W_{LR} is the output of LR. As the weight of LR should be no more than the weight of node R, we further impose an upper bound for W_{LR} as $W_{LR} \le \mathrm{mean}(W_L, W_R) = \frac{W_L+W_R}{2} \le W_R$. Similarly, apart from ensuring $W_{RL} \le W_{RR}$,where W_{RL} is the output of RL and W_{RR} is the output of RR, we also impose a lower bound for W_{RL} as $W_{RL} \ge \mathrm{mean}(W_L, W_R) = \frac{W_L+W_R}{2} \ge W_L$. Consequently, in this tree level we can guarantee $W_{LL} \le W_{LR} \le \mathrm{mean}(W_L,W_R) \le W_{RL} \le W_{RR}$. As a tree is split in a recursive manner, the monotonicity of the whole tree can be guaranteed. A more detailed explanation of the situations if more than one feature is imposed by monotonic constraints is given in Appendix C.

It should also be noted that as XGBoost allows for feature subsampling when constructing each tree, the monotonic feature may not be included in some trees. For those trees, as the samples in I' have the same feature values except for the feature with monotonic constraint, all the samples will be assigned to the same leaf node and thus have the same output. As XGBoost is an additive model, the predicted output of each sample is the sum of the outputs in all the trees where the monotonicity constraints are preserved. Therefore, the feature monotonicity of the final output in the whole model can be preserved.

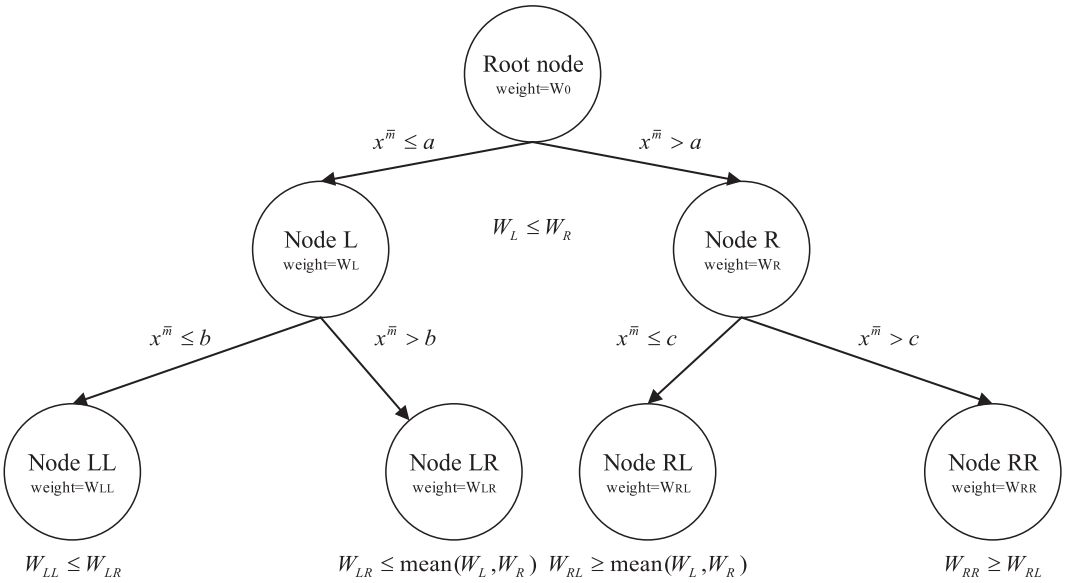

Fig. 1. Illustration of feature monotonicity on XGBoost.

Table 1
Hyperparameters in XGBoost model*.

Hyperparameter	Meaning
learning_rate (c**)	Step size shrinkage used to update the predicted values after each boosting step to prevent overfitting which can be applied to Eq.(B4).
n_estimators (a)	The number of weak learners (decision trees) in the XGBoost model (i.e. *K* in Eq. (B1)).
max_depth (b)	The maximum depth of each tree.
min_child_weight (b)	The minimum sum of sample weight (Hessian) (i.e. H_j^t) needed in a child node. In a regression tree with loss function as MSE, the sum of sample weight in a node equals the number of samples contained in the node.
delta (b)	The minimum loss reduction required to make a split for a node.
sub_sample (b)	The fraction of samples to be randomly sampled for each tree.
colsample_bytree (b)	The fraction of columns (features) to be randomly sampled for each tree.
reg_gamma (b)	L_1 regularization term on tree complexity (i.e. γ in Eq. (B9)).
reg_lambda (b)	L_2 regularization term on tree complexity (i.e. λ in Eq. (B9)).

Note*: to avoid ambiguity, we have renamed some hyperparameters. For example, in the XGBoost Module for Python, '*delta*' is called '*lambda*', and '*reg_gamma*' is called '*reg_alpha*'.
Note**: this indicates the hyperparameter category.

4.3. Construction of monotonic XGBoost

The whole dataset is randomly divided into training set (80% samples) and test set (20% samples, denoted by test set i), which contain 1,524 samples and 384 samples, respectively. The XGBoost model with monotonic constraints enforced on three features, i.e. ship flag, RO and company performance, is constructed using the training set (which we call monotonic XGBoost). Hyperparameters contained in XGBoost are in three categories: a. general parameter, which guides the overall functioning; b. booster parameters, which guide the individual booster at each iteration; and c. learning task parameter, which guides the optimization performed. We use regression decision tree as the weak learner in XGBoost model. The hyperparameters tuned in this study are summarized in Table 1.

Table 1 shows that there are totally nine hyperparameters that need to be tuned in an XGBoost model, which can be a huge burden if we apply cross validation with grid search to tune the hyperparameter tuple directly. To address this issue, we propose a three-step hyperparameter tuning method after giving the initial values of the hyperparameters based on experience. In the first step, the hyperparameters are tuned in turns according to their categories using grid search based on 5-fold cross validation with MSE as the metric, and their initial tuned values can be found. In the second step, an extended

Table 2
Finally adopted hyparameter values in monotonic XGBoost.

Hyperparameter	*n_estimators*	*learning_rate*	*max_depth*	*min_child_weight*	*delta*
value	200	0.02	5	4	0.15
Hyperparameter	*sub_sample*	*colsample_bytree*	*reg_gamma*	*reg_lambda*	
value	0.75	0.4	0.1	0.1	

Table 3
Features of an example in test set i except for flag, RO, and company performance.

Feature	Value
age	12
GT	6813
length	132.6
depth	9.2
beam	19.2
type	container ship
last inspection date	4.3
last deficiency number	6
total detentions	0
the number of flag changes	0
casualty in the last 5 years	0

Table 4
An example of construction variant samples and the prediction results.

Sample	flag	RO	company	Output of monotonic XGBoost	Increase between consecutive values	Output of normal XGBoost	Increase between consecutive values
Original sample	1	1	3	5.3443 (true: 5)	\	5.6563 (true: 5)	\
variant sample 1	**1**	1	3	5.3443	\	5.6563	\
variant sample 2	**2**	1	3	5.9879	0.6437	5.9409	0.2846
variant sample 3	**3**	1	3	6.3320	0.3441	5.8450	**-0.0959**
variant sample 4	1	**1**	3	5.3443	\	5.6563	\
variant sample 5	1	**2**	3	5.5915	0.2473	5.6383	**-0.0180**
variant sample 6	1	**3**	3	5.5915	0	5.6383	0
variant sample 7	1	1	**1**	3.9397	\	3.9384	\
variant sample 8	1	1	**2**	4.6101	0.6703	4.4111	0.4728
variant sample 8	1	1	**3**	5.3443	0.7342	5.6563	1.2452
variant sample 10	1	1	**4**	7.2423	1.8981	7.5755	1.9192

searching space for all the hyperparameters consisting of the initial tuned value and two more candidate values near the tuned value for each hyperparameter is formed. Then, grid search based on 5-fold cross validation with MSE as the metric is conducted on all hyperparameters simultaneously. In the third step, '*learning_rate*' is further reduced and '*n_estimators*' is further increased to improve model generalization ability. The finally adopted values for the hyperparameters are shown in Table 2.

After hyperparameter tuning using cross validation on the training set, the final monotonic XGBoost model is constructed using the whole training set with the optimal hyperparameter values presented in Table 2. Its performance is validated by test set i. The MAE of the monotonic XGBoost model is 2.372 and the MSE is 12.470.

4.4. Analysis of monotonic XGBoost

We form another test set (denote by test set ii) as an extension of test set i to validate the monotonicity in the output of the monotonic XGBoost model regarding the three monotonic features: flag performance, RO performance, and company performance. For each sample in test set i, we form 10 variant samples by setting the values for flag performance from 1 to 3 (i.e. from white to black), RO performance from 1 to 3 (i.e. from high to low), and company performance from 1 to 4 (i.e. from high to very low) respectively while keeping the other features and their values unchanged. Totally we can have 3,840 samples ($3,\ 840 = 384 \times 10$) in test set ii. We use a random sample in test set i as an example to show the construction process and the predicted results using the normal XGBoost model and the monotonic XGBoost model. The sample features except for flag, RO, and company performance are shown in Table 3. The flag, RO and company performance together with the prediction results are shown in Table 4.

Table 5
Increase in predicted deficiency number of consecutive states in test set ii.

State change	Flag performance	RO performance	Company performance
1->2	0.8030	0.2530	0.5312
2->3	0.2236	0	0.7787
3->4	\	\	1.4919

Table 4 indicates that in the monotonic XGBoost model, the predicted deficiency number increases as the performance of flag, RO and company gets worse, respectively. Moreover, increase between consecutive states of company performance is most significant in this example: on average, 1.1009 more deficiencies can be detected if it gets worse by one state. Meanwhile, change in RO performance is the least obvious: when its RO performance change from 1 (high) to 2 (medium), only 0.2473 more deficiencies will be detected; the number of detected deficiencies remains unchanged while the RO performance changes from 2 (medium) to 3 (low). Meanwhile, it can also be seen in Table 4 that in a normal XGBoost model, the monotonicity of the three features cannot be fully guaranteed: when flag performance changes from medium to low, and when RO performance changes from high to medium, the predicted deficiency number decreases instead, which is against domain knowledge.

We further calculate the average increase between consecutive states of each feature over the whole test set as shown in Table 5.

Table 5 indicates that when the states of flag performance change from high to medium and from medium to low, the increase of deficiency number gets smaller. While the state values increase by 1 in company performance, the increase of deficiency number gets larger. On the contrary, when RO performance gets from 2 (medium) to 3 (low), 0 more deficiency number will be detected as suggested by the monotonic XGBoost model. This is because there is only one sample in the training set with RO performance as low, which makes it hard for the model to capture the change in deficiency number when RO performance gets from medium to low. It should also be noted that although in Tokyo MoU the worst performance for RO is “very low”, as there are no such inspection records between 2016 and 2018, we only form variant samples with RO performance to be high, medium, and low.

4.5. Comparison with other popular machine learning models

We compare the performance of the other popular machine learning models with the monotonic XGBoost model using test set i and the same training set. Especially, we compare the performance of normal XGBoost, CART based regression decision tree (DT) (Breiman et al., 1984), random forest (RF) (Breiman, 2001), gradient boosting decision tree (GBDT) (Friedman, 2001), monotonic light gradient boosting machine (LightGBM) (Ke et al., 2017), least absolute shrinkage and selection operator (LASSO) regression (Santosa and Symes, 1986), ridge regression (Hoerl and Kennard, 1970), and support vector machine (SVM) (Drucker et al., 1996) with the monotonic XGBoost model. It should be noted that apart from LightGBM, none of the other machine learning models can guarantee the monotonic constraints of the three features. For SVM, DT, RF, LASSO regression and ridge regression, grid search with 5-fold cross validation is applied directly for hyperparameter tuning as they have fewer hyperparameters. For normal XGBoost, GBDT and monotonic LightGBM, the hyperparameter tuning method is similar to that used in the monotonic XGBoost model. The MSE and MAE in test set i are shown in Table 6.

Table 6 shows that the prediction performance of monotonic XGBoost ranks first regarding both MSE and MAE among all the machine learning models considered. Regarding MSE, monotonic LightGBM ranks second, followed by normal XGBoost. Regarding MAE, SVM is slightly worse than monotonic XGBoost, followed by normal XGBoost. Ridge regression has the worst performance regarding both metrics. Especially, the monotonic XGBoost performs better than the normal XGBoost whose hyperparameter values are tuned by the same hyperparameter tuning method regarding both MSE and MAE, which is in line with the comment that if reasonable monotonic constraints on certain features are enforced, model prediction performance should be improved, meaning that the constrained models may generalize better (Sill, 1997; Duivesteijn and Feelders, 2008; Daniels and Velikova, 2010; Pei et al., 2016).

