Real-Time Delivery Time Forecasting and Promising in Online Retailing: When Will Your Package Arrive?

Nooshin Salari
Rotman School of Management, University of Toronto
Department of Industrial Engineering and Operations Research, University of California, Berkeley, nsalari@mie.utoronto.ca

Sheng Liu
Rotman School of Management, University of Toronto, sheng.liu@rotman.utoronto.ca

Zuo-Jun Max Shen
Department of Industrial Engineering and Operations Research and Department of Civil and Environmental Engineering
University of California, Berkeley, maxshen@berkeley.edu

Problem definition: Providing fast and reliable delivery services is key to running a successful online retail business. To achieve a better delivery time guarantee policy, we study how to estimate and promise delivery time for new customer orders in real time. **Academic/practical relevance:** Delivery time promising is critical to managing customer expectations and improving customer satisfaction. Simply over-promising or under-promising is undesirable due to their negative impacts on short-term/long-term sales. We are the first to develop a data-driven framework to predict the distribution of order delivery time and set promised delivery time to customers in a cost-effective way. **Methodology:** We adapt regression tree and quantile regression forests to generate distributional forecasts by exploiting the complicated relationship between delivery time and relevant predictors, which include the queue-length predictors to model the distribution center operations. We further propose a cost-sensitive classification decision rule to decide the promised delivery day from the predicted distribution. **Results:** Tested on a real-world data set shared from JD.com, our proposed machine learning based models deliver superior forecasting performance. In addition, we demonstrate that our framework has the potential to provide better promised delivery time in terms of both cost and accuracy, as compared to the conventional promised time set by JD.com. **Managerial implications:** Through a more accurate estimation of the delivery time distribution, online retailers can strategically set the promised time to maximize customer satisfaction and boost sales. Our data-driven framework reveals the importance of modeling fulfillment operations in delivery time forecasting, which sheds light on further improvement directions.

Key words: logistics; online retail; forecasting; machine learning.

1. Introduction

Logistics performance has been a critical success factor for e-commerce. Consumers weigh the value of the product against the delivery service quality before making purchase decisions. According to a 2019 survey on online shoppers' shipping experiences and preferences, 44% of online shoppers have abandoned a shopping cart simply because the selected items would not arrive on time (Lauren Freedman 2019). In addition to delivery speed, consumers also value delivery reliability (whether orders can arrive on or before the promised day). It has been shown that consumers punish deliveries arrived later than the promised time, and the resulting dissatisfaction can translate to a lower repurchase rate (Chan et al. 2018). Moreover, delayed deliveries may cause higher product returns, which further cut down online retailers' profit margin (Cui et al. 2020). Consequently, providing fast and reliable delivery services is a central task for online retailers in nowadays competitive markets.

In addition to improving delivery performance by investing in logistics infrastructure (e.g., the extensive fulfillment network built by JD.com and Amazon), disclosing delivery information and promising delivery time to customers is becoming an important lever to manage customer expectations and improve customer satisfaction (Cui et al. 2020). More specifically, the promised delivery time shown at order checkout not only affects the conversion rate but also shapes the customer expectation. Simply adopting an over-promising or under-promising strategy may not necessarily benefit the online retailer. When a retailer over-promises customers on delivery time (i.e., provides faster delivery promises), more customers will be attracted, and higher sales can be generated. However, over-promising poses a higher risk of delivery delays, which would cause a loss of customer goodwill and result in fewer future purchases (Rao et al. 2011). On the other hand, under-promising reduces the risk of delivery delays but signals longer delivery time (slower delivery speed) to customers, which may result in lower sales as customers seek out a competitor who promises faster deliveries. Therefore, the delivery time promise policy should be appropriately designed to help manage customer expectations and drive more sales.

Similar to the newsvendor model where the optimal order quantity relates to the demand distribution, the optimal delivery time promise policy hinges on modeling the delivery time distribution. In online retailing, delivery time is subject to volatility and uncertainty from order arrivals, inventory evolution, warehouse operations, and transport operations (e.g., last-mile delivery). As a result of the complexity of order fulfillment and delivery processes, and the aforementioned uncertainties, predicting the delivery time distribution is a challenging task.

