DEVOTION
Journal of Research and Community Service

Volume 4, Number 7, April 2026
e-ISSN: 2797-6068 and p-ISSN: 2777-0915

Inventory Out of Stock Inference Using Machine Learning With Ensemble Strategies

Oka Mahendra Saputra* , Prima Gumilang Dwi Putra, Bening Abdul Aziz, Alvin Aryanto
Inventory and Warehouse Data Analytics Team, Indonesia
okamahen@gmail.com* , gumilangp@gmail.com, aazis0814@gmail.com, alvinaryanto@gmail.com

KEYWORDS
inventory management; out-of-stock (oos); machine learning; business strategies; wms optimization.

ABSTRACT
Inventory Record Inaccuracy (IRI) poses a significant risk to business continuity, particularly for Maintenance, Repair, and Overhaul (MRO) enterprises. While rigorous record-keeping is intended to ensure data integrity, it paradoxically often leads to IRI, manifesting as Inventory Freezing or Phantom Inventory. This study aims to demonstrate a comprehensive data analytics pipeline—from data acquisition and transformation to analysis and decision-making—using a machine learning approach. The research employed Random Forest Classifier (RFC) and Extreme Gradient Boosting (XGB) models on a dataset of 10,369 observations from a Jakarta-based warehouse, enhanced with 16 supply chain features and 12 Boolean indicators for missing data. The results indicate a clear precision-recall trade-off: RFC achieved 85% precision with 50% recall, while XGB attained 93% recall with 29% precision after hyperparameter tuning. To overcome this, two ensemble strategies are proposed: Consensus (voting) and Cascading (pipeline) ensembles, which offer management practical options to optimize stock-checking efforts. The key implication is that no single model is superior, but a strategic combination of models can effectively predict Out-of-Stock (OOS) events, reducing reliance on heuristic inventory checks

INTRODUCTION

One of the competitive advantages of the Maintenance, Repair, and Overhaul (MRO) industry is its capability to hold inventory within certain time that will be used to generate value via service offered to its customer (Ali, 2022). MRO project planning and business projection rely on the database by looking at historical activities such as consumption and turn-around time of each project compared to current available inventory, that finally broken down as spare part planning, daily activities or menus and workload (Waller, 2019). Accuracy of the inventory status is pivotal point on project planning, thus having part true location with record is non-negotiable for survival of the business in the age of Internet-of-Things (IOT) (Fildes, 2019).

(Choi, 2018) In real-life situation, having perfectly matched inventory versus recording accuracy impose high load on record keeping, resulting in lacking of flexibility and speed to fulfil request, causing Out of Stock (OOS) and finally causing Inventory Record Inaccuracy (IRI). In helicopter view, IRI can be caused from two possible scenarios; Inventory Freezing and Phantom Inventory. Inventory freezing (Kang & Gershwin, 2005a) is a negative discrepancy between stock and record, causing overestimation due to stock on hand is less or actually depleted while record stays available, and automated replenishment or reorder via

reorder point will not work, resulting non added value manual justification and recount. In the other hand, Phantom inventory (LOKAD, 2026) is a positive discrepancy between stock and record, causing underestimation of stock availability and might trigger costly double purchase for stock that is actually exist. In store level view, reason of inventory inaccuracies includes external and internal unlawful transaction (Canella et al., 2014), incorrect incoming and outgoing deliveries, misplaced items (Raman et al., 2001a), cancellation of usage, pooled part record transaction, and excess of production, with estimates causing profit reduction up to 10%. IRI are globally recognized problem, ranging from 51% inventory inaccuracies after manual inventory verification of 500 stores of global retailer (Kang & Gershwin, 2005a), up to 65% observed inaccuracies out of 369.567 inventory record collected from 37 leading retailers in the USA (DeHoratius & Raman, 2008).

Many warehouse all over the world implement basic inventory tools to combat IRI such as periodical stock-opname and frequent stock take or cycle counting. Following the technology roadmap, another approach on reducing IRI also includes the implementation of RFIDs, smart detector, surveillance, and another loss prevention.