To conclude, a tree-based gradient boosting machine called XGBoost, where shipping domain knowledge regarding ship flag/RO/company performance for ship risk prediction in PSC inspection can be incorporated in a natural and rational way, is developed and validated in this section. The structure of XGBoost and detailed steps to develop an XGBoost model, especially how to incorporate monotonic constraints in the model are first introduced. The performance of the developed XGBoost model is then validated and compared with other popular machine learning models. It is shown that the XGBoost model considering domain knowledge has the best performance among all the machine learning models concerned.

5. PSC officer scheduling problem

The PSC officer (PSCO) scheduling model aims to assign the available PSCOs to inspect the foreign visiting ships that need to be inspected as required (i.e. ships with no previous inspection records and ships out of/within the inspection time window). Human and time inspection resources, the predicted deficiency condition of the ships, and the berthing time of the ships at port should be considered in the model. As there are many foreign ships visiting a port for each day while

Table 6
MSE and MAE in test set i of the machine learning models.

Model	monotonic XGBoost*	normal XGBoost	DT	RF	GBDT	Monotonic LightGBM*	LASSO regression	ridge regression	SVM
MSE	**12.470**	12.779	15.625	13.612	13.322	12.747	15.089	15.765	13.421
Rank	**1**	3	8	6	4	2	7	9	5
MAE	**2.372**	2.422	2.672	2.459	2.461	2.475	2.806	2.909	2.411
Rank	**1**	3	7	4	5	6	8	9	2

Note*: monotonicity of the three features can be preserved.

Table 7
Relationship between time periods and units.

Time period	Time unit	Time period	Time unit	Time period	Time unit
8:00 to 8:30	1	11:00 to 11:30	7[a]	14:00 to 14:30	13
8:30 to 9:00	2	11:30 to 12:00	8	14:30 to 15:00	14
9:00 to 9:30	3	12:00 to 12:30	9[b]	15:00 to 15:30	15
9:30 to 10:00	4	12:30 to 13:00	10	15:30 to 16:00	16
10:00 to 10:30	5	13:00 to 13:30	11[c]	16:00 to 16:30	17
10:30 to 11:00	6	13:30 to 14:00	12	16:30 to 17:00	18

a: The latest time unit to start inspection before lunch break.
b: The earliest time unit to start inspection after lunch break.
c: The latest time unit to start lunch break.

Table 8
Notation used in the problem.

Sets	
S	The set of foreign ships that need to be inspected for one day.
P	The set of PSCOs on duty for that day.
H	The set of *inspection templates*. An *inspection template* is a set of ships which is feasible to be inspected by one PSCO while guaranteeing his/her lunch break within the daily work time.
$\tilde{\text{H}}$	The set of *un-dominated inspection templates*.
Indices	
s	The index for a ship in S.
p	The index for a PSCO in P.
τ	The index for a time unit.
η	The index of an *inspection templat*ein H.
Parameters	
T	The total number of time units for a working day.
d_s	The predicted deficiency number of ship s using the XGBoost model.
D_η	The number of deficiencies that can be detected if *inspection template* η is adopted.
e_τ^s	Binary parameter indicating whether ship s is available for inspection in time unit τ.
δ_s^η	Binary parameter indicating whether ship s is contained in *inspection templat eη*.
$B = [b_{s's''}]_{\lvert S\rvert \times \lvert S\rvert}$	Binary matrix indicating the relationship between each of the two ships that need to be inspected.

the inspection resources are scarce, the PSCO scheduling model aims to decide the set of ships to be inspected and assign the selected ships to the PSCOs so as to maximize the inspection benefit, which is represented by the total number of deficiencies that can be identified.

Denote the set of foreign ships that need to be inspected on one day as S and one ship as $s \in S$. Denote the set of PSCOs on duty for this day as P and one PSCO as $p \in P$. The work time for the PSCOs is stable for each day: they work from 8:00 to 11:00 in the morning, and 14:00 to 17:00 in the afternoon. They spend one hour for lunch break during 11:00 to 14:00, and the other two hours for working. For example, if PSCO p has lunch break during 12:00 to 13:00, his/her work time should be from 8:00 to 12:00 and from 13:00 to 17:00. A typical PSC inspection takes about 2 hours, and thus we assume the duration of a PSC inspection to be two hours for all ships. For ship $s \in S$, its deficiency number d_s is predicted by the monotonic XGBoost model which should be treated as a parameter. Each ship berths at the port for a period in each day, and the available time for ship s during 8:00 to 17:00 (i.e. the daily work time for PSCOs) for PSC inspection is reported to the port state in advance. We divide the work hours from 8:00 to 17:00 for PSCOs into $\text{T} = 18$ time units with each time unit as 0.5 hour, indexed by τ. The relationship between the time periods and the time units is illustrated in Table 7.

Based on the ship berthing information reported in advance, we further introduce a parameter e_τ^s which is set to 1 if ship s stays at the port in the whole period of time unit τ. For example, if ship s stays at the port from 01:00 to 12:00, we should set $e_\tau^s = 1, \tau = 1, 2, ..., 8$ and $e_\tau^s = 0, \tau = 9, 10, ..., 18$. We assume that the inspection starting time of a ship and the lunch break starting time of a PSCO are at the beginning of one time unit. The PSCO scheduling problem aims to select the ships for inspection, to decide the inspection starting time of the selected ships, to assign the selected ships to the PSCOs, and to decide the lunch break starting time of the PSCOs to maximize the inspection benefits. The notation used in the PSCO scheduling problem is listed in Table 8.

5.1. PSCO scheduling model M1

To formulate the PSCO scheduling problem, we define two types of main binary decision variables: $x_{sp\tau}$, which is set to 1 if ship s is inspected by PSCO p in time unit τ and 0, otherwise; and r_p^τ, which is set to 1 if PSCO p has lunch break in time unit τ and 0, otherwise. Besides, we also introduce three types of auxiliary binary decision variables: y_{sp}, which is set to 1 if ship s is inspected by PSCO p and 0, otherwise; σ_{sp}^τ, which is set to 1 if ship s starts to be inspected by PSCO $p$from time unit τ and 0, otherwise; and θ_p^τ, which is set to 1 if PSCO p starts to have lunch break from time unit τ and 0, otherwise.To maximize the inspection benefit by maximizing the estimated total number of deficiencies that can be detected, an integer linear optimization model M1 is proposed as follows.

[M1]

$$\max \sum_{s\in S}\sum_{p\in P} d_s y_{sp} \tag{3}$$

s.t.

$$\sum_{p\in P} y_{sp} \le 1, \forall s \in S \tag{4}$$

$$x_{sp\tau} \le e_\tau^s, \forall s \in S, \forall p \in P, \tau = 1, ..., T \tag{5}$$

$$x_{sp\tau} \le 1 - r_p^\tau, \forall s \in S, \forall p \in P, \tau = 1, ..., T \tag{6}$$

$$\sum_{s\in S} x_{sp\tau} \le 1, \forall p \in P, \tau = 1, ..., T \tag{7}$$

$$\sum_{\tau=1}^{T} x_{sp\tau} = 4y_{sp}, \forall s \in S, \forall p \in P \tag{8}$$

$$\sum_{\tau'=\tau}^{\tau+3} x_{sp\tau'} \ge 4\sigma_{sp}^\tau, \forall s \in S, \forall p \in P, 1 \le \tau \le 15 \tag{9}$$

$$\sum_{\tau=1}^{T} \sigma_{sp}^\tau = y_{sp}, \forall s \in S, \forall p \in P \tag{10}$$

$$\sigma_{sp}^\tau = 0, \forall s \in S, \forall p \in P, 16 \le \tau \le 18 \tag{11}$$

$$\sum_{\tau=7}^{12} r_p^\tau = 2, \forall p \in P \tag{12}$$

$$r_p^\tau = 0, \forall p \in P, \tau \in [1, 6] \cup [13, 18] \tag{13}$$

$$\sum_{\tau'=\tau}^{\tau+1} r_p^{\tau'} \ge 2\theta_p^\tau, \forall p \in P, 7 \le \tau \le 11 \tag{14}$$

$$\sum_{\tau=1}^{T} \theta_p^\tau = 1, \forall p \in P \tag{15}$$

$$\theta_p^\tau = 0, \forall p \in P, \tau \in [1, 6] \cup [12, 18] \tag{16}$$

$$x_{sp\tau} \in \{0, 1\}, \forall s \in S, \forall p \in P, \tau = 1, ..., T \tag{17}$$

$$y_{sp} \in \{0, 1\}, \forall s \in S, \forall p \in P \tag{18}$$

$$\sigma_{sp}^\tau \in \{0, 1\}, \forall s \in S, \forall p \in P, \tau = 1, ..., T \tag{19}$$

$$r_p^\tau \in \{0,1\}, \forall p \in P, \tau = 1, ..., T \tag{20}$$

$$\theta_p^\tau \in \{0,1\}, \forall p \in P, \tau = 1, ..., T. \tag{21}$$

Objective function (3) maximizes the inspection benefits by maximizing the estimated total number of deficiencies that can be detected. Constraints (4) ensure that each ship can only be inspected by at most one PSCO. Constraints (5) and (6) guarantee that a ship can only be inspected when it is at port and when the corresponding PSCO does not have lunch break. Constraints (7) ensure that a PSCO can only inspect one ship in one time unit. Constrains (8) to (11) guarantee that if a ship is inspected, it should be inspected during 4 consecutive time units, and the start inspection time unit is between 1 and 15. Constraints (12) to (16) guarantee that each PSCO can have a one-hour consecutive lunch break between time units 7 and 12. Constraint (17) to (21) ensure the domain of the decision variables.

5.2. PSCO scheduling model M2

As the PSCOs are indifferent from each other, there will be an exponential number of optimal solutions to mathematical model M1, which will reduce the efficiency to solve M1. A possible approach is dynamic programming. For example, Wang et al. (2018) applied dynamic programming for the selection of waste disposal ports in cruise shipping; Yi and Sutrisna (2021) proposed a novel dynamic programming method that elegantly addresses the drone scheduling problem for construction site surveillance. However, after examination, we find that our problem suffers from the "curse-of-dimensionality" and cannot use dynamic programming. As the total work time of a PSCO for one day is 8 hours and an inspection would take 2 hours, a PSCO can inspect 0, 1, 2, 3, or 4 ships for one day. Therefore, the PSCO scheduling problem can be reformulated as identifying and assigning the sets of ships that can be inspected by one PSCO to the available PSCOs. Define L as the number of ships inspected by one PSCO, $L = 0, 1, 2, 3, 4$. Given the value for L, the total number of combinations of L ships from the total $|S|$ ships is $C_{|S|}^L$, $C_{|S|}^L = \binom{|S|}{L} = \frac{|S|!}{L!(|S|-L)!}$. Given a combination of L ships, denoted by set S', $S' \subset S$, $|S'| = L$. we examine whether it is feasible to inspect all the ships in S' by one PSCO. If it is feasible, then we call set an *inspection template* and our aim is to choose *inspection templates* (each template is assigned to one PSCO) that maximize the total number of deficiencies that can be detected while ensuring a ship is included in at most one chosen template (i.e. a ship is inspected at most once). Here the concept of "template" is similar with the concept of berth template (Zhen, 2015) and yard template (Zhen 2016), which have been widely used in some pioneering work such as Zhen et al. (2011) in the field of port and shipping management.