In this paper, we study the delivery time distribution forecasting problem: how to generate an accurate distributional estimation of the delivery time for new customer orders in real time? Based on the distributional forecasts, we also address the problem of setting promised delivery time to customers in a cost-effective manner. We aim to develop a framework that is adaptive, scalable, and easy to implement by practitioners. We build and test the model on the JD.com data set shared from the 2020 MSOM Data Driven Research Challenge.

1.1. JD.com and Its Logistics Operations

JD.com is China's largest online retailer with a net revenue of US$67.2 billion in 2018 and over 320 million annual active customers (Shen et al. 2019). JD.com is well known for its high-quality delivery services in terms of both speed and reliability, which can be attributed to its extensive fulfillment network. According to the survey performed by the State Post Bureau of the People's Republic of China, JD.com is consistently ranked in top positions based on consumer satisfaction and on-time performance (State Post Bureau of the People's Republic of China 2020).

Before the order placement (checkout), an estimated delivery time (i.e., promised delivery time) will be provided to the customer. Once the order is confirmed, the order will be processed through the fulfillment system. The process of order fulfilment varies among online retailers, for instance, after an order is received by Amazon, the inventory level is checked and one of the domestic fulfillment centers with the least cost is assigned to fulfill the order (Hsu et al. 2015). The order fulfillment strategy of JD.com is complicated as it aims to optimize the overall system performance while satisfying delivery promises, which adds to the complexity of delivery time estimation (Shen et al. 2019). According to the optimized fulfillment strategy, the order will be picked, packed, and shipped at a distribution center (hereafter referred to as DC) in the same or different district from the customer location.

The JD.com data set provides detailed information about product characteristics, orders, deliveries, customers, and the fulfillment network (however, the inventory information is relatively limited). From this data set we are able to extract the actual delivery time of each placed order, which is formally defined as the difference between the actual arrival time at the customer home and the order placement time: *delivery time* = order arrival time - order placement time (delivery time is also referred to as the delivery speed in some papers and we use them interchangeably). By harnessing the power of machine learning, we are able to exploit the relationship between delivery time and relevant predictors.

1.2. Our Contributions

To the best of our knowledge, we are the first to develop distributional forecasting models (beyond point estimates) to predict the distribution of parcel delivery time conditional on predictor variables in an online retailing context. These predictors help capture the effects from order characteristics (such as order time and quantity), customer profiles, and distribution center operations. Motivated by the service operations literature, we model distribution center operations using fluid models and construct relevant queue-length predictors. Because these predictors can be updated online, we are able to generate distributional forecasts for new customer orders in real time.

Observing that the distribution of delivery time is dynamic and multimodal, we propose to use machine learning based approaches that provide nonparametric models to exploit the nonlinear relationship between predictors and delivery time. Two tree-based models, regression tree and quantile regression forests, are adapted to generate distributional forecasts, which are well suited for practical implementations as they are flexible, scalable, and interpretable. Tested on a large data set from JD.com, the two models provide superior forecasting performance as measured with various scoring rules. Specifically, the proposed quantile regression forests improve the forecasting accuracy by 41%-66%, as compared to an online empirical forecasting benchmark.

Furthermore, we develop a classification decision rule to facilitate the effective promised delivery time management. Considering different costs associated with late deliveries and early deliveries, we design a framework that transforms the distributional forecasts to promised delivery days in a classification setting. Our proposed classification decision rule can be trained efficiently to minimize the expected misclassification cost. Evaluated on the JD.com data set, we demonstrate the effectiveness of our framework and it's potential to achieve considerable improvements over the conventional promised time set by JD.com in both cost and accuracy.

2. Related Literature

Our paper contributes to the growing stream of papers on service quality forecasting in various operational contexts. In particular, wait time (delay) prediction has been studied in different service systems. Ibrahim and Whitt (2009a) compare queue-length delay estimators (computed as the ratio between queue length and processing rate) to various estimators that are based on recent delay history such as the delay of the last customer to enter service (LES). Through simulation studies, it is shown that queue-length based methods often outperform the delay history based estimators in simple and complex queueing systems (Ibrahim and Whitt 2009b, 2011). Built upon the queue-length based estimators, Ang et al. (2016) apply the least absolute shrinkage operator (Lasso) and propose the so-called Q-Lasso model for wait time prediction in the emergency department. However, these papers are focused on point forecasts, while distributional forecasts in service systems have received less attention until recently. Shang et al. (2017) propose to use a Bayesian nonparametric model to estimate the distribution of air cargo transport risks. Guo et al. (2019) study the airport transfer passenger flow forecasting problem, and develop a regression tree based method for distributional forecasts of passenger connection times. Most recently, Arora et al. (2020) apply regression forests to generate distributional forecasts of wait time in an emergency department. As far as we know, the delivery time forecasting problem has not been studied in the real-world online retailing setting. Motivated by the above literature, we propose to incorporate the queue-length predictors (from distribution centers) into delivery time predictive modeling.

