Predicting Reach on Time Shipments: A Comparative Analysis of Decision Tree, Random Forest, and Gradient Boosting

Dastin Darmawan
Computer Science Department
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia, 11480
dastin.darmawan@binus.ac.id

Raditya Tamam
Computer Science Department
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia, 11480
raditya.tamam@binus.ac.id

Rudy Kurniawan Efendy
Computer Science Department
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia, 11480
rudy.efendy@binus.ac.id

Lili Ayu Wulandhari, S.Si., M.Sc., Ph.D
Computer Science Department
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia, 11480
lili.wulandhari@binus.ac.id

***Abstract* — This study aims to predict on-time shipment delivery for an international e-commerce company using three machine learning models: Decision Tree, Random Forest, and Gradient Boosting. The dataset comprises customer demographics, shipment methods, product characteristics, and customer interactions, with the target variable indicating whether a shipment was delivered on time. We conducted extensive data preprocessing, including handling missing values, encoding categorical features, and scaling numerical features. The models were evaluated based on their accuracy, precision, recall, and F1-score. Despite significant efforts in data preprocessing and model tuning, the challenge of accurately predicting on-time deliveries persists due to inherent limitations within the dataset. Current conflicts in this domain highlight the inadequacy of traditional models and underscore the necessity for advanced machine learning approaches. In this context, we propose a comparative analysis of the three aforementioned models, applying rigorous hyperparameter tuning using GridSearchCV and RandomizedSearchCV. Our experiments demonstrated that the Gradient Boosting model achieved the highest accuracy at 69%, outperforming both the Decision Tree and Random Forest models. However, even after optimization, the models could not surpass the 70% accuracy threshold, revealing substantial issues related to data imbalance, noise, and missing relevant features. Feature importance analysis identified shipment weight and prior purchases as significant predictors of delivery times. The study concludes that Gradient Boosting is the most reliable model among those tested.**

Keywords— Shipment, Prediction, Machine Learning, Gradient Boosting, On Time,

I. INTRODUCTION

Nowadays, the market has reached the globalization era and it is very competitive, efficient shipping logistics are crucial for businesses striving to maintain customer satisfaction and operational excellence and Shipping Delay can lead to significant financial losses, customer dissatisfaction, and operational inefficiencies[1].

There are challenges that can determine the timeliness of the shipment such as varying transportation modes, unpredictable weather conditions, and geographical location.

The problem addressed in this study revolves around predicting whether the shipping will reach on time or not. We used some of the attributes in the data to make a classification that can predict the delivery of goods will be on time or not[2].

The objective of this study is to explore various models of machine learning to show the accuracy of the classification based on the data that we gave and enhance the model to increase the accuracy it shows.

This paper has five sections and is structured as follows. Section I provides the introduction of this paper, consisting of the problem domain, problem background, problem definition and the objective of this paper. Section II provides a comprehensive review of literature related to reached on time shipping, shipping logistics and on-time performance. Section III provides the methodology that we used in our research. Section IV provides the result and discussion of the methodology that we used in our research. Section V provides the conclusion of our research

II. RELATED WORKS

The studies done by A. Jonquais and F. Krempl tested the data using three machine learning models including: Random Forest, Neural Network, and Linear Regression. The results show that the Random Forest model performs better to show the high accuracy value. Apart from the performance, they used the random forest model because it also proved to be easier to train and implement in a production system[1].

A study in shipment delivery prediction by Ermagun, Punel, and Stathopoulos (2020) explored the use of machine learning to predict shipment status on crowd-sourced online shipping platforms. They used a random forest algorithm to analyze data from 14,858 crowd-shipping requests collected across the United States. The model aimed to predict bidding, acceptance, and delivery phases of crowd-shipping using features such as shipping request details, built environment, and socioeconomic factors. The study found that the model could provide accurate delivery forecasts even without detailed package information, demonstrating

the potential of machine learning to enhance the efficiency and reliability of crowd-shipping services. [3]

In his 2020 thesis, Shehryar Shahid tackled the prediction of shipment delivery delays using machine learning, focusing on airfreight shipments. Shahid utilized real-world data conforming to International Air and Transport Association (IATA) standards and applied random forest and gradient boosting algorithms to build predictive models. Key to his research was meticulous data preprocessing to manage the event-driven nature of airfreight logistics. The models were evaluated on various metrics, including accuracy, precision, recall, F1-score, specificity, and sensitivity. Notably, the models achieved high specificity (over 90%) but lower sensitivity (under 44%), with overall accuracy surpassing 75%. [4].