To examine whether it is feasible to inspect all the ships in S' by one PSCO, $|S'| = L$, we note that a PSCO has to carry out $L + 1$ activities to inspect all the ships in S' between time unit $\tau = 1$ (8:00) and $\tau = 18$ (17:00), that is, inspecting each of the L ships and having lunch break. We define α as an activity, and each activity has a duration t_α, an earliest start time ω_α, and a latest completion time ϖ_α. If an activity α is inspecting a ship, denoted by ship s, then $t_\alpha = 4$, ω_α is the start time of the ship's berthing between $\tau = 1$ and $\tau = 18$ of the day, i.e., $\omega_\alpha = \min\{\tau = 1, ..., 18 | e_\tau^s = 1\}$, ϖ_α is the ship's departure time if it departs before $\tau = 18$ and otherwise $\varpi_\alpha = 18$, i.e., $\varpi_\alpha = \max\{\tau = 1, ..., 18 | e_\tau^s = 1\}$; if an activity α is having lunch break, then $t_\alpha = 2$, $\omega_\alpha = 7$, $\varpi_\alpha = 12$. There are a total of $(L + 1)!$ different sequences for the PSCO to conduct the activities (note that some, or even all of the sequences may be infeasible). For a particular sequence, we denote the activities carried out by $\alpha_1 \to \alpha_2 \to ... \to \alpha_{L+1}$, that is, α_ℓ is the ℓth activity, t_{α_ℓ}, ω_{α_ℓ}, ϖ_{α_ℓ} are the duration, earliest start time, and latest completion time of activity α_ℓ, respectively. To check whether the $L + 1$ activities can be carried out in the above sequence, we define $\mathcal{T}_\ell$ as decision variable representing the start time of carrying out activity α_ℓ, then the $L + 1$ activities can be carried out in the above sequence by one PSCO if and only if there is a solution $\mathcal{T}_\ell$, $\ell = 1, ..., L + 1$, that satisfies the following constraints:

$$\mathcal{T}_\ell \geq \omega_{\alpha_\ell}, \ell = 1, ..., L+1 \tag{22}$$

$$\mathcal{T}_\ell + t_{\alpha_\ell} - 1 \leq \varpi_{\alpha_\ell}, \ell = 1, ..., L+1 \tag{23}$$

$$\mathcal{T}_{\ell+1} \geq \mathcal{T}_\ell + t_{\alpha_\ell} \ell = 1, ..., L. \tag{24}$$

Note that if an activity α_ℓ with duration t_{α_ℓ} starts at the beginning of time unit $\mathcal{T}_\ell$, its completion time should be at the end of time unit $\mathcal{T}_\ell + t_{\alpha_\ell} - 1$.

Proposition 1: For an *activity sequence*, whether constraints (22)–(24) have a feasible solution can be checked below: for activity α_1, let its start time $\mathcal{T}_1^* = \omega_{\alpha_1}$; for activity α_l, $l = 2, ..., L + 1$, let its start time $\mathcal{T}_\ell^* = \max\{\mathcal{T}_{\ell-1}^* + t_{\alpha_{\ell-1}}, \omega_{\alpha_\ell}\}$, $\ell = 2, ..., L + 1$; if $\mathcal{T}_\ell^* \leq \varpi_{\alpha_\ell} - t_{\alpha_\ell} + 1, l = 1, ..., L + 1$, then the *activity sequence* is feasible; otherwise it is infeasible.

Proof:

The "if" part of the proposition is straightforward because it is easy to check that $(\mathcal{T}_\ell^*, \ell = 1, ..., L+1)$ is indeed feasible to constraints (22)–(24). To prove the "only if" part, suppose that constraints (22)–(24) have a feasible solution $(\mathcal{T}_\ell^\#, \ell = 1, ..., L+1) \neq (\mathcal{T}_\ell^*, \ell = 1, ..., L+1)$. Denote by $\hat{\ell}$ the index of the first different elements of vectors $(\mathcal{T}_\ell^\#, \ell = 1, ..., L+1)$ and

$(\mathcal{T}^*_\ell, \ell = 1, ..., L+1)$, that is $\mathcal{T}^{\#}_\ell = \mathcal{T}^*_\ell$, $\ell = 1, ..., \hat{\ell} - 1$ and $\mathcal{T}^{\#}_{\hat{\ell}} \neq \mathcal{T}^*_{\hat{\ell}}$. If $\hat{\ell} = 1$, we define a new vector $(\mathcal{T}^{\&}_\ell, \ell = 1, ..., L+1)$ such that $\mathcal{T}^{\&}_1 = \omega_{\alpha_1}$ and $\mathcal{T}^{\&}_\ell = \mathcal{T}^{\#}_\ell$, $\ell = 2, ..., L+1$. If $\hat{\ell} = 2, ..., L+1$, we define a new vector $(\mathcal{T}^{\&}_\ell, \ell = 1, ..., L+1)$ such that $\mathcal{T}^{\&}_\ell = \mathcal{T}^{\#}_\ell$, $\ell = 1, ..., \hat{\ell} - 1$, $\mathcal{T}^{\&}_{\hat{\ell}} = \max\{\mathcal{T}^{\&}_{\hat{\ell}-1} + t_{\alpha_{\hat{\ell}-1}}, \omega_{\alpha_{\hat{\ell}}}\}$ and $\mathcal{T}^{\&}_\ell = \mathcal{T}^{\#}_\ell$, $\ell = \hat{\ell} + 1, ..., L+1$. In both cases, it is easy to check that $(\mathcal{T}^{\&}_\ell, \ell = 1, ..., L+1)$ is feasible to constraints (22)–(24). We can now set $(\mathcal{T}^*_\ell, \ell = 1, ..., L+1) \leftarrow (\mathcal{T}^{\&}_\ell, \ell = 1, ..., L+1)$ and repeat the above procedure. It can be seen that by repeating the above procedure at most $L+1$ times, we will generate a feasible solution $(\mathcal{T}^{\&}_\ell, \ell = 1, ..., L+1)$ that is identical to $(\mathcal{T}^*_\ell, \ell = 1, ..., L+1)$. In other words, constraints (22)–(24) have a feasible solution only if $(\mathcal{T}^*_\ell, \ell = 1, ..., L+1)$ is feasible. This concludes the proof of the proposition. □

We use the following example to illustrate the steps to decide whether an *activity sequence* $\alpha_1 \rightarrow \alpha_2 \rightarrow ... \rightarrow \alpha_{L+1}$ is feasible.

Example 1. Given $L = 3$ and $S' = \{s_1, s_2, s_3\}$ for *activity sequence* $\alpha_1 \rightarrow \alpha_2 \rightarrow \alpha_3 \rightarrow \alpha_4$. Particularly, activities α_1, α_2, and α_4 are ship inspections for s_1, s_2, and s_3 respectively and activity α_3 is lunch break. The berthing periods of s_1, s_2, and s_3 are during 8:00 to 13:00, 9:00 to 18:30, and 13:00 to 17:30, respectively. Therefore, we have $\omega_{\alpha_1} = 1$ and $\varpi_{\alpha_1} = 10$ for activity α_1, $\omega_{\alpha_2} = 3$ and $\varpi_{\alpha_2} = 18$ for activity α_2, $\omega_{\alpha_3} = 7$ and $\varpi_{\alpha_3} = 12$ for activity α_3, and $\omega_{\alpha_4} = 11$ and $\varpi_{\alpha_4} = 18$ for activity α_4. The earliest start time of each activity should be $\mathcal{T}^*_1 = \omega_{\alpha_1} = 1$, $\mathcal{T}^*_2 = \max\{\mathcal{T}^*_1 + 4, \omega_{\alpha_2}\} = 5$, $\mathcal{T}^*_3 = \max\{\mathcal{T}^*_2 + 4, \omega_{\alpha_3}\} = 9$, and $\mathcal{T}^*_4 = \max\{\mathcal{T}^*_3 + 2, \omega_{\alpha_4}\} = 11$. The earliest start time of each activity satisfies $\mathcal{T}^*_1 \leq \varpi_{\alpha_1} - 4 + 1 = 7$, $\mathcal{T}^*_2 \leq \varpi_{\alpha_2} - 4 + 1 = 15$, $\mathcal{T}^*_3 \leq \varpi_{\alpha_3} - 2 + 1 = 11$, and $\mathcal{T}^*_4 \leq \varpi_{\alpha_4} - 4 + 1 = 15$, and thus the *activity sequence* is feasible and S' is an *inspection template*.□

Proposition 2: Given a combination of $L = 1, 2, 3$ ships denoted by $\bar{S}$, if it is not an *inspection template*, any set of $L+1$ ships (denoted by $\hat{S}$) containing all ships in $\bar{S}$, i.e. $\bar{S} \subset \hat{S}$ cannot be an *inspection template*.

Proposition 3: Given an *inspection template* containing $L = 2, 3, 4$ ships denoted by $\bar{S}$, all subsets of $\bar{S}$ containing $L' = L - 1$ ships are *inspection templates*.

Proposition 2and **Proposition 3**are the basis of *inspection template* construction. They are intuitive and thus we omit their proof. Based on the two propositions, the following two properties of *inspection templates* can be derived to reduce the trials and the total number of generated *inspection templates*.

Property 1: If there is a combination with $L = 1$ ship associated with berthing period smaller than 4 time units or from time unit 8 to time unit 11, or with zero predicted deficiency number, we can simply ignore it as it cannot be inspected during its berthing period or inspecting the ship will not bring benefits.

Property 2: Candidate $L \geq 2$ *inspection templates* can be formulated by combining all pairs of $L - 1$ *inspection templates* with the first $L - 2$ items the same.

Property 1 and **Property 2** can highly improve the efficiency of *inspection templates* generation. The overall procedure to generate the set of all *inspection templates* (denoted by H) is shown in Procedure 1.

After obtaining H and δ^η_s, $s \in S$, $\eta \in$ H by executingProcedure 1, the estimated number of deficiencies that can be detected in *inspection template*η is $D_\eta = \sum_{s \in S} \delta^\eta_s d_s$, $\eta \in$ H. To assign the *inspection templates* to the PSCOs, we introduce binary decision variable z_η which is set to 1 if *inspection template* $\eta \in$ H is adopted and 0, otherwise. The PSCO scheduling problem aiming to maximize the total number of detected deficiencies based on *inspection templates* can be formulated by mathematical model M2.

[M2]

$$\max \sum_{\eta \in \mathrm{H}} D_\eta z_\eta \tag{25}$$

s.t.

$$\sum_{\eta \in \mathrm{H}} z_\eta \leq |P| \tag{26}$$

$$\sum_{\eta \in \mathrm{H}} \delta^\eta_s z_\eta \leq 1, \forall s \in S \tag{27}$$

$$z_\eta \in \{0, 1\}, \forall \eta \in \mathrm{H}. \tag{28}$$

Objective function (25) maximizes the estimated total number of deficiencies that can be detected. Constraint (26) ensures that the total number of adopted *inspection templates* should be no more than the total number of PSCOs. Constraints (27) guarantee that each ship can only be inspected at most once.

5.3. PSCO scheduling model M3

Model M2 considers all the *inspection templates* in H indifferently, which is time-consuming when |H| is large. Meanwhile, it is noted that if we reformulate constraints (27) which require that a ship can only be inspected at most once, we can only consider the *inspection templates* that are not contained in any other *inspection template(s)*, which we denote by *un-dominated inspection templates*, as inspecting them can always detect more deficiencies than inspecting the *inspection templates* contained in them according to **Property 1**. In this way, the number of *inspection templates* considered in

the PSCO scheduling optimization model can be reduced largely. However, one problem is that the number of deficiencies of one ship might be calculated several times in the objective function of M2 as it can be contained in several *inspection templates* selected by a solution. To overcome this issue, we further introduce binary decision variables ξ_s, $s \in S$ which is set to 1 if ship s is inspected and 0, otherwise. In addition, we form set $\tilde{\mathrm{H}}$ which contains all *un-dominated inspection templates* using **Procedure 1** by adding only the *inspection templates* with $L = 1, 2, 3$ that cannot be further combined with others to generate larger valid *inspection templates* and all the *inspection templates* with $L = 4$. Mathematical model M3 is developed based on *inspection template* set $\tilde{\mathrm{H}}$ and decision variables ξ_s, $s \in S$ and z_η, $\eta \in \mathrm{H}$ to reduce the number of *inspection templates* considered in the master problem as follows.

[M3]

$$\max \sum_{s \in S} d_s \xi_s \tag{29}$$

s.t.

$$\xi_s \le \sum_{\tilde{\eta} \in \tilde{\mathrm{H}}} \delta_s^{\tilde{\eta}} z_{\tilde{\eta}}, \forall s \in S \tag{30}$$

$$\sum_{\tilde{\eta} \in \tilde{\mathrm{H}}} z_{\tilde{\eta}} \le |P| \tag{31}$$

$$z_{\tilde{\eta}} \in \{0, 1\}, \forall \tilde{\eta} \in \tilde{\mathrm{H}} \tag{32}$$

$$\xi_s \in \{0, 1\}, \forall s \in S. \tag{33}$$

Like Eq. (25), objective function (29) also maximizes the total estimated number of deficiencies that can be detected. Constraints (30) indicate the relationship between $\xi_s$and $z_{\tilde{\eta}}$. Constraints (31) require the maximum number of *un-dominated inspection templates* that can be selected. Constraints (32) and (33) guarantee the domain of the decision variables. It should be noted that although M3 does not require that each ship can only be inspected at most once, the objective function only calculates its estimated deficiency number once it is inspected and thus model M3 is equivalent to model M2.