Our paper is also related to the literature on understanding the value of delivery service quality in online retailing. It is expected that providing fast and reliable delivery services can positively impact customer demand while failing to deliver a high-quality delivery service often causes customer churns. On the speed side, Fisher et al. (2019) show that one business-day reduction in delivery time can translate to, on average, 1.45% of increase in online sales, based on a quasi-experiment conducted on a US apparel retailer. Cui et al. (2020) analyze the effects of varying the delivery speed disclosure policy on an online photo retailer. They find that promising customers one day faster shipping increases sales by 0.73% and profits by 2.0% while promising customers one day slower decreases sales by 0.51% and profits by 2.7%, which highlights the importance of managing the delivery speed disclosure policy adequately. On the reliability side, Rao et al. (2011) empirically show that the order fulfillment delay results in a higher customer anxiety level and decreased future order frequency and size. Chan et al. (2018) find that consumers penalize late deliveries and reward early deliveries (in terms of review scores), but the delay penalty is much larger than the early arrival reward. In addition, the information flow on the delivery tracking record also matters for customer satisfaction: Bray (2019) shows that postponing the tracking activities can yield higher delivery scores. The overall quality of delivery service is highlighted in Cui et al. (2019), where the authors find that the elimination of a high-quality delivery option is highly detrimental to the online retailer, in both the short-term and long-term. Our paper complements this literature by developing real-time delivery time (speed) distributional forecasting methods, which can serve as a stepping stone to better management of delivery promise and quality.

3. Data Overview and Preprocessing

The transactional data from JD.com is provided for $9,159$ different stock-keeping units (SKUs) shipped from 56 different districts to 60 districts for the month of March in 2018. The transaction-level data contains the information associated with each customer order such as order quantity for each SKU, the date and time when the order was placed, the promised delivery time of the order, shipment, inventory, etc. The data also provides information regarding product pricing and promotional activities (coupons and discounts), which varies across SKUs and users over time. For all SKUs, the availability of the inventory at the end of the day is known rather than the amount of inventory.

There are two types of SKUs on JD.com: first-party owned (1P) and third-party owned (3P) SKUs that are differentiated by their inventory ownership. All 1P SKUs are managed by JD.com, including shipment, inventory replenishment, and deliveries. For 3P SKUs, the merchants can decide whether to use the logistics services provided by JD logistics or not. The promised delivery time information is not available for most orders involving 3P SKUs.

Each order is associated with two warehouse (DC) identifiers: dc-ori and dc-des, where dc-ori indicates the warehouse from which the order was shipped, and dc-des represents the district to which the order was shipped (the district is coded by the nearest warehouse to the customer's designated shipping address). Depending on the order fulfillment decision, an order can be fulfilled by the warehouse closest to the customer (dc-ori = dc-des) or some other warehouses in a different district (dc-ori $\neq$ dc-des).

The data set also offers basic customers demographic information including gender, age, residence city level, education, and marital status. Membership and aggregated past purchase information are reported as well.

3.1. Preprocessing

After matching each order with its delivery information, and dropping orders with missing shipment information, we obtained 293,204 unique orders from 1,784 SKUs. This includes 244,333 orders from 1P SKUs and 48,871 orders from 3P SKUs.

We removed any single-item orders that only include a gift item because those orders might contain other items (from other categories) for which the data is not available (13,055 orders). The transactional data for the first day was filtered out, as some of our proposed predictors require recent history information. We also excluded orders with multiple packages (22 orders). For every DC, the transactional data was cleansed for errors on extremely long delivery time that is greater than the 99% quantile (in terms of days), and negative delivery time. We further restrict our attention to the 20 most popular DCs that received more than 75% of total orders to make sure we have enough observations for each of them (this is mainly due to the fact that the provided data set only includes one category of SKUs. In practice we expect that each DC will have sufficient data for training.). The resulting data set contains 221,290 orders (observations) for 1,493 SKUs.