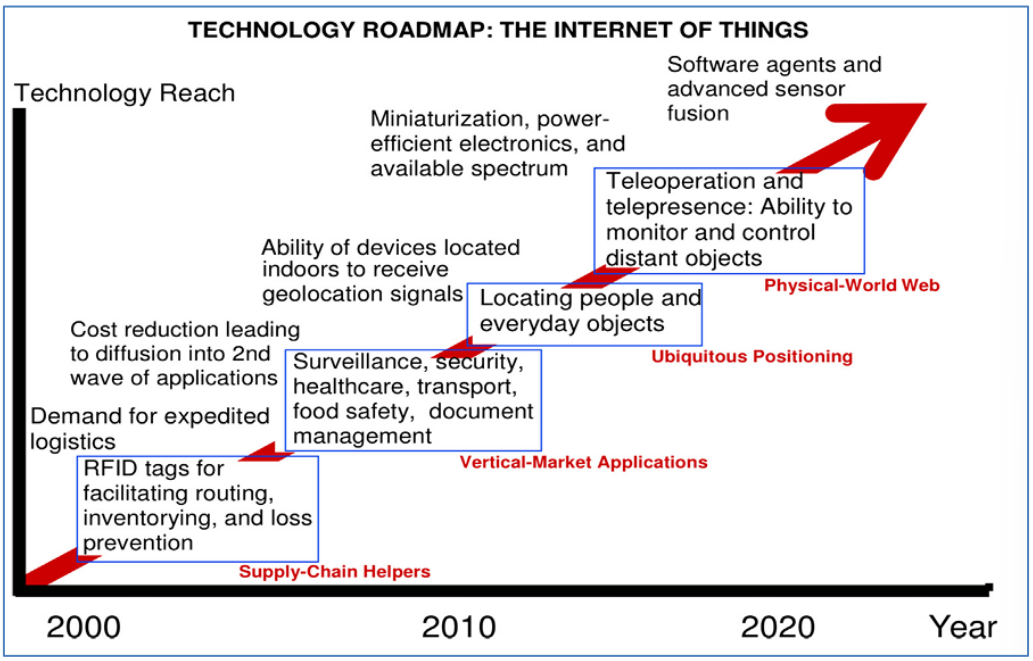

Figure 1. Technology Roadmap: The Internet of Things (Tejesh and Neeraja 2017)

The adaptation and implementation advanced prediction tools by using Machine Learning (ML) in the warehouse floor is started from Fisher and Raman at 2010 (Dolgui, 2020). However, due to the nature of warehouse worker and leader that is operational focused, there were some lacking and insignificant insight using advanced prediction. Even so, all warehouse practitioner shares the same anxiety toward this particular case; is it possible to know the possibility of OOS happens before it happens, and how to detect OOS before it causing further problem? To detect OOS, most warehouses implement basic inventory control tools using common classification method such as Pareto technique based on contributing values, or ABC technique that captures volume-values pair (Saran & Satsangi, 2025).

(Hofmann, 2021) Classical method to detect OOS is similar with zero balance walks strategy (physical audit or stock take), where employee walk the store periodically to check for

possible OOS, perform report and adjustment (Fisher & Raman, 2010). Improvement on this method is approach to detect OOS is by using Point of Sale (POS) data and combined with the implementation of RFID to reduce manpower usage and error from human limitation (Gruen and Corsten, 2007). While RFID implementation is proven to increase productivity, many raise concern regarding possibilities to produce false negatives (read rate as low as 30%) on goods containing metals and also for stacked textiles, also with the high cost of implementation itself (Metzger, 2013). While this issue already tackled by tags manufacturer using special RFID tags on chemical parts and metal parts, together with tuning of the receptor antenna, implementation of RFID to reach 100% accuracy the question of policies and compliances. Further approach for detecting OOS provided by Sonali using probabilistic based detection using movement of SKU with the period of zero sales, provided with algorithm to run the checking process (Sonali et al., 2019). Sonali approach gives a rule of thumb on how to detect OOS on all industries, and enhancement of such technique is possible given to the raise of accessible ML model worldwide.

In real world best practices, it is almost impossible to have single model that performs best to all of business requirements and targets, since each of ML techniques have their own strength and weaknesses (Kourentzes, 2020). One of its main problems is that for each model to work, we should set the Detection Threshold (a number between 0 and 1) as confidence of how "sensitive" the detection should be, and this Threshold will influence the False Positive and False Negative of each model; Precision improves as False Positive decrease, and Recall improves as False Negative decrease. It is highly rare for a single model to have high value on both recall and precision (Boute, 2021).

The urgency of this research is driven by the limitations of existing methods and the operational anxiety shared by warehouse practitioners: the need to predict OOS events before they cause disruption. Classical techniques like ABC analysis (Saran & Satsangi, 2025) classify inventory based on volume-value pairs but do not predict future discrepancies. While RFID and probabilistic models offer improvements, they often require significant investment or are limited by their underlying assumptions. Therefore, a robust, data-driven method that leverages readily available warehouse metrics to provide actionable predictions is critically needed. This study addresses this urgency by developing a machine learning pipeline that can be integrated into existing Warehouse Management Systems (WMS).

The novelty of this research lies in its strategic application and combination of two powerful, yet distinct, machine learning models: Random Forest Classifier (RFC) and Extreme Gradient Boosting (XGB). Unlike many studies that focus on a single model, this research acknowledges that no single algorithm excels in all performance metrics. RFC is known for its bagging and stability through parallel voting, while XGB is recognized for its boosting capability to create robust and strong trees through local inference strengthening. The novelty is not just in applying these models to IRI detection, but in explicitly designing and evaluating two ensemble strategies—Consensus (voting) and Cascading (pipeline)—to harness the complementary strengths of each model. This approach transforms the trade-off between precision and recall from a problem into a strategic asset.

This research aims to demonstrate a full-pipeline data analytics approach to infer potential IRI and its indicators. The specific objectives are threefold: (1) to demonstrate how to take warehouse metrics and structure them into a data-analytics-ready format; (2) to perform

exploratory data analysis and data handling for problematic data, such as missing values; and (3) to create baseline models and propose implementation strategies based on model performance. The contribution of this research is a practical framework for management to arrange more cost-effective cycle counting activities. The key deliverables include performance metrics (Precision, Recall, F1-Score, Feature Importance) and a set of business options with their associated risk-reward profiles.