Shahid also proposed an early prediction method to evaluate model performance when complete process information was unavailable, which yielded promising results. His study highlights the potential of machine learning to enhance logistics efficiency by predicting delays and emphasizes the importance of high-quality data and thorough feature selection. However, the exact source of the dataset was only described as adhering to IATA standards and applied to actual air freight shipping scenarios [4].

A study conducted by Hathikal et al.(2020) to develop a predictive model for estimating the shipment lead time using machine learning methods for ocean import freight. The dataset used in this study was obtained from an industry partner and multinomial logistic regression classifier outperform others classifier with accuracy 85.63%, followed by a decision tree method with accuracy 85.07%[5].

III. METHODOLOGY

A. Data Collection

The dataset in this study was collected from Kaggle[6], an online platform for datasets and competitions. An international e-commerce company collected the dataset to study about the customers in 2021. It includes 10999 values and 12 attributes. Table 1. is a detailed description of 12 attributes.

Table 1. Description of dataset.

No	Attributes	Description
1	ID	ID number of customers.
2	Warehouse block	The company have big warehouse which is divided into block such as A, B, C, D, and F.
3	Mode of shipment	The company ships the products in multiple way such as Ship, Flight, and Road.
4	Customer care calls	The number of calls made from enquiry of the shipment.
5	Customer rating	The company has rated from every customer. 1 is the lowest (Worst), 5 is the highest (Best).
6	Cost of the product	Cost of the product in US Dollars.
7	Prior purchases	The number of prior purchases.
8	Product importance	The company has categorized, the product in the various parameter such as low, medium, and high.
9	Gender	Male (M) and Female (F).
10	Discount offered	Discount offered on that specific product.
11	Weight in gms	It is the weight in grams.
12	Reached on time	It is the target variable, where 1 indicates it has NOT reached on time and 0 it has reached on time.

In this dataset, Reached on time was chosen as a target attribute to predict the outcome whether the product reached on time (0) or not (1).

B. Feature Selection

Feature selection was used to find and eliminate unnecessary attributes from the dataset. Since an attribute such as ID didn't offer any meaningful information, it was eliminated from the prediction. After removing the ID attribute, the dataset contained 11 attributes.

C. Data Splitting

Training and testing datasets were created from the original dataset. The training dataset used 80%(0.8) and the testing dataset used 20% (0.2).

D. Data Preprocessing

Checking missing values: The training and testing datasets were checked for missing values. Results show that both datasets had no missing values.

Encoding: Encoding was used to ensure training and testing datasets can be effectively used in building models, since some models can not perform while there were categorical attributes so categorical attributes needed to be encoded[7]. One-Hot Encoder and Label Encoder were used to encode the attributes. One-Hot Encoder was used to encode Warehouse block, Mode of shipment, and Product

importance. As a result of encoded, new attributes were added to the dataset as results of the encoder.

Warehouse block attribute created five new attribute such as Warehouse block A, Warehouse block B, Warehouse block C, Warehouse block D, and Warehouse block F. Mode of Shipment attribute created three new attributes such as Mode of Shipment Road, Mode of Shipment Ship, and Mode of Shipment Flight. Product Importance attribute created three new attributes such as Product Importance low, Product importance medium, and Product importance high. Gender was encoded using Label Encoder. The attributes that have been encoded will change from categorical to numerical so that the attributes can be used to develop the models. After the attributes were encoded, the dataset had 18 attributes.

Scaling: To enhance the performance of models, values of different numerical attributes need to be scaled so the attributes have a similar range of value. Standardization or Z-Score Transformation is a technique to rescale the values of attributes so the attributes had a standard normally distributed data with a zero mean and standard deviation of one. The formula of transformation by StandardScaler, Standardization method can be written as:

$$Z = \frac{(x-u)}{s}$$

where, Z is standardized attribute value, x is the original attribute value, u is mean of attribute, and s is standard deviation of attribute.

In this study, StandardScaler was used for training and testing datasets that had attributes with different scales of value than other attributes such as Cost of the product and Weight in gms. Both attributes had different range of values than others attributes. Both attributes had range of values at hundreds and thousand while the others had range at tens. After both attributes were scaled, the output of scaling was reshaped within range (-1, 1) so fit and transform can be applied to the output.

E. Model

The dataset that passed preprocessing steps were applied to the models. This study used machine learning algorithms to develop models to predict whether the product reached on time or not. Three algorithms were used such as Decision Tree, Random Forest, and Gradient Boosting.