We split the data into two parts: a training set that includes all orders placed from day 2 to the end of day 25, and a test set consisting of observations from day 26 to day 31. There exist observations with no promised delivery time that were used during the training, however later we excluded those observations from the test set so we can compare our approach with the one proposed by JD.com.

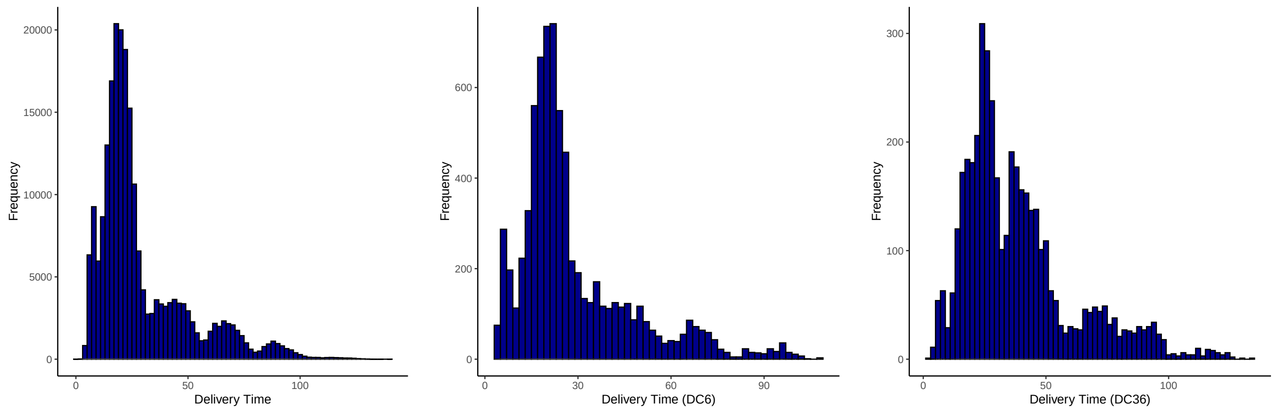
3.2. Summary Statistics and Observations

For the 221,290 processed observations, the average delivery time across the chosen distribution centers is 29.4 hours and the standard deviation is 20.8 hours. Additional summary statistics about delivery time are presented in Table 1. Among these orders, 56.8% have one-day delivery promises, 28% have two-day delivery promises, and 14.7% have delivery promises longer than two days (0.5% have no delivery promises).

Table 1 **Summary Statistics of Delivery Time (in hours)**

Mean	Median	Standard deviation	25% Quantile	75% Quantile
29.4	22.0	20.8	16.4	37.0

Distribution of delivery time (in hours) for all selected distribution centers and two specific distribution centers (DC 6 & 36) are shown in Figure 1. We find that the delivery time distributions are non-symmetric and multi-modal. Although most observations have delivery time less than 50 hours, there is a long tail that may stretch to more than 100 hours. Furthermore, different DCs can exhibit different distributional behaviors, as shown in Figure 1b and 1c. These features are also present at different aggregation levels in days or SKUs. Based on these observations, traditional parametric distributional models may not be suitable for our application.



(a) All distribution centers (b) DC 6 (c) DC 36

Figure 1 **Histogram of delivery time (in hours) for (a) all distribution centers, (b) distribution center 6, and (c) distribution center 36**

We also compare the actual delivery time with the promised delivery time from JD.com (as presented in the data set). Because the provided promised delivery time is measured in days, we transform the actual delivery time (in hours) into delivery days in a way that is consistent with JD.com: a delivery on the k-th day following the order placement indicates a k-day delivery while a delivery within the same day is still coded as a one-day delivery.[1] Figure 2 summarizes the comparison results. We observe that 98% of orders were shipped within one day, of which 70% were delivered to customers in one day. Among the orders that were delivered within one day, 36% of

[1] Based on Shen et al. (2019), promised one-day delivery may indicate either same-day delivery or next-day delivery depending on the order placement time. Order placed before 11 am (e.g., 10 am) can be provided with same-day delivery services (e.g., delivery time $\leq$ 14 hours), while orders received after 11 am will be provided next-day delivery services.