(Elsayed, 2024;Raman et al., 2001b;Kang & Gershwin, 2005b) Based on these conditions, we decided to implement Random Forest Classifier (RFC) and Extreme Gradient Boosting (XGB) algorithm as means to predict potential IRI and its indicator. We choose both models due to capability of XGB to perform local inference strengthening on each branch, creating more robust and stronger trees, while RFC perform bagging and stability through parallel voting. Performance of both models will be evaluated, and possible solution or strategies is presented harnessing the strength and weakness of both (Carbonneau, 2018).

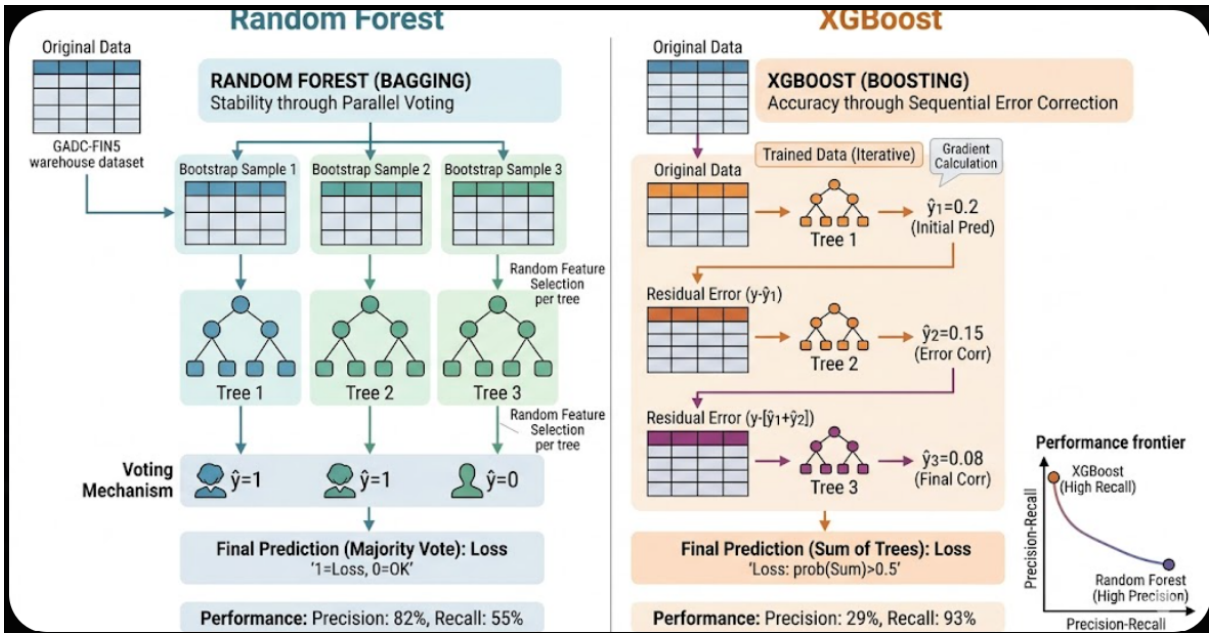

Figure 2. Random Forest Classifier and XGBoost (Image Generated by Gemini AI)

More specifically, we wanted to show how to:

1) Take warehouse metrics and structure in a data-structured format for data analytics.
2) Perform exploratory data analytics and data handling for problematic data.
3) Create model from the database as baseline and strategies of implementation.

The results are consideration for any management on planning cycle counting activities and menus that will be cost effective in terms of manpower and increased detection of OOS.

METHOD

Method used in this work consist of 5 steps as summarized in Table 1 below. Description on each steps is given and will be referred on the "Result and Discussion" section :

1. Data Collection and Handling
•Data collection from all warehouses and handling of missing data
2. Feature enhancement
•Addition of computable statistical feature for each SKU
3. Dataset Construction and Exploratory Data Analysis (EDA)
•Construction of dataset based on initial data collection and feature enhancement
4. Machine Learning Modelling and Visualization
•Perform Random Forest Classification (RFC) and Extreme Gradient Boosting (XGB) •Analyze performance on both models and the trade-offs
5. Operation Strategies for OOS Inference
•Analyze possible strategies to implement inference in business process, by using Consensus Ensembles or Cascading Ensembles.

Table 1. Step for Data Analysis and Model Creation

During data collection and handling stage, data that will be used is gathered from warehouse in Jakarta region, where "golden sample" data for training initial model was taken from 2023 inventory data including both OOS and AVAIL classification after stock check. Analysis of dataset will be provided as basis of model and process selection, since difference dataset condition will give different ways of inference results.

At feature enhancement stage, we check the statistical (computed) feature for each SKU and add in the new column on each SKU. Having this data is beneficial to check if the statistical feature can also give insight of possibilities of OOS. Feature that is planned to be used will adopt the insight given from summarised at Table 2 below.