To further improve the efficiency of model M3, we propose the following proposition:

Proposition 4: For two ships s_1 and s_2, if $d_{s_1} > d_{s_2}$ and $\{e_\tau^{s_2} | e_\tau^{s_2} = 1, \forall \tau \in \mathrm{T}\} \subseteq \{e_\tau^{s_1} | e_\tau^{s_1} = 1, \forall \tau \in \mathrm{T}\}$, i.e. ship s_1 has larger estimated number of deficiencies than ship s_2 and the set of berthing period of ship s_2 is a sub-set of that of ship s_1 (we denote the relationship between s_1 and s_2 by "s_1 dominates s_2"), we must have $\xi_{s_1} \ge \xi_{s_2}$ in an optimal solution.

Proof:

Consider two ships s_1 and s_2 with $d_{s_1} > d_{s_2}$ and $\{e_\tau^{s_2} | e_\tau^{s_2} = 1, \forall \tau \in \mathrm{T}\} \subseteq \{e_\tau^{s_1} | e_\tau^{s_1} = 1, \forall \tau \in \mathrm{T}\}$, i.e. s_1 dominates s_2. If an optimal solution chooses a set of ships S' for inspection, there can be several situations regarding ships $s_1$and s_2:

Situation 1: if $s_1 \in S'$ and $s_2 \in S'$, $\xi_{s_1} \ge \xi_{s_2}$ is satisfied.

Situation 2: if $s_1 \notin S'$ and $s_2 \notin S'$, $\xi_{s_1} \ge \xi_{s_2}$ is satisfied.

Situation 3: if $s_1 \in S'$ and $s_2 \notin S'$, $\xi_{s_1} \ge \xi_{s_2}$ is satisfied.

Situation 4: if $s_1 \notin S'$ and $s_2 \in S'$, we can expect that another feasible set of ships $\tilde{S}'$ formulated by substituting ship s_2 by s_1 in S' can increase the value of the objective function by $d_{s_1} - d_{s_2}$ and thus S' should not be an optimal solution, which is contradictory to the given conditions. Therefore, Situation 4 cannot be a case in any optimal solution, and $\xi_{s_1} \ge \xi_{s_2}$ can always be satisfied in the optimal solution(s).□

To incorporate **Proposition 4** into model M3, we introduce a binary matrix $B = [b_{s's''}]_{|S| \times |S|}$ which can be derived directly from ship visiting information and deficiency condition to indicate whether ship $s' \in S$ dominates $s'' \in S$. If s'dominates s'', we set $b_{s's''} = 1$; otherwise, $b_{s's''} = 0$. Especially, we require $b_{s's''} = 0$ if $s' = s''$. The following strengthened constraints based on B can be added to M3 to improve its efficiency:

$$\xi_{s'} - \xi_{s''} \ge b_{s's''} - 1, s' \in S, s'' \in S. \tag{34}$$

6. Computational experiments

We take the port of Hong Kong as an example to validate the proposed PSCO scheduling models M1, M2, and M3. Particularly, we first compare the computing performance of the three models in section 6.1. Then, comparisons between the current and proposed PSCO scheduling models are conducted in section 6.2. In section 6.3, results of extensive sensitivity analysis are presented to further validate the proposed models.

6.1. Comparison of computing performance of M1, M2, and M3

To compare the computing performance of M1, M2, and M3 (including generation of all *inspection templates, un-dominated inspection templates*, and binary matrix $B = [b_{s's''}]_{|S| \times |S|}$), we set the number of PSCOs to 4, 6, 8 and 10, and the number of ships that need to be inspected to 30, 40, 50, and 60 and combine them one by one in several scenarios. Ships for

Table 9
Comparison of computing performance of M1, M2, and M3.

No. of PSCOs	Scenarios	Model	Number of ships			
			30	40	50	60
4	Average total computation time*	M1	5.75	7.80	10.18	13.35
		M2	0.48	2.30	9.11	25.47
		M3	0.46	2.09	7.95	23.89
	Standard deviation of computation time*	M1	4.21	2.69	7.98	7.51
		M2	0.18	0.74	3.52	8.65
		M3	0.16	0.64	2.87	8.01
	The number of *inspection templates* in H	M2	1883.0	5465.8	13834.8	26972.7
	The number of *un-dominated inspection templates* in $\tilde{H}$	M3	1189.6	3871.7	10456.4	21226.0
	$(\lvert\tilde{H}\rvert - \lvert H\rvert)/\lvert H\rvert \times 100\%$	\	36.82%	29.16%	24.42%	21.31%
	Average improvement of M1/M2/M3 over random scheduling case	\	22.10%	34.57%	35.56%	43.72%
	Average gap between M1/M2/M3 and the perfect-forecast policy	\	9.64%	11.24%	16.28%	17.52%
6	Average total computation time	M1	19.16	30.85	36.26	47.98
		M2	0.47	2.32	8.58	25.06
		M3	0.46	2.07	7.83	23.25
	Standard deviation of computation time	M1	10.74	13.49	23.90	28.24
		M2	0.18	0.70	3.37	8.13
		M3	0.15	0.63	3.16	6.61
	The number of *inspection templates* in H	M2	1883.0	5465.8	13834.8	26972.7
	The number of *un-dominated inspection templates* in $\tilde{H}$	M3	1189.6	3871.7	10456.4	21226.0
	$(\lvert\tilde{H}\rvert - \lvert H\rvert)/\lvert H\rvert \times 100\%$	\	36.82%	29.16%	24.42%	21.31%
	Average improvement of M1/M2/M3 over random scheduling case	\	13.93%	20.17%	24.57%	29.81%
	Average gap between M1/M2/M3 and the perfect-forecast policy	\	5.88%	7.83%	10.90%	12.95%
8	Average total computation time	M1	31.97	64.18	102.95	195.14
		M2	0.52	2.21	8.22	24.78
		M3	0.49	2.00	7.66	24.04
	Standard deviation of computation time	M1	57.54	48.71	70.15	257.11
		M2	0.23	0.67	3.08	7.39
		M3	0.23	0.62	2.83	8.03
	The number of *inspection templates* in H	M2	1883.0	5465.8	13834.8	26972.7
	The number of *un-dominated inspection templates* in $\tilde{H}$	M3	1189.6	3871.7	10456.4	21226.0
	$(\lvert\tilde{H}\rvert - \lvert H\rvert)/\lvert H\rvert \times 100\%$	\	36.82%	29.16%	24.42%	21.31%
	Average improvement of M1/M2/M3 over random scheduling case	\	6.04%	13.74%	18.32%	24.73%
	Average gap between M1/M2/M3 and the perfect-forecast policy	\	4.19%	5.94%	7.52%	8.97%
10	Average total computation time	M1	89.53	614.45	295.13	377.39
		M2	0.49	2.24	8.53	25.22
		M3	0.43	2.05	7.46	22.36
	Standard deviation of computation time	M1	195.82	1492.04	340.93	366.52
		M2	0.19	0.67	3.39	7.80
		M3	0.17	0.64	2.73	6.44
	The number of *inspection templates* in H	M2	1883.0	5465.8	13834.8	26972.7
	The number of *un-dominated inspection templates* in $\tilde{H}$	M3	1189.6	3871.7	10456.4	21226.0
	$(\lvert\tilde{H}\rvert - \lvert H\rvert)/\lvert H\rvert \times 100\%$	\	36.82%	29.16%	24.42%	21.31%
	Average improvement of M1/M2/M3 over random scheduling case	\	1.21%	8.07%	13.90%	18.55%
	Average gap between M1/M2/M3 and the perfect-forecast policy	\	1.40%	3.94%	5.74%	6.47%

Note*: the computation time of M2 includes the time to generate all *inspection templates*, and the computation time of M3 includes the time to generate matrix *B* and all *un-dominated inspection templates*.

inspection are selected from test set i and the number of deficiencies of them is predicted by the XGBoost model developed in Section 3. We assume that a ship can arrive at a port at any time during a day, and their staying period ranges from 0 to 18 consecutive time units from 8:00 to 17:00. As all the PSCO scheduling models M1, M2, and M3 are integer linear programming (ILP) models, they are solved by the off-the-shelf optimization solver CPLEX. In addition, we compare the performance of PSCO scheduling decisions generated by M1, M2, and M3 with the current greedy PSCO scheduling strategy applied at the Hong Kong port, whose detailed description is presented in Appendix D. We call the current scheduling strategy "random scheduling case", and it aims to assign as many ships as possible to each available PSCO for inspection in a greedy manner. Besides, we present the performance of the proposed PSCO scheduling model utilizing the predicted deficiency number from a perfect-foresight prediction which knows the actual deficiency number of all ships in advance (denoted by "perfect-forecast policy"). The identified deficiency number based on the perfect-foresight policy is an upper bound in theory which cannot be achieved.

All experiments are conducted on a laptop (Intel Core i7, 3.40 GHz, 16GB RAM) using programming language Python. The *inspection templates* in M2 are generated using Procedure 1, and the *un-dominated inspection templates* in M3 are generated based on Procedure 1.Table 9 summarizes the computing performance of the three models, including the average computation time (in CPU seconds), the standard deviation of computation time, the number of *inspection templates* generated, the number of *un-dominated inspection templates* generated, the reduction in percentage of the number of *un-dominated*

inspection templates compared to that of all the *inspection templates*, the average improvement of M1/M2/M3 over random scheduling case, and the average gap between M1/M2/M3 and the perfect-forecast policy in all cases of each scenario.

For the average computation time, it is indicated in Table 9 that in almost all the cases, much less time is required to solve M2 and M3 compared to the time used to solve M1, except when the number of PSCOs is 4 and the number of ships is 60. The difference of the computation time between M1 and M2/M3 becomes larger as the number of PSCOs increases. Meanwhile, the difference of the computation time between M2 and M3 shows an increasing trend when there are more visiting ships. To be more specific, when the number of PSCOs is fixed and the number of ships increases, i.e. from left to right in each row of the table, the computation time of all the three models shows an increasing trend as expected.When the number of ships is fixed and the number of PSCOs increases, i.e. from top to bottom in each column of the table, the model computation time increases faster and faster in M1. Meanwhile, there are only some minor fluctuations in the model computation time of M2 and M3. This is because the number of PSCOs has no influence on the generation of *inspection templates* and *un-dominated inspection templates*, which occupies most of the computation time of M2 and M3, respectively.

The standard deviation of model computation time of M1 is much larger than that of M2 and M3 in most of the cases listed in Table 9, and M2 is a little bit larger than M3 in most cases. Particularly, in M1, when the number of ships is fixed and the number of PSCOs increases, the standard deviation of computation time shows a rapid upward trend. There is also a general upward trend in the standard deviation of computation time when the number of ships increases while the number of PSCOs remains unchanged in M1. Meanwhile, in M2 and M3 with similar pattern, the standard deviation of computation time increases dramatically when the number of ships increases given a certain number of PSCOs. When the number of PSCOs increases with a fixed number of ships, there are many fluctuations in the standard deviation of the computation time of both M2 and M3.

When the number of visiting foreign ships increases from 30 to 60, the numbers of *inspection templates* and *un-dominated inspection templates* grow, while the difference between them decreases. On average, the number of *un-dominated inspection templates* considered in M3 is about 72% of the *inspection templates* considered in M2. In addition, M1/M2/M3 perform better than the currently implemented random PSCO scheduling strategy at the ports in all cases. When the number of PSCOs increases given a certain number of visiting ships, the advantage of M1/M2/M3 over random scheduling and the advantage of perfect-forecast policy over M1/M2/M3 are reduce. When there are more visiting ships while the number of PSCOs is fixed, both the gap between M1/M2/M3 and random scheduling and the gap between perfect-forecast policy and M1/M2/M3 increase.

To summarize, the average model computation time and its standard deviation of M2 are much smaller than those of M1 in most cases, and the average total model computation time and its standard deviation of M3 are smaller than those of M2 in most cases as shown in Table 9. Besides, model computation time of M2 and M3 is less sensitive to the increase of the number of PSCOs given a fixed number of ships, as the process to generate *inspection templates* and *un-dominated inspection templates* is not influenced by the number of PSCOs. In all scenarios, the proposed M1/M2/M3 perform better than the current random scheduling strategy, and the gap between M1/M2/M3 and the perfect-forecast policy decreases when there are more PSCOs or fewer visiting ships. We can therefore conclude that M3 is the most efficient, stable, and flexible model among M1, M2, and M3. Especially, M3 is more suitable to be applied to the ports where there are a larger number of available PSCOs or more visiting ships.

6.2. Comparison of current and the proposed PSCO scheduling strategies

We compare the performance of current PSCO scheduling strategy applied at port and the proposed models in this section. For each day, we randomly select 20 ships from test set i as the visiting ships that need to be inspected at the Hong Kong port. We further assume that the number of PSCOs on duty for that day is 3, and their daily work time is fixed as mentioned in section 3. As M1, M2 and M3 are equivalent and section 6.1 shows that M3 is more efficient than M1 and M2, the following experiments are only conducted on M3. We randomly generate 30 groups of ships from test set i in the experiment. The performance of random scheduling case (average of 100 runs), M3, and the perfect-forecast policy solved by M3 and their comparisons are presented in Table 10.