- Decision Tree

Decision Tree is a machine learning algorithm represented as tree structures that are used to make decisions or predictions. The algorithms had a structure similar to a tree where a root as a node of starting point, each branch represents the output of the model, each leaf represents the decision or prediction, and internal node represents an attribute of the dataset. The algorithm selected the best attribute to be splitted based on metric and the forming of decision tree involves recursively partitioning the dataset[8]. The metric used in this algorithm are calculated as:

Entropy: It measures the amount of uncertainty in the dataset.

$$Entropy = -\sum_{i=1}^{n} pi * log_2(pi)$$

where, pi is the probability of an instance being classified into a particular class.

Gini: It measures the incorrect classification of a new instance.

$$Gini = 1 - \sum_{i=1}^{n} (pi)^2$$

where, pi is the probability of an instance being classified into a particular class.

Gain: It measures the reduction in entropy after the dataset split.

$$Gain = Entropy_{Parent} - \sum_{i=1}^{n} (\frac{xi}{x} * Entropy(xi))$$

Where, $Entropy_{Parent}$ is the entropy of the entire dataset (x), xi is the subset of the dataset, Entropy(xi) the entropy of the subset.

- Random Forest

Random Forest is a machine learning algorithm that uses ensemble learning methods. Ensemble learning algorithm is an algorithm that combines multiple models trained at the same time with different subset of training dataset. Random Forest is an algorithm that combines multiple decision trees. Random Forest created multiple decision trees to perform on different subsets and the output was combined from all output of each tree as the final prediction[9]. The output for classification tasks often used mode from output of each tree and can be calculated as:

$$Y = mode\{T_1(x), T_2(x), ... T_a(x)\}$$

where, Y is the predicted output, $T_a(x)$ is the prediction made by the a-th for the input x.

- Gradient Boosting

Gradient Boosting is an ensemble learning algorithm in machine learning . Gradient Boosting uses various algorithms that consider weak learner algorithms such as decision trees, linear models, and similar models to create a strong learner model. This algorithm creates a model to be trained, then iteratively adding a new model to correct the first model error. The process continues until the dataset is predicted correctly or the maximum number of models are added. This algorithm is less sensitive to outliers and can handle imbalanced data so high accuracy and robust output can be achieved[10].

F. Tuning Hyperparameter

Hyperparameter is a parameter of a model that can be adjusted to before the training phase. Tuning

hyperparameters was applied to the models to enhance and increase the performance of models. GridSearchCV and RandomizeSearchCV are two methods of tuning that were applied to the models. Both methods were used to identify hyperparameters that can boost performance for models. The results showed that five parameters were the best combination for the models. Table 2. is a detailed description of the parameters.

Table 2. Description of parameters

No	Parameters	Description
1	learning_rate	A parameter that refers to how quickly a model can learn from the dataset. The range of learning_rate within (0.0) to (1.0).
2	max_depth	A parameter that refers to the maximum depth of a tree.
3	min_sample_leaf	A parameter that refers to the minimum of needed samples to be at a leaf node.
4	min_sample_split	A parameter that refers to the minimum of needed samples to split an internal node.
5	n_estimators	A parameter that refers to numbers of boosting stages or iteration that were used to perform.

The hyperparameter was applied to models to see the performance of each model. Evaluated each performance using evaluation matrixes and repeated the process to check if the results showed any improvement.

G. Evaluation Matrix

Accuracy - A metric to measure the proportion of correct prediction made by the model.

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Ne}$$

Precision - A metric to measure the proportion of true positive predictions among all positive predictions made by the model

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall - A metric to measure the proportion of true positive predictions among all actual positive instances.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1 Score - A metric to measure the balance between Precision and Recall.

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

IV. RESULT AND DISCUSSION

This research used three machine learning models—Decision Tree, Random Forest, and Gradient Boosting to predict whether shipments would arrive on schedule. We used data from an international e-commerce corporation that sells electronics[6]. Consumer interactions, product attributes, shipping methods, and the demographics of the consumer base were all important components.

A. Model

Results from Decision Tree Classification

```
Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.56      0.57      0.56       895
           1       0.70      0.69      0.70      1305

    accuracy                           0.64      2200
   macro avg       0.63      0.63      0.63      2200
weighted avg       0.64      0.64      0.64      2200
```

Fig . 1. Decision Tree Classification Result

The Decision Tree model has an accuracy rate of 64%. Although its overall performance was good, its precision and recall ratings of 0.70 and 0.69 indicated that it was especially good at predicting llate deliveries (class 1). A solid balance between recall and accuracy was shown by the F1-score of 0.70 for late deliveries. In the case of on-time delivery (class 0), the model's accuracy, recall, and F1-score were 0.56, 0.57, and 0.56, respectively.