Table 2. List of features for detection with enhancement

Category	Description	Category Name	Feature Source	Ref
Identity Mask	ID number for each part masked	CatName	Calculated	New
SKU Feature	Identity of SKU for each plant	SF1	Information System	New
	Description of the SKU as available in market	SF2	Information System	New
	Consumable or Rotable type of SKU	SF3	Information System	New
	Lot or Batch Number recorded during incoming	SF4	Information System	New
	Unit of measure, minimum physical quantity per SKU	SF5	Information System	New
	If another SKU with similar identity exist and causing possible confusion during stock take	SF6	Calculated	New
	If SKU with similar Lot or Batch Number and causing possible confusion during stock take	SF7	Calculated	New
	Size of SKU stored inside the warehouse	SF8	Calculated	New

Warehousing Feature	Warehouse location in Jakarta region	WF1	Information System	New
	Total quantity recorded per SKU	WF2	Information System	New
	Actual quantity after stock take per SKU	WF3	Stock Take	New
	Storage type on warehouse for SKU	WF4	Information System	New
	Small storage bin location on warehouse for SKU	WF5	Information System	New
	Frequency of SKU located in transit bin per total location of internal movement	WF6	Calculated	New
	Shift of crew on duty handling the transaction	WF7	Information System	New
	Storage bin fullness, quantity stored per storage bin	WF9	Calculated	New
	Notification from Stock Take team that SKU was wrongly placed	WF8	Stock Take	New
Supply Chain Feature	Total external movement (transfer to other warehouse) of SKU per frequency of movement SKU	SCF1	Calculated	[Juan 2021]
	Quantity of movement per external movement (transfer to other warehouse)	SCF2	Calculated	[Juan 2021]
	Frequency of sales per frequency of total frequency of movement	SCF3	Information System	New
	Quantity of SKU sold per total quantity SKU available	SCF4	Information System	[Juan 2021]
	Days of part stored until sold	SCF5	Calculated	[Juan 2021]
	Days of unsold SKU	SCF6	Calculated	[Juan 2021]
	Days of unsold SKU per total working days	SCF7	Calculated	[Juan 2021]
	Average quantity of item sold	SCF8	Calculated	[Juan 2021]
	Standard deviation of item sold in the same period	SCF9	Calculated	[Juan 2021]
	Average SKU sold per average of days sold in the same period	SCF10	Calculated	[Juan 2021]
	Quantity of SKU returned to warehouse from sell cancellation per total working days	SCF11	Calculated	New
	Quantity of SKU returned to warehouse from sell cancellation per total days of return transaction	SCF12	Calculated	New
	Total quantity of SKU sold in the same period	SCF13	Information System	New
	Frequency of SKU sold in the same period	SCF14	Information System	New
	Quantity of SKU consume per total quantity of all SKU consume	SCF15	Calculated	New
	Quantity of SKU sold per total frequency of item sold	SCF16	Calculated	New
	Boolean Missing data handling	SCx1 to SCx12	Calculated	New

At dataset construction and EDA stage, we check on the data constructed and based on previous enhancement. We choose not to perform data dimensionality reduction process in the initial training to avoid possible significant feature loss before training proceed. Imputation

method chosen for handling missing data is using Boolean masking feature instead of dropping the whole cell to preserve the information on the value that originally missing, since it is possible that the missing data itself can be an important feature for detection of OOS in the next stage.

At machine learning and visualization stage, we perform data splitting into train-test pair dataset using python library scikit-learn, with selected model Random Forest Classifier (RFC) and Extreme Gradient Boosting (XGB) (Andaur et al., 2021). For this study, hyperparameter tuning was performed manually on specific item to improve detection threshold. To check the model performance, we will compare internal parameter for both model such as Feature Importance, Recall, Precision, and F1-Score. Heatmap analysis is not performed since both models is robust against multicollinearity. Classification of each detection can be seen in Table 3 as below:

Table 3. Observation and Prediction Matrix

		Prediction	
		False (0)	True (1)
Actual Or Observation	False (0)	True Negative (TN)	False Positive (FP)
	True (1)	False Negative (FN)	True Positive (TP)

Accuracy is ratio of correctly predicted observation (either positive or negative) from total observation. High accuracy is best used for predicted classes (in this case, OOS and AVAIL) is balanced.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision is ratio of correctly predicted observation from total predicted positive. Total positive is the actual TP value added with FP (positive detection but negative actual). High value is crucial if the cost of FP is high (example, classification of inventory level that might lead to inventory increase).

$$Precision = \frac{TP}{TP + FP}$$

Recall is ratio of correctly predicted positive observation to all actual positives, which is actual TP value with FN (negative prediction but positive actual). High Recall value is desired when the cost of FN is high (example, detection of COV-19 or detection of spam email).

$$Recall = \frac{TP}{TP + FN}$$

F1-Score is the harmonic mean that balance the effect of Precision and Recall. High F1-Score is desired if the data is imbalanced and accuracy alone might result in misleading inference.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