Table 10 shows that the average improvement of M3 with the prediction of XGBoost as the input over the random PSCO scheduling case is over 20%. This implies that the combination of XGBoost model for ship deficiency number prediction and the mathematical models M1/M2/M3 for PSCO scheduling can identify 20% more deficiencies than the current PSCO scheduling scheme with the same inspection resources. Besides, the gap between the proposed model and the perfect-forecast policy is about 8%, which indicates that the proposed combined model can identify about 92% of all existing deficiencies.

6.3. Sensitivity analysis

In this section, we analyze how the number of ships to be inspected, the number of available PSCOs for conducting inspection, and ship berthing duration and period will influence the performance of M3 (and M1, M2). Four groups of sensitivity analysis (SA) are performed: SA1: different numbers of ships for inspection; SA2: different numbers of available PSCOs; SA3: different berthing durations of ships; SA4: different berthing periods of ships. In each group of SA, the number of deficiencies identified is calculated based on 10 runs.

Table 10
Performance and comparison of PSCO scheduling models.

Group	Actual identified deficiency number of random scheduling case	Actual identified deficiency number of M3	Identified deficiency number under perfect-forecast policy as solved by M3	Improvement of M3 over random scheduling case	Gap between M3 and the perfect-forecast policy
1	16	24	30	46.6%	20.0%
2	59	68	71	15.4%	4.2%
3	36	37	43	4.0%	14.0%
4	46	74	78	61.8%	5.1%
5	56	57	65	2.3%	12.3%
6	53	58	62	10.4%	6.5%
7	27	41	41	51.5%	0.0%
8	65	73	78	12.6%	6.4%
9	55	71	72	29.6%	1.4%
10	37	41	51	9.9%	19.6%
11	43	57	65	33.8%	12.3%
12	17	25	31	46.6%	19.4%
13	59	75	79	27.0%	5.1%
14	42	47	63	12.5%	25.4%
15	42	54	56	29.6%	3.6%
16	60	69	75	15.2%	8.0%
17	59	66	66	11.6%	0.0%
18	41	48	55	15.9%	12.7%
19	39	55	59	39.8%	6.8%
20	55	66	67	20.8%	1.5%
21	61	64	68	5.4%	5.9%
22	46	49	54	7.2%	9.3%
23	34	48	49	42.5%	2.0%
24	33	45	46	36.2%	2.2%
25	32	36	37	11.6%	2.7%
26	63	77	78	22.1%	1.3%
27	29	39	47	35.4%	17.0%
28	47	55	58	18.3%	5.2%
29	51	56	65	9.5%	13.8%
30	50	61	72	22.2%	15.3%
Average	**45.0067**	**54.5333**	**59.3667**	**21.2%**	**8.1%**

Table 11
Performance of the groups in SA1.

Group	SA1G1	SA1G2	SA1G3	SA1G4	SA1G5	SA1G6	SA1G7	SA1G8
Number of ships	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**
Random scheduling case	34.1	44.4	48.3	52.9	54.5	55.7	55.9	57.4
M3	41.2	54.4	58.0	66.8	71.5	78.8	81.2	84.0
Perfect-foresight policy	44.3	59.1	66.1	76.6	82.9	91.4	97.7	104.8
Superiority of M3 over random scheduling case	20.9%	22.5%	20.0%	26.2%	31.2%	41.4%	45.3%	46.3%
Gap between M3 andthe perfect-foresight policy	7.5%	8.6%	14.0%	14.7%	15.9%	16.0%	20.3%	24.8%

6.3.1. SA1: different numbers of ships for inspection

First, we analyze how the number of ships that need to be inspected would influence the performance of M3. We set the number of ships to 15, 20, 25, 30, 35, 40, 45, and 50, respectively while fixing the number of PSCOs to 3 in SA1G1 to SA1G8. The performance of random scheduling case (based on 100 runs), M3, and the perfect-foresight policy and their comparison are presented in Table 11.

Table 11 shows that when the number of ships increases from 15 to 50 while the number of PSCOs remains unchanged, the numbers of deficiencies identified in random scheduling case, M3, and the perfect-foresight policy increase. This can be explained as follows. In random scheduling case which aims to assign as many ships to each PSCO as possible, a larger number of ships can be inspected by one PSCO as the total number of visiting ships increases. For M3 and the perfect-foresight policy, although the inspection resources are fixed, more ships with larger number of deficiencies can be selected for inspection when the total number of visiting ships grows. Meanwhile, Table 11 also indicates that both the superiority of M3 over random scheduling case and the gap between M3 and the perfect-foresight policy show an increasing trend. This is because as the perfect-foresight policy can capture the ships with more deficiencies more efficiently than M3, the gap between them became larger as the number of visiting ships increases. This explanation can also be applied for the changes in the gap between M3 and random scheduling case.

Table 12
Performance of the groups in SA2.

Group	SA2G1	SA2G2	SA2G3	SA2G4
Number of PSCOs	2	3	4	5
Random scheduling case	37.4	53.1	63.9	70.5
M3	51.8	66.8	77.8	86.4
Perfect-foresight policy	61.6	76.6	86.1	92.9
Superiority of M3 over random scheduling case	38.5%	25.8%	21.8%	22.6%
Gap between M3 andthe perfect-foresight policy	18.9%	14.7%	10.7%	7.5%

Table 13
Performance of the groups in SA3.

Group	SA3G1	SA3G2	SA3G3	SA3G4	SA3G5	SA3G6	SA3G7	SA3G8
Berthing duration of all ships	2 hours	3 hours	4 hours	5 hours	6 hours	7 hours	8 hours	9 hours
Random scheduling case	38.3	42.8	48.4	50.7	52.2	45.9	48.8	51.3
M3	58.9	62.6	66.4	72.4	74.4	72.0	74.2	75.0
Perfect-foresight policy	71.8	82.9	89.3	94.5	96.0	90.7	95.1	95.0
Superiority of M3 over random scheduling case	53.6%	46.1%	37.2%	42.9%	42.4%	56.8%	52.0%	46.1%
Gap between M3 andthe perfect-foresight policy	21.9%	32.4%	34.5%	30.5%	29.0%	26.0%	28.2%	26.7%

6.3.2. SA2: different numbers of available PSCOs

Second, we analyze how the number of available PSCOs to carry out PSC inspection would influence the performance of M3. We set the number of ships for inspection to 30, and the number of PSCOs to 2, 3, 4, and 5 in SA2G1 to SA2G4, respectively. The performance of random scheduling case (based on 100 runs), M3, and the perfect-foresight policy and their comparison are presented in Table 12.

Table 12 indicates that when the number of ships that need to be inspected remains to be 30 while the number of PSCOs increases from 2 to 5, the total number of deficiencies that can be detected grows as expected. In addition, both the superiority of M3 over random scheduling case and the gap between M3 and the perfect-foresight policy show a decreasing trend. Particularly, such decreasing trend is more obvious in the gap between M3 and the perfect-foresight policy. This can be explained by the fact that as the number of available PSCOs increases, more ships can be assigned for inspection and thus to reduce the superiority of models with better performance as more ships with large number of deficiencies can be captured. Especially, for M3 which is based on the prediction given by XGBoost, more ships with larger real deficiency number can be captured for inspection although the XGBoost model is not perfect. As a consequence, the gap between M3 and the perfect-foresight policy gets closer more quickly than the superiority of M3 over random scheduling case as the number of inspected ships grows.

6.3.3. SA3: different berthing durations of ships

Third, we analyze model performance when the berthing duration of ships varies. We assume that the number of ships for inspection is 30 and the number of available PSCOs is 3. As only when a ship berths at a port for no less than two hours can the ship be inspected, we consider eight groups where the berthing duration of all ships is 2, 3, …., 8, 9 hours respectively denoted by SA3G1 to SA3G8. The consecutive berthing time units are randomly generated for all ships in each group. The performance of random scheduling case (based on 100 runs), M3, and the perfect-foresight policy and their comparison are presented in Table 13.

Table 13 shows that as the berthing duration of all ships increases, the total number of deficiencies detected also shows an upward trend although there are fluctuations due to the randomness in ship conditions. Meanwhile, there is no obvious pattern in the change of the gap between random scheduling case and M3 and the gap between M3 and the perfect-foresight policy when ship berthing duration increases. The superiority of M3 over random scheduling case is maximized at 56.8% when the berthing duration of all ships is 7 hours. The gap between M3 and the perfect-foresight policy is maximized at 34.5% when the berthing duration of all ships is 4 hours.

6.3.4. SA4: different berthing periods of ships

Fourth, we analyze how ship berthing period (during the work time of PSCOs) can influence the model performance. We set the number of ships for inspection to be 30 and the number of available PSCOs to be 3. We consider four groups of berthing periods as denoted by SA4G1 to SA4G4, respectively. In SA4G1, the berthing period of all ships is only in the morning (from 8:00 to 12:30). In SA4G2, the berthing period of all ships is only in the afternoon (from 12:30 to 17:00). In SA4G3, the berthing period of one-third of the ships is in the morning, in the afternoon, and both in the morning and in the afternoon, respectively. In SA4G4, the berthing period of half of the ships is in the morning and the other half is in the afternoon. The berthing duration is randomly generated for all ships. The performance and comparison of random scheduling case (based on 100 runs), M3, and the perfect-foresight policy are presented in Table 14.

Table 14 shows that the number of deficiencies identified is smaller when there is more overlap in ship berthing period (i.e. in SA4G1 and SA4G2) than less overlap (i.e. in SA4G3 and SA4G4). Meanwhile, the average gaps between random

Table 14
Performance of the groups in SA4.

Group	SA4G1	SA4G2	SA4G3	SA4G4
Distribution of berthing period	All ships in the morning	All ships in the afternoon	1/3 ships in the morning, 1/3 ships in the afternoon, and 1/3 ships in the morning and afternoon	1/2 ships in the morning and 1/2 ships in the afternoon
Random scheduling case	23.1	22.2	47.1	50.5
M3	41.4	40.8	69.5	66.2
Perfect-foresight policy	61.1	59.1	88.2	87.1
Superiority of M3 over random scheduling case	79.4%	83.5%	47.7%	31.0%
Gap between M3 and the perfect-foresight policy	47.6%	44.9%	26.9%	31.6%

Procedure 1.
Generation of the set of *inspection templates* H.

Input: the set of foreign ships that need to be inspected S, duration of a PSC inspection, ship berthing information e_τ^s, $s \in S$, $\tau = 1, ..., T$, the duration and period of PSCO lunch break, the total number of time units T.

Output: the set of all feasible *inspection templates* H, binary parameter δ_s^η, $s \in S$, $\eta \in H$ indicating whether ship s is contained in *inspection template* η.
Initialize $H = \emptyset$, δ_s^η, $s \in S$, $\eta \in H$.
for $L = 0, 1, 2, 3, 4$ do if $L < 2$: Formulate all combinations containing L ships that can be inspected based on **Property 1** from S denoted by Q.
else: Formulate the combinations containing L ships as denoted by Q such that each combination contains two items of $L - 1$ ships from H and they have the same $L - 2$ ship based on **Property 2**.
end if
for each combination $\hat{Q} \in Q$ do Initialize feasibility = False.
Formulate set Q that contains all permutations (i.e. *activity sequences*) of the activities of inspecting the ships in $\hat{Q}$ and having lunch break.
for each *activity sequence*$q \in Q$ do
Test the feasibility of q using **Proposition 1**.
if q is feasible:
Add $\hat{Q}$ to H by updating $H = H \cup \hat{Q}$. Update parameter $\delta_s^\eta = 1$, $s \in \hat{Q}$, $\eta \in H$.
Update feasibility = True.
Break
else:
Continue
end if
end for
if feasibility = True:
Break
else:
Continue
end for
end for
Return H and δ_s^η.

scheduling case and M3 as well as between M3 and the perfect-foresight policy in SA4G1 and SA4G2 are much larger than those in SA4G3 and SA4G4. This is also because more ships can be inspected when their berthing period is more scattered, which would reduce the superiority of prediction models with better performance.