Results from Random Forest

```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.56      0.68      0.62       895
           1       0.74      0.64      0.69      1305

    accuracy                           0.66      2200
   macro avg       0.65      0.66      0.65      2200
weighted avg       0.67      0.66      0.66      2200
```

Fig . 2. Random Forest Result

At 66% accuracy, the Random Forest model outperformed the Decision Tree. When it came to class 1 predictions (late deliveries), it did better; recall was 0.64, accuracy was 0.74, and F1-score was 0.69. For deliveries that occurred on schedule (class 0), the model produced an F1-score of 0.62, a recall of 0.68, and an accuracy of 0.56. It's amazing how well the Random Forest model detected on-time delivery, outperforming the Decision Tree model with a higher recall for class 0.

Results from Gradient Boosting

```
Gradient Boosting Classification Report:
              precision    recall  f1-score   support

           0       0.58      0.86      0.69       895
           1       0.86      0.57      0.68      1305

    accuracy                           0.69      2200
   macro avg       0.72      0.72      0.69      2200
weighted avg       0.74      0.69      0.69      2200
```

Fig . 3. Gradient Boosting Result

The best performing model was the Gradient Boosting model, which attained the highest accuracy of 69%. It achieved an F1-score of 0.69 by accurately predicting on-time delivery (class 0) with an accuracy of 0.58 and a high recall of 0.86. Its precision was 0.86 and recall was 0.57 for late deliveries (class 1), resulting in an F1-score of 0.68. This model demonstrated its capacity to handle complicated data patterns by successfully balancing recall and precision for both classes.

B. Tuning Hyperparameter

The best performing model in our experiment was the Gradient Boosting model. To enhance the performance, we used two hyperparameter tuning methods: GridSearchCV and RandomizedSearchCV.

GridSearchCV performs a thorough search over a provided parameter grid.. Table 3 details the best parameters that we got.

Table 3. Best parameter using GridsearchCV.

Parameter	Best Value
learning_rate	0.05
max_depth	3
min_samples_leaf	1
min_samples_split	2
n_estimators	50

With these best parameters, the performance of the Gradient Boosting model was:

```
Gradient Boosting Classifier Classification Report (GridSearchCV):
              precision    recall  f1-score   support

           0       0.57      0.95      0.71       895
           1       0.94      0.52      0.67      1305

    accuracy                           0.69      2200
   macro avg       0.75      0.73      0.69      2200
weighted avg       0.79      0.69      0.69      2200
```

Fig . 4. Result using GridsearchCV

After tuning with GridSearchCV, the Gradient Boosting model maintained an overall accuracy of 69%. The precision for on-time deliveries (class 0) was 0.57 and the recall was significantly high at 0.95, with an F1-score of 0.71. For late deliveries (class 1), the precision was 0.94 and the recall was 0.52, resulting in an F1-score of 0.67. The tuning improved recall for on-time deliveries significantly.

RandomizedSearchCV performs a random search over a specified parameter distribution. Table 4 details the best parameters that we got.

Table 4. Best parameter using GridsearchCV.

Parameter	Best Value
learning_rate	0.05
max_depth	3
min_samples_leaf	1
min_samples_split	5
n_estimators	50

With these best parameters, the performance of the Gradient Boosting model was:

```
Gradient Boosting Classifier Classification Report (RandomizedSearchCV):
              precision    recall  f1-score   support

           0       0.57      0.95      0.71       895
           1       0.94      0.52      0.67      1305

    accuracy                           0.69      2200
   macro avg       0.75      0.73      0.69      2200
weighted avg       0.79      0.69      0.69      2200
```

Fig . 5. Result using RandomizedSearchCV

The results of RandomizedSearchCV tuning were similar to GridSearchCV. The model's overall accuracy stayed at 69%. Precision and recall were constant across both classes, with on-time delivery (class 0) having a precision of 0.57 and a recall of 0.95, and late deliveries (class 1) having a precision of 0.94 and a recall of 0.52.

C. Discussion

Our experiments found that the Gradient Boosting model had a highest accuracy of 69% in predicting on-time shipment delivery, even after significant modification. We used two hyperparameter tuning strategies, GridSearchCV and RandomizedSearchCV, to improve the model's performance. The optimum parameters determined using these approaches produced consistently accurate results. Despite these efforts, the accuracy did not reach 70%, exposing various issues with the dataset.

We found multiple challenges, including data imbalance, which was not entirely resolved by using SMOTE[11] to balance the classes, resulting in biased model predictions. The dataset's characteristics may not fully capture all aspects driving shipment delays, including important factors such as real-time traffic conditions, inspection delays, and special handling needs.