At operation strategies for OOS inference, we devise two main strategies considering performance, strength and weakness from each model: the Consensus Ensemble or Cascading Ensemble. Let sensitivity threshold (or XGB detection threshold) as T_S, auditor threshold (or RCF detection threshold) as T_A, and prediction probability is $prob_{XGB}$ and $prob_{RFC}$. Consensus Ensemble is analogous to independent voting mechanism, where if both model "agrees" that observation is potentially causing OOS, it will be flagged as OOS and *Decision(x)* for stock checking for said observation is a go.

$$\text{Decision}(x) = \begin{cases} \text{Audit} & \text{if } (prob_{XGB} > T_S) \cap (prob_{RFC} > T_A) \\ \text{Ignore} & \text{otherwise} \end{cases}$$

Cascading Ensemble is analogous to continuous two stages "Pipeline" process; Stage 1 ($prob_{XGB} > T_s$) is initial filtering process where XGB model is used for initial classification, and only when Stage 1 determine possible OOS, Stage 2 ($prob_{RFC} > T_A$) can start as final data audit and *Decision(x)* for stock checking for said observation is a go.

$$\text{Decision}(x) = \begin{cases} \text{Audit} & \text{if } (prob_{XGB} > T_S) \rightarrow (prob_{RFC} > T_A) \\ \text{Ignore} & \text{otherwise} \end{cases}$$

These strategies will be evaluated with snapshot of data on the next period, together with its performance, risk and benefit, will be presented.

RESULT AND DISCUSSION

Data Cleaning and Enhancement Steps (Step 1 – 3)

(Cannella, 2017) Initial data collection is performed swiftly, and since the nature of the OOS dataset is always imbalance (only x% from total population is OOS), we chose not to use regression due to its focus on global average that might miss the subtle change of OOS, and also not KNN due to prediction is only considering its neighbouring value resulting in potentially low Recall and Precision overall (Papakiriakopoulos, 2009). Another deviation in inventory found in data is extra stock in inventory, and to avoid data loss we decide to add classification "*Extra Stock*" coded XST. We chose classification model to continue the analysis, which is XGB and RFC. All data on warehouse point of view is captured and collected in feature by including in Warehouse factor (category named WF). The underlying SKU data (most of category named SF) are proprietary assets and are restricted to ensure data confidentiality, hence we use the ordering number as identity of the part, however it poses contributing error when model think of its cardinal number as observation value, so we perform data masking using Label Encoder, results in Table 4.

Table 4. SKU with ID conversion results

Original data	Data with Mask
2132	WF5_WWE-2802
n_{th} data	WF5_WWE-n_{th}

To enhance the dataset, we use calculated statistical data to enhance the current dataset by including category Supply Chain factor (category named SCF). Beside of the nature condition of imbalance dataset, there are also plenty of data that is not available (valued as #N/A), and these data is spread well along the dataset itself. We suspect that the loss of statistical data might also be the indicator of OOS, so increasing the "Signal" is chosen as method of enhancement by adding Boolean indicator (category named SCx) instead of removing them. Both original data and its enhancement merged into one single dataframe using '*pandas'* library and exported as comma separated value (.csv) using '*openpyxl'* conversion library.

Table 5. Data enhancement results

Observation	**Total Observation**	
	Original	**Enhanced**
Original Dataset	10.369	10.369
Total N/A observation	6306	0
Enhanced to signal	0	6306
Signal to Noise rate	60,81%	100%

Note: all data not available converted to signal.

Fully enhanced data is used as primary data to train the model for RCF and XGB. Both Hyperparameter tuned and detection threshold set is given with various trial and chosen after finding local best performance. Detailed as Table 6 below:

Table 6. Manual hyperparameter tuning on both XGB and RFC

Model name	Hyperparameter (Hyp) or Threshold (Thr).	Function	Original Value	Tuned
Random Forest Classifier				
	n_estimators (Hyp)	Number of decision trees built	100	100
	random_state (Hyp)	Control of randomization	42	42
	class_weight (Hyp)	Internal weight for imbalance data handling	default	'balanced_subsample'
	stratify (Hyp)	Stratified sampling for imbalance data handling	default	default
	sensitivity threshold (Thr)	Detection threshold using RFC	default	0.5
Extreme Gradient Boosting				
	n_estimators (Hyp)	Number of decision trees built	100	200
	learning_rate (Hyp)	Scaling factor controlling contribution of each trees	0.3	0.5
	max_depth (Hyp)	Maximum number depths on each decision trees	6	6
	Objective (Hyp)	Multi-class classification tasks with output probability distribution	'squarederror'	'multi:softprob'
	random_state (Hyp)	Control of randomization	42	42
	auditor threshold (Thr)	Detecton threshold using XGB	N/A	0.2

XGB works by increasing global capability of each branch to infer OOS or AVAIL part, and it perform good for imbalance data that need specific detection. Hence, we think it is important for the model hyperparameter to be tuned slightly higher. RCF in other ways work

well in its "vanilla" mode, so most of the hyperparameter is unchanged. For *n_estimators* hyperparameters, the detection is set 2 times from default to increase its detection capability.

Training and Evaluation steps (Step 4)

Since heatmap is useless to be used for RFC and XGB, we use Feature Importance with Permutation to evaluate which features contribute best toward model inference. We check for both models and extract 20 top features that contribute most on each model. Interestingly we found 40% of top 20 features importance is overlapping between RFC and XGB, which means that both models have "agreed" that these highlighted features do significant contribution on OOS conditions. Another 60% of top 20 features importance is independent, indicates that both models exhibit strong specialization after training.