7. Conclusion

PSC inspection is a safeguard of maritime safety, the marine environment, and the rights of seafarers. To improve ship selection efficiency, this study first proposes an accurate XGBoost model to predict ship deficiency number. Particularly, domain knowledge regarding ship flag, RO, and company performance is considered in the XGBoost model, which improves its accuracy and fairness. Based on the predictions, an initial PSCO scheduling model is proposed to assign the PSCOs to inspect the predicted high-risk ships which also considers the number of available PSCOs and their work and rest time. To reduce problem size and improve model computation efficiency and flexibility, concepts of *inspection template* and *undominated inspection template* are further proposed and incorporated in the PSCO scheduling models.

In numerical experiments, we use the real PSC inspection records at the Hong Kong port from January 2016 to December 2018 as the case dataset to construct and validate the proposed models. Numerical experiments show that the MSE and MAE of the XGBoost model is 12.5 and 2.4 in the test set, respectively, which are better than the other popular machine learning models compared in this study. Moreover,when ship flag performance gets worse from white to grey and from

grey to black, 0.8 and 0.2 more deficiency will be detected on average, respectively. When RO performance gets worse from high to medium, 0.3 more deficiency will be detected on average. When company performance gets worse from high to medium, from medium to low, and from low to very low, 0.5, 0.8, and 1.5 more deficiencies will be detected on average, respectively. When combining the predictions with PSCO scheduling models, it is shown that the superiority of the proposed PSCO scheduling models over the current inspection scheme regarding the number of deficiencies identified is more than 20%. The gap between the proposed model and the model under perfect-forecast policy is about 8% regarding the number of deficiencies identified. Meanwhile, computation efficiency and flexibility of the PSCO scheduling model with *inspection templates* are higher than the initial PSCO scheduling model. Problem size can be reduced and the computation efficiency can be further improved in the PSCO scheduling model which takes *un-dominated inspection templates* and the relationship between each of the two ships into consideration. Extensive sensitivity analysis shows that when changing the numbers of ships for inspection, the numbers of available PSCOs, the berthing durations of ships, and the berthing periods of ships, the performance of the proposed PSCO scheduling model is stable and it is always better than the current model used at ports.

This study addresses an important practical problem in maritime industry. Theoretically, it proposes the first ship risk prediction model for PSC inspection considering domain knowledge. It also develops the first PSCO scheduling models based on the predictions to efficiently allocate scarce inspection resources for ship inspection. Moreover, the concepts of *inspection template* and *un-dominated inspection template* are proposed and incorporated in the PSCO scheduling model to improve computation efficiency and model flexibility. Practically, it helps port states to identify high-risk ships and assign the PSCOs more efficiently. Therefore, the main objectives of PSC to eliminate substandard shipping and safeguard the sea can be enhanced.

Declaration of Competing Interest

No.

Acknowledgement

The authors thank the editors and the anonymous referees for their constructive comments and suggestions. This study is supported by the Policy Innovation and Co-ordination Office (PICO) of the Government of the HKSAR (Project number: 2020.A6.148.20A).

Author statement

Ran Yan: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Visualization, Writing – original draft. **Shuaian Wang:** Supervision, Conceptualization, Methodology, Writing - review & editing, Project administration, Funding acquisition. **Jiannong Cao:** Conceptualization, Writing - review & editing, Supervision. **Defeng Sun**: Conceptualization, Writing - review & editing, Supervision.

Appendix A. Introduction ofprediction target and features

Table A1

Table A1
Variable explanation, encoding method, and descriptive statistics.

Variable name	Explanation	Encoding	Mean value	Min value	Max value
deficiency number	The number of deficiencies identified in the current PSC initial inspection.	No encoding	4.31	0	51
age	The time interval (in years) between the keel laid date and the current PSC inspection date.	No encoding	10.8	0	47
GT	A nonlinear measure of a ship's internal volume, with 100 cubic feet as the unit.	No encoding	44,908	497	266,681
length	The overall maximum length of a ship (in meters).	No encoding	214.88	32.29	400
depth	The vertical distance (in meters) measured from the top of the keel to the upper deck at side measured inside the plating.	No encoding	17.79	4.28	36.02
beam	The width of ship hull (in meters).	No encoding	31.93	7.38	60.05

(*continued on next page*)

Table A1 (*continued*)

type	Ships in the dataset are classified into the following types: bulk carrier, container ship, general cargo/multipurpose, passenger ship, tanker, and other.	One-hot encoding: is_bulk_carrier: 1 for bulk carrier and 0, otherwise; is_container_ship: 1 for container ship and 0, otherwise; is_ general cargo/multipurpose: 1 for general cargo/multipurpose and 0, otherwise; is_ passenger_ship: 1 for passenger ship and 0, otherwise; is_tanker: 1 for tanker and 0, otherwise; is_other: 1 for other ship types and 0, otherwise.	\	\	\
flag performance	Ship flag performance is calculated based on the flag Black-Grey-White list provided by Tokyo MoU (Tokyo MoU, 2018).	Label encoding: white->1*; grey->2; black->3.	\	\	\
RO performance	Ship RO performance is calculated based on RO performance list provided by Tokyo MoU (Tokyo MoU, 2018).	Label encoding: high->1; medium->2; low->3.	\	\	\
company performance	Ship company performance is calculated based on company performance matrix provided by Tokyo MoU (Tokyo MoU, 2018)	Label encoding: high->1; medium->2; low->3; very low->4.	\	\	\
last inspection date	The time interval between the last and current PSC initial inspections within Tokyo MoU (in months). For ships that are inspected for the first time (i.e. with no previous inspection records), the state of this variable is set to be "−1".	No encoding.	10.2	0	180.7
last deficiency number	The number of deficiencies identified in last PSC initial inspection within Tokyo MoU. For ships that are inspected for the first time, the state of this variable is set to be "−1".	No encoding.	2.46	0	38
total detentions	The total number of detentions of a ship in all previous PSC inspections since the keel laid date.	No encoding.	0.59	0	18
the number of flag changes	The total number of times of ship flag change from keel laid date to the current PSC inspection date.	No encoding.	0.66	0	8
casualty in last 5 years	A binary variable indicating whether a ship was involved in casualties in the last five years.	One-hot encoding: casualty-in-5-years: 1 for any casualty occurs in the last 5 years and 0, otherwise.	\	\	\

Note *: this indicates that the state of "white" is encoded to be "1".

Appendix B. Detailed construction process of a XGBoost model

The detailed procedure of constructing a XGBoost model is as follows (Chen, 2014; Chen and Guestrin, 2016). Given a dataset with n samples and m features, denoted by $D = \{(\mathbf{x}_i, y_i), i = 1, ..., n\}$, $\mathbf{x}_i \in R^m$, $y_i \in R$, a tree ensemble model uses K additive functions to predict the target y_i (the predicted value is denoted by $\hat{y}_i$) is

$$\hat{y}_i = \phi(\mathbf{x}_i) = \sum_{k=1}^{K} f_k(\mathbf{x}_i),\ f_k \in F, \tag{B1}$$

where$F$ is a space of functions that contains all CART based regression trees. In XGBoost, the learning objective function to be minimized, which aims to draw a balance between model accuracy and complexity, is as follows:

$$\overline{obj} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k). \tag{B2}$$

The first term in Eq. (B2) is the training loss regarding all training samples, and the second term is the tree complexity. In regression problems, a common choice for the training loss function is half of the MSE, which is given by

$$\sum_{i=1}^{n} l(y_i, \hat{y}_i) = \frac{1}{2}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \tag{B3}$$

where the multiplication of 1/2 is for the ease of calculation. Eq. (B3) is also the loss function used in this study. As XGBoost is developed based on additive training, the prediction value after finishing 0 to $t = 1, ..., K$ iterations can be written as

$$\begin{aligned} \hat{y}_i^{(0)} &= 0, \\ \hat{y}_i^{(1)} &= f_1(\mathbf{x}_i) = \hat{y}_i^{(0)} + f_1(\mathbf{x}_i), \\ \hat{y}_i^{(2)} &= f_1(\mathbf{x}_i) + f_2(\mathbf{x}_i) = \hat{y}_i^{(1)} + f_2(\mathbf{x}_i), \\ &\cdots \\ \hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(\mathbf{x}_i) = \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i). \end{aligned} \tag{B4}$$

By combining Eqs. (B2) to (B4), the objective function in the tth iteration can be given by

$$\begin{aligned} \overline{obj}^{(t)} &= \tfrac{1}{2}\sum_{i=1}^{n} (y_i - \hat{y}_i^{(t)})^2 + \sum_{k=1}^{t} \Omega(f_k) \\ &= \tfrac{1}{2}\sum_{i=1}^{n} (y_i - (\hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)))^2 + \Omega(f_t) + \sum_{k=1}^{t-1} \Omega(f_k). \end{aligned} \tag{B5}$$

Eq. (B5) contains three terms. The first term is the loss function of the tth iteration. The second term is the penalty for tree complexity in the tth iteration. The last term is the sum of penalties for tree complexity of all the first $t-1$ iterations. Define $g_i^t = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ and $h_i^t = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ as the first and second order gradients of the loss function in Eq. (B5). The concrete expressions for g_i^t and h_i^t can be given if the loss function is explicitly defined. As we choose Eq. (B3) as the loss function in this study, we can have $g_i^t = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) = \hat{y}_i^{(t-1)} - y_i$ and $h_i^t = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) = 1$. It should be mentioned that the values for g_i^t and h_i^t for sample i of the tth iteration are fixed as they are only related to the output generated in the $(t-1)$th iteration. The second order Taylor expansion at $\hat{y}_i^{(t-1)}$ of Eq. (B5) should be

$$\overline{obj}^{(t)} \simeq \sum_{i=1}^{n} \left[\frac{1}{2}\sum_{i=1}^{n} (y_i - \hat{y}_i^{(t-1)})^2 + g_i^t f_t(\mathbf{x}_i) + \frac{1}{2} h_i^t f_t(\mathbf{x}_i)^2 \right] + \Omega(f_t) + \sum_{k=1}^{t-1} \Omega(f_k). \tag{B6}$$

In the first term of Eq. (B6), $\frac{1}{2}\sum_{i=1}^{n} (y_i - \hat{y}_i^{(t-1)})^2$ is the loss of the $(t-1)$th iteration and thus it is a constant. The last term of Eq. (B6), i.e. $\sum_{k=1}^{t-1} \Omega(f_k)$, is the penalty of tree complexity of all the first $t-1$ iterations and thus is also a constant. All the constants can be removed. Therefore, we represent the objective function in the tth iteration as

$$obj^{(t)} = \sum_{i=1}^{n} \left[g_i^t f_t(\mathbf{x}_i) + \frac{1}{2} h_i^t f_t(\mathbf{x}_i)^2 \right] + \Omega(f_t). \tag{B7}$$

The goal of the tth iteration is to construct a tree to minimize Eq. (B7), which requires to decide the outputs of the leaf nodes and the structure of the tree. We first assume that the tree structure is fixed and discuss the way to determine the outputs of the leaf nodes. Define a tree by a vector of outputs (which are also called weights) in leaves, and a leaf index mapping function that maps a sample to a leaf as

$$f_t(\mathbf{x}) = w^t_{q^t(\mathbf{x})}, \mathbf{w}^t \in R^{T_t}, q^t : R^m \to \{1, 2, ..., T_t\}, \tag{B8}$$

where T_t is the number of leaves in the tree, $\mathbf{w}^t$ is the vector of outputs in all the leaves, and q^t is the function assigning each sample to the corresponding leaf in the tth iteration. We use the following toy example to exemplify the notations used in Eq. (B8).