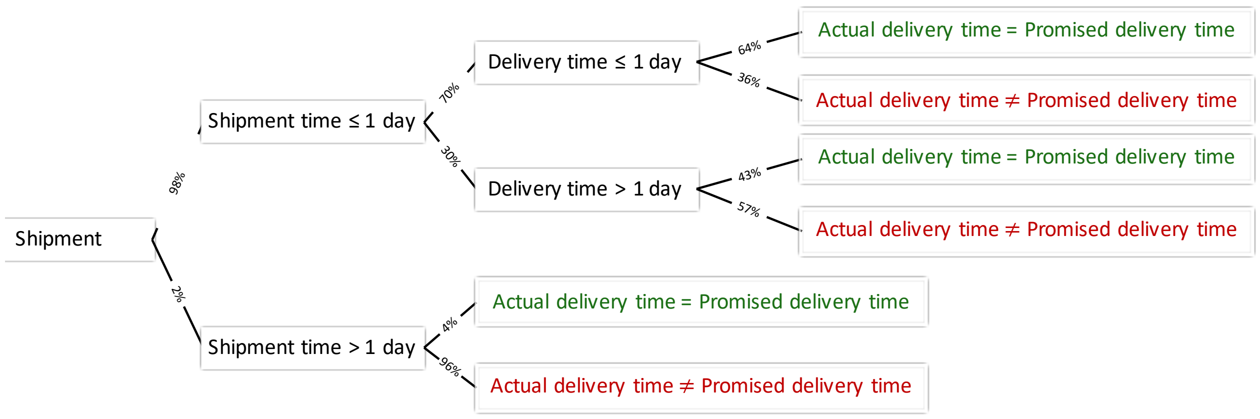


Figure 2 **Percentage of mismatched promised delivery time (including both early deliveries and late deliveries)**

them have mismatched promised delivery time (actual delivery time $\neq$ promised delivery time, in terms of days). In contrast, for orders shipped but not delivered within one day, 57% of them have mismatched promised delivery time. Furthermore, when an order was not shipped in one day, the promised delivery time could hardly match the actual delivery time, which implies that potential shipment delays are not well captured by JD.com's current delivery time promising policy. The high mismatch rate stresses the need to develop a reliable approach to setting promised delivery time, which should be based on an accurate distributional forecasting model.

4. Predictor Variables Construction and Selection

Predictor variables selection (also known as feature engineering) plays a vital role in machine learning and predictive modeling. Because order fulfillment and delivery is a complicated process consisting of multiple stages (e.g., picking, packing, shipping, and last-mile delivery), we have explored various types of candidate features in the data set based on our interactions with the industry experts. Moreover, the distribution center can be viewed as a large complex service system, for which we can exploit the use of queue-length (or fluid model) based predictors. The main classes of candidate features are summarized as follows (a detailed description of features is provided in Appendix A):

1. **Time effect**. We consider both the hour of the day and the day of the week to capture the variation of delivery time across different hours and days. The delivery time can be lower during working hours of the day.

2. **Distribution center operations**. The first stage of order fulfillment takes place at the distribution center. The fulfillment delay occurred in a distribution center will propagate to delivery

delay at the final stage. As in a typical service system, the "wait time" of a new order (i.e., time-to-shipment) in the DC can be estimated as (Ang et al. 2016):

$$\frac{\text{Workload}}{\text{Processing Rate}},$$

where workload refers to the work required to process the existing orders in the DC (received but not yet shipped), and processing rate is associated with the staff level. Therefore, we construct workload and processing rate predictors to capture the wait time effects (we do not directly use the ratio of workload and processing rate to allow more flexibility). Based on the data, we measure the number of orders waiting to be shipped at the time of an order placement to represent the workload (note that the data lacks detailed timestamps for picking or packing, so we can not construct the corresponding features). Because the data does not reveal the staff or processing capacity information, we calculate the number of orders shipped in the last two hours as a proxy for the processing rate. Furthermore, we consider SKU-specific workload and processing rate to incorporate the potential heterogeneity among different SKUs.

In order to utilize the DC related predictors in real time prediction, we need to know which DC will process a new customer order. As discussed before, the order fulfillment strategy of JD.com is potentially complicated and the inventory levels at different DCs can be critical to the fulfillment decision (see, e.g., Acimovic and Graves 2015). Unfortunately, the real-time inventory information is not available, and the exact inventory levels can not be identified from the provided data. As a result, instead of leveraging the inventory information, we propose two different approaches to estimate the DC for a new order (on the test set): a) full-history estimation: we estimate the DC based on the observations in the training set, wherein a DC is selected with the highest probability of satisfying orders associated with a SKU (in the new order); b) recent-history estimation: we perform an online estimation based on the most recent observations. Specifically, we look at the last 5 or 10 orders containing the given SKU, and the DC that satisfies most of these orders is selected as the estimation. We also provide a perfect-information benchmark where we assume the DC is known (before order checkout) for every new order in real time, which may be possible if the order fulfillment algorithm provides updated outputs continuously (e.g., through online simulations).