Table 7. Feature Importance using mean and standard deviation of importance probability

Models	Feature_Consensus	Feature	Importance_Mean	Importance_Std
RFC	Consensus	CatName	0.012825	0.001242
	Consensus	WF4_CSN	0.010125	0.000682
	Consensus	WF6	0.004339	0.001011
	Consensus	SCF13	0.004147	0.000722
	Consensus	WF2	0.002411	0.000682
	Consensus	WF8	0.001736	0.000654
	Consensus	SCF10	0.000868	0.001030
	Consensus	WF3	0.000579	0.001537
	Independent	SCx9	0.000868	0.000709
	Independent	SCF16	0.000771	0.000894
	Independent	SCF9	0.000771	0.000236
	Independent	SCx4	0.000771	0.000386
	Independent	SCx8	0.000771	0.000579
	Independent	SCx10	0.000675	0.000654
	Independent	SCx11	0.000675	0.000783
	Independent	SCF15	0.000579	0.000640
	Independent	SCF14	0.004339	0.000807
	Independent	SCF2	0.002411	0.000528
	Independent	SCF1	0.001736	0.001082
	Independent	SCF4	0.000964	0.000431
XGB	Consensus	CatName	0.059466	0.004585
	Consensus	WF4_CSN	0.005304	0.000804
	Consensus	WF6	0.003857	0.001345
	Consensus	SCF13	0.022581	0.003320
	Consensus	WF2	0.008116	0.002896
	Consensus	WF8	0.008920	0.001548
	Consensus	SCF10	0.004179	0.002239
	Consensus	WF3	0.002170	0.000732
	Independent	SCF11	0.031903	0.001390
	Independent	WF4_23Q	0.015429	0.000227
	Independent	SCF5	0.010045	0.001480

Independent	WF9	0.004018	0.000278
Independent	SCF7	0.002893	0.001306
Independent	SCF3	0.002652	0.000475
Independent	WF4_23I	0.002411	0.000161
Independent	WF4_23H	0.001205	0.000139
Independent	WF5_WWE-2501	0.000643	0
Independent	WF5_AUR-3202	0.000563	0.000139
Independent	WF5_0110	0.000482	0.000161
Independent	WF5_5217	0.000402	0.000139

RFC seems to be more sensitive on checking data loss – signal conversion (SCx), while XGB is more sensitive on evaluating warehouse and supply chain factors. Having strong Consensus and Independent feature explains potential robustness for both models to be performed simultaneously or in tandem.

We dig deeper into the "agreed" features from both models, comparing both metrics shows insight that feature *CatName* gives highest contribution on model inference. This indicates that some SKUs does have high possibility of OOS compared to others and knowing which SKUs that exhibit correlation to OOS is an important characteristic for priority of stock check. XGB highly sensitive toward WF8, WF2, SCF13 and *CatName*, and RFC is highly sensitive toward SCF13 and WF6.