Suppose that we have a total of six samples in a toy training set, and the developed regression tree in the tth iteration is shown in Figure B1. The notations in Eq. (10) can be exemplified as follows: $T_t = 3$, $\mathbf{w}^t = \{2, -1, 0.5\}$, $q^t(\text{ship1}) = 1$, $q^t(\text{ship2}) = 1$, $q^t(\text{ship3}) = 1$, $q^t(\text{ship4}) = 2$, $q^t(\text{ship5}) = 2$, and $q^t(\text{ship6}) = 3$. It should be noted that the leaf output in XGBoost is different from the leaf output in traditional CART regression tree: the leaf output in XGBoost is calculated by optimization models whereas the leaf output in CART regression tree is simply the mean of the output of the samples in that leaf node in regression problems. The tree complexity in the objective function of XGBoost is defined as

$$\Omega(f_t) = \gamma T_t + \frac{1}{2}\lambda \sum_{j=1}^{T_t} w_j^{t\,2}, \tag{B9}$$

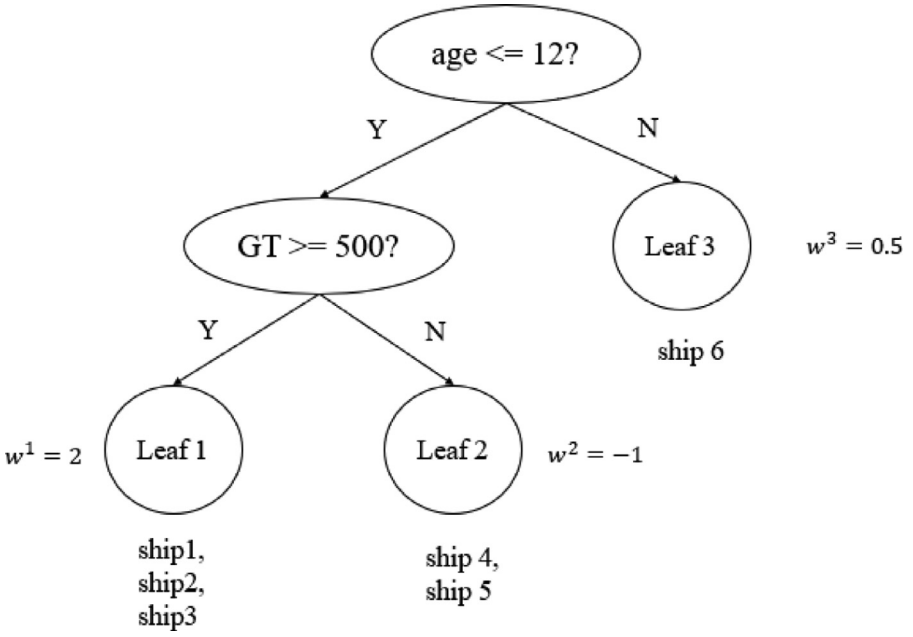

Fig. B1. . A toy regression tree in the tth iteration of a XGBoost model.

where the first term is the penalty on the total number of leaves and the second term is the penalty on the sum of squares of the weights in the leaves in the tth iteration. γ and λ are two hyperparameters that need to be tuned and are used to balance model accuracy and complexity. Define the sample set in leaf j on the tree of the tth iteration as $I_j^t = \{i|q^t(\mathbf{x}_i) = j\}$, $i = 1, ..., n$, we can regroup the objective function in Eq. (B7) by leaf and combine with Eq. (B9) to be

$$\begin{aligned} obj^{(t)} &= \sum_{i=1}^{n}\left[g_i^t f_t(\mathbf{x}_i) + \tfrac{1}{2}h_i^t f_t(\mathbf{x}_i)^2\right] + \Omega(f_t) \\ &= \sum_{i=1}^{n}\left[g_i^t w^t_{q^t(\mathbf{x}_i)} + \tfrac{1}{2}h_i^t {w^t_{q^t(\mathbf{x}_i)}}^2\right] + \gamma T_t + \tfrac{1}{2}\lambda\sum_{j=1}^{T_t} {w_j^t}^2 \\ &= \sum_{j=1}^{T_t}\left[(\sum_{i\in I_j^t} g_i^t)w_j^t + \tfrac{1}{2}(\sum_{i\in I_j^t} h_i^t + \lambda){w_j^t}^2\right] + \gamma T_t. \end{aligned} \tag{B10}$$

For simplicity, we define $G_j^t = \sum_{i\in I_j^t} g_i^t$ and $H_j^t = \sum_{i\in I_j^t} h_i^t$, Eq. (B10) can be written as

$$obj^{(t)} = \sum_{j=1}^{T_t}\left[G_j^t w_j^t + \frac{1}{2}(H_j^t + \lambda){w_j^t}^2\right] + \gamma T_t. \tag{B11}$$

As we have assumed that the tree structure (i.e. q^t) is fixed, and thus G_j^t, H_j^t, and T_t are all fixed. The optimal output w_j^t (denoted by w_j^t*) can be found by letting the first derivative of $obj^{(t)}$ with respect to w_j^t be 0, which is

$$w_j^t* = -\frac{G_j^t}{H_j^t + \lambda}. \tag{B12}$$

The optimal value of the objective function is

$$obj^{(t)*} = -\frac{1}{2}\sum_{j=1}^{T_t}\frac{{G_j^t}^2}{H_j^t + \lambda} + \gamma T_t. \tag{B13}$$

After the outputs in the tree leaves are determined by assuming the tree structure is given, the last question is how to decide the tree structure (i.e. split a node into two child nodes) in an XGBoost tree. In practice, we grow the tree in a greedy manner by splitting nodes from the tree root by enumerating all values (or quantiles of values) of all features (or a subset of features) and calculating the reduction in objective function after adding a candidate split by

$$\begin{aligned} gain &= obj_{L+R}^{(t)} - (obj_L^{(t)} + obj_R^{(t)}) \\ &= \frac{1}{2}\left[\frac{{G_L^t}^2}{H_L^t+\lambda} + \frac{{G_R^t}^2}{H_R^t+\lambda} - \frac{(G_L^t+G_R^t)^2}{H_L^t+H_R^t+\lambda}\right] - \gamma, \end{aligned} \tag{B14}$$

where $obj_{L+R}^{(t)}$, $obj_{L}^{(t)}$, and $obj_{R}^{(t)}$ are the objective functions of the node for splitting, the objective function of the left child node if adding this candidate split, and the objective function of the right child node if adding this candidate split, respectively. *gain* is calculated for each candidate split of the current node. As G_L^t and G_R^t (H_L^t and H_R^t) are the sum of first (second) derivative of the samples contained in left and right child leaf respectively, different splits would lead to different values for G_L^t and G_R^t (H_L^t and H_R^t). If $gain < 0$, the candidate split is not considered. For all positive values for *gain*, we choose the feature and value corresponding to the maximum value of *gain* to split the node as it could reach the maximum reduction of the objective function after the splitting.

Appendix C. Monotonic constraints imposed on multiple features

We also present the situation where we have more than one feature imposed by monotonic constraint. Suppose we have a total of m features, and opposite monotonic constraints are imposed on two features, namely m_1 (monotonically increasing) and m_2 (monotonically decreasing). As mentioned in Section 4.2, for each feature imposed by monotonic constraint, it works in the context that all the features (including other features imposed by monotonic constraint, if any, and normal features) are with the same value respectively except for this monotonic feature in the samples. This means that although we put monotonic properties on two features, they can be viewed as independent of each other. To be more specific, for m_1, it works in the samples with feature m_2 and all the other $m-2$ features the same; for m_2, it works in the samples where feature m_1 and all the other $m-2$ features are the same. In this way, to illustrate the splitting process using either of the two features imposed by monotonic constraint, we can simplify the tree structure to only contain the nodes using the monotonic features concerned for splitting and their child nodes. Then, we can further split the tree into two sub-trees where each tree containing nodes using either monotonic feature for splitting and their child nodes like that presented in Figure C1. We further use the following example to present the process. Suppose we have a total of $m = 3$ features, and we have a total of $n = 7$ samples as shown as follows.

For samples 1 to 4, monotonically increasing constraint on m_1 should be followed. For samples 1 and 5 to 7, monotonically decreasing constraint on m_2 should be followed. Suppose the developed tree structure as well as the splitting point and samples contained in each node is shown in Figure C1.

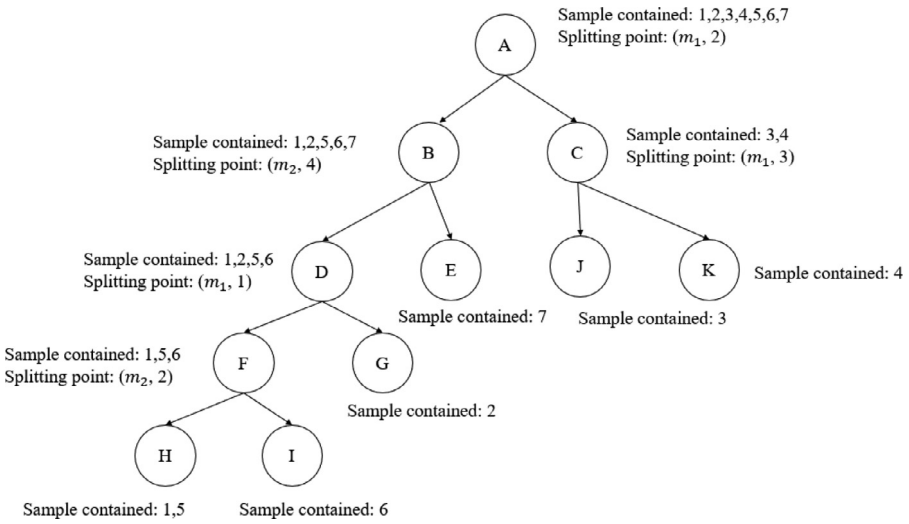

Fig. C1. Tree structure to demonstrate monotonicity.

Table C1
Samples used to show model monotonicity.

Sample ID /Feature value	m_1 (monotonically increasing)	m_2 (monotonically decreasing)	m_3 (no constraint)
1	1	1	0
2	2	1	0
3	3	1	0
4	4	1	0
5	1	2	0
6	1	3	0
7	1	4	0

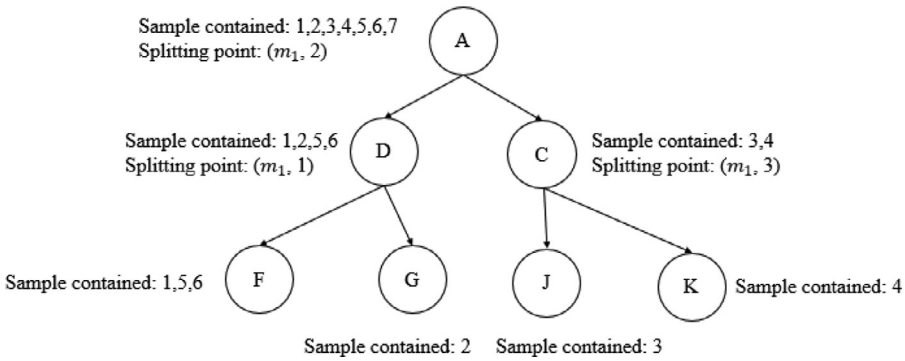

Fig. C2. Structure of sub-tree containing nodes use m_1 as splitting feature.

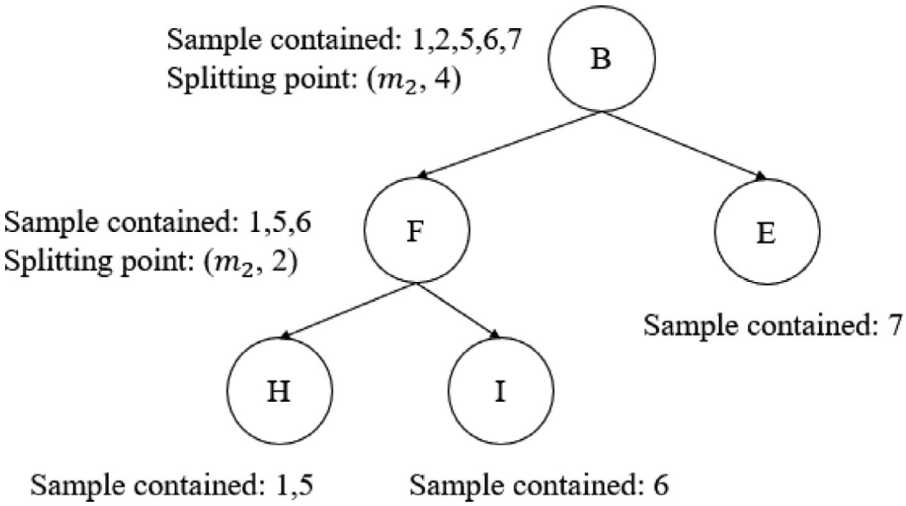

Fig. C3. Structure of sub-tree containing nodes use m_2 as splitting feature.

It is indicated in Figure C1 that as the tree grows, m_1 and m_2 are interactively used as the splitting feature. As m_1 and m_2 are independent, we can divide the developed tree into two sub-trees with each sub-tree only containing the nodes and their child nodes using m_1 or m_2 as the splitting feature as shown as Figure C2 and Figure C3.

In Figure C2, we can expect that the predicted target value of sample 1 is the smallest, followed by sample 2 and sample 3, and finally sample 4 if the monotonic constraints shown in Figure 1 are imposed. Similarly, if we apply the opposite monotonic constraints shown in Figure 1 to Figure C3, we can expect that the predicted outputs of samples 1 and 5 are the largest, followed by sample 6, and then by sample 7. If three or more features are imposed by monotonic constraints, and no matter if they are opposite, they can be processed in the above way to guarantee their respective monotonicity.