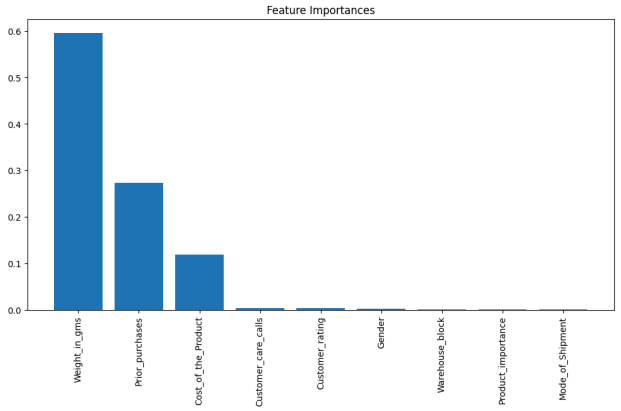

Fig . 6. Feature importances from the dataset.

The analysis of feature importance reveals important insights into the factors affecting shipment delivery times. The weight of the shipment is the most influential factor, which aligns with practical expectations as heavier shipments might require more handling and longer transportation times. This high importance score suggests

that any logistical improvements targeting the handling of heavy shipments could significantly impact delivery efficiency.

The number of prior purchases by a customer is another significant predictor. Customers with frequent purchases might have a more streamlined process in place, or their shipments might be prioritized, reflecting better on-time delivery rates.

V. CONCLUSION

The Gradient Boosting model achieved the best accuracy in predicting on-time shipment delivery, with a score of 69%. Despite our efforts to balance the dataset using SMOTE, and fine-tune the model with GridSearchCV and RandomizedSearchCV, the accuracy did not reach 70%. It means that the dataset has inherent issues that must be addressed in order to progress further.

The feature importance revealed that the weight of the shipment and the number of prior purchases are the most influential factors in predicting delivery times. However, several features, such as customer care calls, customer rating, gender, warehouse block, product importance, and mode of shipment, showed minimal importance. This suggests that the dataset might be missing or need more variables that significantly impact delivery times, such as real-time traffic conditions, customs delays, or specific handling requirements.

Overall, the Gradient Boosting model proved to be the most reliable among the other tested models; however, the dataset's issues emphasize the need for a more comprehensive and cleaner data gathering strategy to attain greater forecast accuracy in shipment delivery timeframes. Future work should focus on incorporating additional relevant features and ensuring a representative sample to better capture the dynamics of shipment deliveries.

REFERENCES

[1] A. Jonquais and F. Krempl, “Predicting Shipping Time with Machine Learning,”2019.Available:https://dspace.mit.edu/bitstream/handle/1721.1/121280/Jonquais_Krempl_2019.pdf?sequence=1&isAl

[2] A. Rokoss, et al., “Case study on delivery time determination using a machine learning approach in small batch production companies,” Journal of Intelligent Manufacturing, Jan. 2024, doi: https://doi.org/10.1007/s10845-023-02290-2.

[3] A. Ermagun, A. Punel, and A. Stathopoulos, “Shipment status prediction in online crowd-sourced shipping platforms,” Sustainable Cities and Society, vol. 53, p. 101950, Feb. 2020, doi: https://doi.org/10.1016/j.scs.2019.101950.

[4] S. Shahid, “Predicting Delays In Delivery Process Using Machine Learning-Based Approach,” Dec. 2020, doi: https://doi.org/10.25394/pgs.13350764.v1.

[5] S. Hathikal, S. H. Chung, and M. Karczewski, “Prediction of ocean import shipment lead time using machine learning methods,” SN Applied Sciences/SN Applied Sciences, vol. 2, no. 7, Jun. 2020, doi: 10.1007/s42452-020-2951-5.

[6] Nayana Ck, “Shipping,” Kaggle.com, 2024. https://www.kaggle.com/datasets/nayanack/shipping

[7] J. Gong and T. Chen, “Learning Software Performance? An Empirical Study on Encoding Schemes,” 2022, doi: https://doi.org/10.1145/3524842.3528431.

[8] “sklearn.tree.DecisionTreeClassifier — scikit-learn 0.22.1 documentation,” Scikit-learn.org, 2019. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[9] Scikit-learn, “sklearn.ensemble.RandomForestClassifier — scikit-learn 0.20.3 documentation,” Scikit-learn.org, 2018. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[10] “3.2.4.3.5. sklearn.ensemble.GradientBoostingClassifier — scikit-learn 0.20.3 documentation,” Scikit-learn.org, 2009. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

[11] “SMOTE — Version 0.9.0,” imbalanced-learn.org. https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html