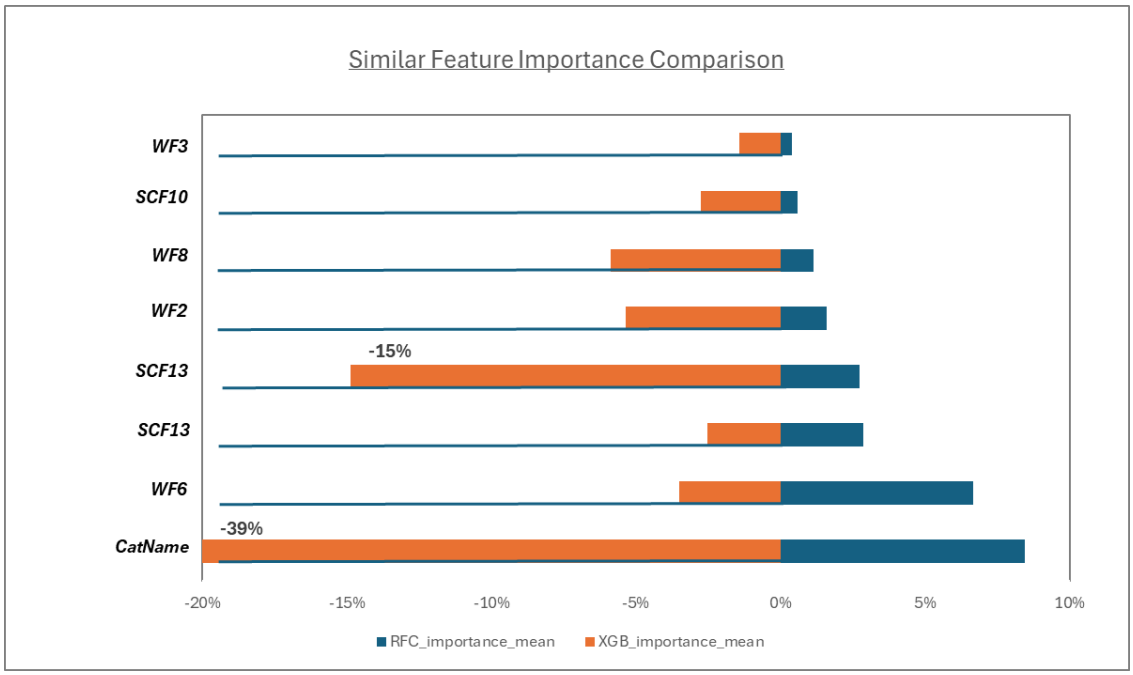

Figure 3. Similar features and comparison between XGB and RFC

Model inference performance gathered using internal '*sklearn.metrics'* function after the end of train-test run, potential strength and weakness is evaluated. RFC model shows exceptionally high precision, and tuning the hyperparameter shows increase 4,71% of precision (81% to 85%) but reduce the Recall from 54% to 50%. XGB model perform poorly in "vanilla" mode, and tuning the hyperparameter with threshold sets to 0.15 increase the model detection

to 221% (29% to 93%) while sacrificing Precision (45% to 29%). F1-score using default and enhancement does not change drastically, indicating that both models is working with its own strength on one type of detection, either Precision or Recall. This finding is consistent with the theory regarding trade-off of both detection performance in real dataset settings mentioned on "Method" section. Further manual tuning shows that it is almost impossible to get high value on both Precision and Recall, but it left room for next improvement by hyperparameter tuning using automatic tuning via python library. Results before and after parameter tuning can be seen in Table 8 below.

Table 8. Model Performance (Precision, Recall, and F1-Score)

Model	Hyperparameter Tuning	Precision	Recall	F1-Score
RandomForest	Default-XST	0.00	0.00	0.00
RandomForest	Default-OOS	0.81	0.54	0.64
RandomForest	Default-AVAIL	0.95	0.99	0.97
RandomForest	AllClassBalance-XST	0.12	0.03	0.05
RandomForest	AllClassBalance-OOS	0.85	0.50	0.63
RandomForest	AllClassBalance-AVAIL	0.94	0.99	0.97
XGBoost	Default-XST	0.08	0.06	0.06
XGBoost	Default-OOS	0.45	0.29	0.35
XGBoost	Default-AVAIL	0.93	0.96	0.94
XGBoost	Tres015-XST	0.07	0.41	0.12
XGBoost	Tres015-OOS	0.29	0.93	0.44
XGBoost	Tres015-AVAIL	0.99	0.72	0.83

Operation Strategies step (Step 4)

As expected, none of the models proposed exhibit perfect inference capability by having high Precision and high Recall at the same time; each model does have its own strength and weakness, using the model alone will unstable and highly misleading, that finally lead to wasted resources. This is consistent with Precision-Recall trade-off we mentioned above and unless there are any significant improvement in the modelling, the challenge will be the same for 3-5 years to come. To increase the inference capability, we propose to use both models in tandem or in sequence; Consensus Ensemble or Cascading Ensemble.

Table 9. Management options for improving inventory check and reduce OOS.

Strategy	Math Gate	Cost	Benefit	Management Justification
Consensus (Voting)	$P_1 \cap P_2$	Medium cost	Medium to High benefit	Low to Zero False OOS inference, Low to Moderate Manpower
Cascading (Pipeline)	$P_1 \rightarrow P_2$	Low cost	Medium to High benefit	Efficient Screening with Acceptable False Positive of OOS inference, Moderate Manpower
Casual Operations	#N/A	High cost	Low to Medium benefit	Potential waste on checking the part in OK condition

Using Consensus Strategy, we will expect close to zero false alarm, meaning that we only act when the models or "experts" are in 100% agreement. This approach is considered

conservative since it requires both to “agree” first before included in list of checking. The model size will grow bigger and the inference time will be increased, but since the inference cost is in the matter of minutes, inference cost in term of time spent in machine learning model can be neglected. The code block to run this strategy is parallel and both results is visible and can be compared at the same time, and only when both models detect positive it will be considered as OOS potential. This model is more robust to internal parameter change since changes in one model does not necessarily affect result in another model, and the effect of process flow is insignificant.

Using Cascading Ensemble strategy, the analogy is fishermen that throw fishing nets to get the fish, and fishery expert to classify the fish into something that is valuable or invaluable; both action is done in serial format. We should perform consciously first using any model that have highest Recall value to catch as much OOS potential as possible and then perform auditing using model that have highest Precision value to accurately detect observation is likely to become OOS and focus on that collection of the data. This method is prone to error if the user performs first using high Precision model continued with high Recall model, that might cause miss leading by increasing the False Negative, one that flagged not OOS but actually have high potential of OOS. This should be considered if the cost of wrong flagging is important. As rule of thumb, first chose any model with higher Recall but mediocre F1-score, followed by audit model with higher Precision but significantly high F1-score.