Appendix D. Current PSCO scheduling strategy

The current PSCO scheduling strategy adopted at the Hong Kong port is in a greedy manner: it aims to assign as many ships as possible to one PSCO for inspection on the morning of each workday. The set of ships assigned to one PSCO should satisfy that (a) they are berthing at the port when inspecting. (b) The PSCO can only inspect one ship in a time unit. (c) The lunch break and off work time of the PSCO should be guaranteed. Denote the number of PSCOs on duty for that day by $|P|$. The procedure of the current scheduling strategy is presented in Figure D1.

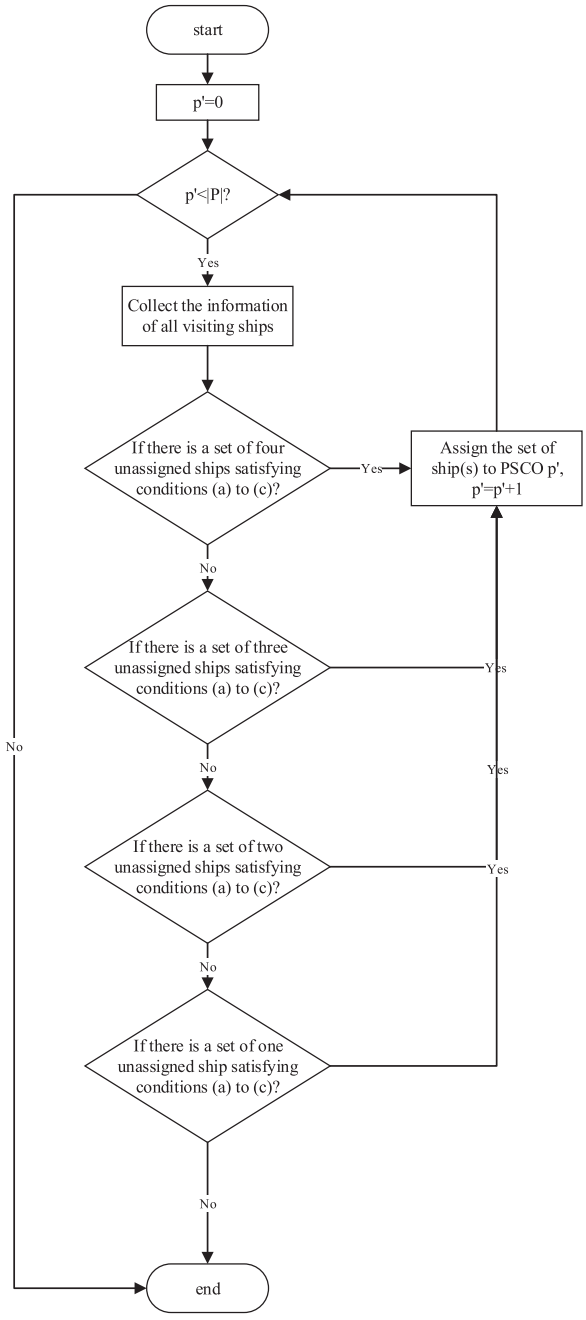

Fig. D1. Procedure of current PSCO scheduling strategy.

Reference

Akpinar, H.Sahin, B., 2020. Strategic management approach for port state control. Maritime Business Review 5 (3), 279–290.
Bell, M., Pan, J., Teye, C., Cheung, K., Perera, S., 2020. An entropy maximizing approach to the ferry network design problem. Transp. Res. Part B Methodol.132 15–28.
Breiman, L., 2001. Random forests. Mach. Learn.45 (1) 5–32.
Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A., 1984. Classification and Regression Trees. Taylor & Francis, Abingdon.
Chang, S. M.Huang, Y. Y., Shang, K. C., Chiang, W. T., 2020. Impacts of regional integration and maritime transport on trade: with special reference to RCEP. Maritime Business Review 5 (2), 143–158.
Chen, T., 2014. Introduction of boosted trees. Accessed. 14 July 2020 https://web.njit.edu/~usman/courses/cs675_fall16/BoostedTree.pdf .
Chen, T., 2016. Monotonic Constraints in Tree Construction. Accessed. https://github.com/dmlc/xgboost/issues/1514.
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.

Christodoulou, A.Gonzalez-Aregall, M., Linde, T., Vierth, I., Cullinane, K., 2019. Targeting the reduction of shipping emissions to air. Maritime Business Review 4 (1), 16–30.
Chung, W.H., Kao, S.L., Chang, C.M., Yuan, C.C., 2020. Association rule learning to improve deficiency inspection in port state control. Maritime Policy & Management 47 (3), 332–351.
Daniels, H., Velikova, M., 2010. Monotone and partially monotone neural networks. IEEE Trans. Neural Netw.21 (6) 906–917.
Degré, T., 2007. The use of risk concept to characterize and select high risk vessels for ship inspections. WMU J. Maritime Affairs 6 (1), 37–49.
Degré, T., 2008. From black-grey-white detention-based lists of flags to black-grey-white casualty-based lists of categories of vessels? J. Navigat.61 (3) 485–497.
Dinis, D., Teixeira, A.P., Soares, C.G., 2020. Probabilistic approach for characterising the static risk of ships using Bayesian networks. Reliab. Eng. Syst. Safe., 107073.
Drucker, H., Burges, C.J., Kaufman, L., Smola, A., Vapnik, V., 1996. Support vector regression machines. Adv. Neur. Inform. Process. Syst.9 155–161.
Duivesteijn, W., Feelders, A., 2008. Nearest neighbour classification with monotonicity constraints. In: proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 301–316.
Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comp. Syst. Sci.55 (1) 119–139.
Friedman, J., Hastie, T., Tibshirani, R., 2000. Additive logistic regression: A statistical view of boosting. Ann. Stat.28 (2) 337–407.
Friedman, J., Hastie, T., Tibshirani, R., 2001. The Elements of Statistical Learning. Springer Publisher, Berlin.
Fu, J., Chen, X., Wu, S., Shi, C., Wu, H., Zhao, J., Xiong, P., 2020. Mining ship deficiency correlations from historical port state control (PSC) inspection data. PLoS one 15 (2), e0229211.
Heij, C., Knapp, S., 2019. Shipping inspections, detentions, and incidents: An empirical analysis of risk dimensions. Maritime Policy & Management 46 (7), 866–883.
Hoerl, A.E., Kennard, R.W., 1970. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12 (1), 55–67.
Huisman, D., 2007. A column generation approach for the rail crew re-scheduling problem. Eur. J. Oper. Res.180 (1) 163–173.
IMO, 2017. Resolution A.1119(30): Procedure for port state control, 2017. Accessed 17 May 2019, http://www.imo.org/en/KnowledgeCentre/IndexofIMOResolutions/Assembly/Documents/A.1119%2830%29.pdf.
Janacek, J., Kohani, M., Koniorczyk, M., Marton, P., 2017. Optimization of periodic crew schedules with application of column generation method. Transp. Res. Part C Emerg. Tech.83 165–178.
Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T., 2017. Lightgbm: A highly efficient gradient boosting decision tree. Adv. Neur. Inform. Process. Syst.30 3146–3154.
Knapp, S., Heij, C., 2020. Improved strategies for the maritime industry to target vessels for inspection and to select inspection priority areas. Safety 6 (2), 1–21.
Kulkarni, S., Krishnamoorthy, M., Ranade, A., Ernst, A.T., Patil, R., 2018. A new formulation and a column generation-based heuristic for the multiple depot vehicle scheduling problem. Transp. Res. Part B Methodol.118 457–487.
Li, K.X., 1999. The safety and quality of open registers and a new approach for classifying risky ships. Transp. Res. Part E 35 (2), 135–143.
Muñoz, G., Espinoza, D., Goycoolea, M., Moreno, E., Queyranne, M., Letelier, O.R., 2018. A study of the Bienstock–Zuckerberg algorithm: Applications in mining and resource constrained project scheduling. Comput. Optim. Appl.69 (2) 501–534.
Natekin, A., Knoll, A., 2013. Gradient boosting machines, a tutorial. Front. Neurorobot.7 21.
Paris MoU, 2010. Parismemorandum of understanding on port state control Accessed 3 Dec 2019 https://www.lesiagroup.com/Resources/Paris%20MOU%20NIR.pdf .
Pazzani, M.J., Mani, S., Shankle, W.R., 2001. Acceptance of rules generated by machine learning among medical experts. Method Inform. Med.40 (05) 380–385.
Pei, S., Hu, Q., Chen, C., 2016. Multivariate decision trees with monotonicity constraints. Knowl.-Based Syst.112 14–25.
Santosa, F., Symes, W.W., 1986. Linear inversion of band-limited reflection seismograms. SIAM J. Sci. Stat. Comput.7 (4) 1307–1330.
Sill, J., 1997. Monotonic networks. Adv. Neur. Inform. Process. Syst.10 661–667.
Tokyo MoU, 2014. Information sheet of the new inspection regime (NIR) Accessed 13July 2019 http://www.tokyo-mou.org/doc/NIR-information%20sheet-r.pdf .
Tokyo MoU, 2018. Memorandum of understanding on port state control in the Asia-Pacific Region.Accessed 19 October 2019, http://www.tokyo-mou.org/.
Tokyo MoU, 2020. Annual report on port state control in the Asia-Pacific Region 2019. Accessed 1 Aug 2020 http://www.tokyo-mou.org/doc/ANN19-f.pdf .
Tsou, M.C., 2019. Big data analysis of port state control ship detention database. J. Marine Eng. Tech.18 (3) 113–121.
Van Den Akker, M., Hoogeveen, H., Van De Velde, S., 2005. Applying column generation to machine scheduling. Column generation. Springer, Boston, MA.
Wang, S.Zhen, L., Zhuge, D., 2018. Dynamic programming algorithms for selection of waste disposal ports in cruise shipping. Transportation Research Part B 108, 235–248.
Wang, S., Yan, R., Qu, X., 2019. Development of a non-parametric classifier: Effective identification, algorithm, and applications in port state control for maritime transportation. Transp. Res. Part B 128, 129–157.
Wu, L., Wang, S., Laporte, G., 2021. The Robust Bulk Ship Routing Problem with Batched Cargo Selection. Transp. Res. Part B Methodol.143 124–159.
Xu, R., Lu, Q., Li, W., Li, K.X., Zheng, H., 2007. A risk assessment system for improving port state control inspection. In: Proceedings of 2007 International Conference on Machine Learning and Cybernetics, pp. 818–823.
Yan, RanWang S., Fagerholt K., 2021c. Coordinated approaches for Port State Control Inspection planning. Maritime Policy & Management doi:10.1080/03088839.2021.1903599.
Yan, R., Wang, S., 2019. Ship inspection by port state control—review of current research. Smart Transportation Systems 233–241 2019.
Yan, R., Wang, S., Fagerholt, K., 2020a. A semi-“smart predict then optimize”(semi-SPO) method for efficient ship inspection. Transp. Res. Part B Methodol.142 100–125.
Yan, R., Wang, S., Peng, C., 2020b. An artificial intelligence model considering data imbalance for ship selection in port state control based on detention probabilities. J. Comput. Sci.48, 101257.
Yan, R., Wang Peng, S.C, 2021a. Ship selection in port state control: Status and perspectives. Maritime Policy and Management in press.
Yan, R., Zhuge, D., Wang, S., 2021b. Development of two highly-efficient and innovative inspection schemes for PSC inspection. Asia-Pacific J. Oper. Res. in press.
Yang, Z., Yang, Z., Yin, J., 2018a. Realising advanced risk-based port state control inspection using data-driven Bayesian networks. Transp. Res. Part A 110, 38–56.
Yang, Z., Yang, Z., Yin, J., Qu, Z., 2018b. A risk-based game model for rational inspections in port state control. Transp. Res. Part E 118, 477–495.
Yi, W.Sutrisna, M., 2021. Drone scheduling for construction site surveillance. Computer-Aided Civil and Infrastructure Engineering 36 (1), 3–13.
Zhang, P.Zhao, L., Vata, O., Rajagopal, S., 2020. Restructuring seafarers’ welfare under the Maritime labour convention: An empirical case study of Greece. Maritime Business Review 5 (4), 373–389.
Zhen, LuChew, E.P., Lee, L.H., 2011. An integrated model for berth template and yard template planning in transshipment hubs. Transportation Science 45 (4), 483–504.
Zhen, Lu, 2015. Tactical berth allocation under uncertainty. European Journal of Operational Research 247 (3), 928–944.
Zhen, Lu, 2016. Modeling of yard congestion and optimization of yard template in container ports. Transportation Research Part B 90, 83–104.
Zhen, L., Wu, Y., Wang, S., Laporte, G., 2020. Green technology adoption for fleet deployment in a shipping network. Transp. Res. Part B Methodol.139 388–410.