CONCLUSION

This study is ventured to show that it is possible to use Machine Learning model to perform stock checking rather than only relying to the current heuristic process of inventory check. Our model consistently shows that there is currently no model that is good on both Recall and Precision, even high F1-score. By understanding each weakness and strength, it is highly possible to use both in tandem or continuously. The distribution of top 20 Important Features proves that both model is well trained and is specialized in different point of view. RFC results shows that it can handles conversion of N/A valued features by including most of them in the top 20, indicating that our suspicion of data loss can also lead to possible OOS. RFC also found to be more focused on Supply Chain factor (9 out of 20, or 45%), and XGB more focused on Warehouse factor (13 out of 20, or 65%) while completely ignore SCx factor. Hyperparameter tuning was proven to improve the detection by increasing Precision up to 4,7% on RFC and Recall up to 221% for XGB, but due to limitation in manual tuning, it is advised to use auto hyperparameter tuning library on the future research. While setting threshold to 0.15 is helpful to increase Recall in XGB, the effect should be studied further. For class “XST”, it is hard to justify the possibilities given that the population is small compared to both “AVAIL” and “OOS”, so we can consider the concatenation of both “XST” and “OOS” in the next venture. It is imperative that next venture must also improve the implementation process and see if only both method is available to perform or other flow is also possible such as model auto selection.

REFERENCE

Ali, M. (2022). Predictive analytics in supply chain. *Transportation Research Part E*. https://doi.org/10.1016/j.tre.2021.102610

Andaur, J. M. R., Ruz, G. A., & Goycoolea, M. (2021). Predicting Out-of-Stock Using Machine Learning: An Application in a Retail Packaged Foods Manufacturing Company. *Electronics*, *10*(22), 2787. https://doi.org/10.3390/electronics10222787

Boute, R. (2021). Forecast accuracy and inventory performance. *Omega*. https://doi.org/10.1016/j.omega.2020.102281

Canella, D. S., Levy, R. B., Martins, A. P. B., Claro, R. M., Moubarac, J. C., Baraldi, L. G., Cannon, G., & Monteiro, C. A. (2014). Ultra-processed food products and obesity in Brazilian households (2008-2009). *PLoS ONE*, *9*(3). https://doi.org/10.1371/journal.pone.0092752

Cannella, S. (2017). Inventory record inaccuracy – The impact of structural complexity and lead time variability. *Omega*, *68*, 123–138. https://doi.org/10.1016/j.omega.2016.06.002

Carbonneau, R. (2018). Machine learning demand forecasting. *International Journal of Production Economics*. https://doi.org/10.1016/j.ijpe.2018.01.004

Choi, T. M. (2018). Big data analytics in operations management. *Production and Operations Management*. https://doi.org/10.1111/poms.12838

DeHoratius, N., & Raman, A. (2008). Inventory Record Inaccuracy: An Empirical Analysis. *Management Science*, *54*(4), 627–641. https://doi.org/10.1287/mnsc.1070.0789

Dolgui, A. (2020). Smart inventory management using AI. *International Journal of Production Research*. https://doi.org/10.1080/00207543.2019.1702229

Elsayed, N. (2024). Improving Prediction Accuracy using Random Forest Algorithm. *IJACSA*, *15*(4).

Fildes, R. (2019). Forecasting and inventory control. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2018.09.015

Fisher, M., & Raman, A. (2010). *The new science of retailing: how analytics are transforming the supply chain and improving performance*. Harvard Business Review Press.

Hofmann, E. (2021). Machine learning in supply chain management. *International Journal of Production Research*. https://doi.org/10.1080/00207543.2020.1839957

Kang, Y., & Gershwin, S. (2005a). Information Inaccuracy in Inventory Systems. *Management Science*, *51*(6), 843–859. https://doi.org/10.1287/mnsc.1050.0419

Kang, Y., & Gershwin, S. B. (2005b). Information inaccuracy in inventory systems: stock loss and stockout. *IIE Transactions*, *37*(9), 843–859.

Kourentzes, N. (2020). Demand forecasting with machine learning. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2019.04.026

Metzger, C. (2013). RFID-based shelf inventory control policies. *Computers & Operations Research*, *40*(7), 1864–1873. https://doi.org/10.1016/j.cor.2012.10.019

Papakiriakopoulos, D. (2009). A decision support system for detecting products missing from the shelf. *Decision Support Systems*, *46*(3), 685–694. https://doi.org/10.1016/j.dss.2008.11.004

Raman, A., DeHoratius, N., & Ton, Z. (2001a). Execution: The missing link in retail operations. *California Management Review*, *43*(3), 136–152.

Raman, A., DeHoratius, N., & Ton, Z. (2001b). Execution: The Missing Link in Retail Operations. *California Management Review*, *43*(3), 136–152. https://doi.org/10.2307/41166088

Saran, A., & Satsangi, P. (2025). Inventory & Logistical Operations in Warehousing. *GRANTHAALAYAH*, *13*(9), 206–216.

Sonali, P., Santosh, W., Bhagyashri, G., Pranit, K., & Yash, J. (2019). Anti-HIV/AIDS Drugs: An Overview. *Journal of Drug Delivery & Therapeutics*, *9*(3).

Waller, M. A. (2019). Big data analytics in supply chain. *Journal of Business Logistics*. https://doi.org/10.1111/jbl.12214

Lokad. (2026). *Stockout*. https://www.lokad.com/stockout/