3. **Order effect**. This class of predictors captures the characteristics of an order that may impact delivery time such as the number of items (SKUs), order size (quantity), order type (1P or 3P), and whether the order includes a gift item. The delivery time may be higher for orders with more SKUs and larger quantities.

4. **User effect**. The customer profiles may affect the delivery time as well. The order fulfillment process may prioritize certain customers over others, e.g., customers with a PLUS membership or higher past purchase values.[2]

To further identify useful predictors and improve model's performance, we apply least absolute shrinkage selection operator (Lasso) regression to filter out irrelevant predictors. Lasso regression introduces an ℓ_1-norm penalty term of coefficients in the loss function, which would shrink the coefficients of irrelevant predictors to zero (Friedman et al. 2001). As a result, only informative predictors with nonzero coefficients are included in our predictive modeling.

5. The Forecasting Model and Classification Rule

In this section we introduce the main distributional forecasting models, a regression tree based model and a quantile regression forests model, that can efficiently utilize relevant predictors to generate conditional distributional forecasts of delivery time. Then we propose a cost-sensitive classification rule for disclosing promised delivery time to customers.

5.1. Regression Tree

Regression tree is a simple nonparametric regression approach used to predict a target variable through binary recursive partitioning. Regression trees can uncover non-linear relationships in data by creating a hierarchical tree of branching rules to predict the target variable. The algorithm for regression tree automatically decides on the splitting variables, split points, and shape of the tree. Suppose that for N observations with p predictors (x_i, y_i) for $i = 1, 2, ...N$, with $x_i = (x_{i1}, ...x_{ip})$, the feature space is partitioned into M non-overlapping regions $R_1, ..., R_M$, and the response is modeled as a constant in each region for a sample x (a region is corresponding to a leaf node in the tree structure):

$$\hat{y}(x) = \sum_{m=1}^{M} c_m I(x \in R_m). \tag{1}$$

where c_m is a constant value that the model predicts for an observation that falls in R_m. If the goal is to minimize the sum of squared errors, then the best value c_m^* is the average of y_i in region R_m. Greedy algorithm is often applied to find the best binary partition in a top-down fashion (we refer interested reader to Friedman et al. (2001) for more details about regression trees).

[2] PLUS membership is a subscription-based program that provides certain benefits varying from free shipping to exclusive discounts (Shen et al. 2019).

By its design, regression tree has been mainly used to generate point estimates. However, it can be extended to derive distributional forecasts conditional on x. The idea is to extract the observations that fall in the same region with x, which is denoted by $R(x)$. As such the empirical distribution of $\{y_i : x_i \in R(x)\}$ can be used to approximate the conditional distribution $F(y|x)$. Specifically, we can estimate the conditional probability distribution of delivery time using a regression tree in two steps: (1) A regression tree is first fitted to the training set using delivery time as the target variable; (2) Then a nonparametric distribution is estimated for observations in each leaf node by a kernel method. The kernel estimation in the second step has been shown to improve the prediction performance in Guo et al. (2019). Consequently, each region R_m is associated with a kernel density function (as a proxy for the empirical distribution), which will be used for forecasting delivery time of new customer orders. To avoid overfitting to the training set, the maximum depth of the final tree (maxdepth) and the minimum number of observations in any terminal node (minbucket) are tuned using a three-fold cross validation by minimizing the average pinball loss over the five quantiles: $2 \leq maxdepth \leq 7$ with an increment of 1; and $10 \leq minbucket \leq 30$ with an increment of 10 (the pinball loss measures the distributional forecasting accuracy and will be discussed later in Section 6.1.1).

5.2. Quantile Regression Forests

In the regression context, random forests, as introduced by Breiman (2001), are a powerful class of machine learning models. It applies random sampling of observations (i.e., bagging) and features to create a set of regression trees to improve the out-of-sample performance. Hence random forests can be viewed as an ensemble learning method that combines multiple learners together. Similar to regression trees, random forests were primarily designed for point estimates (e.g., conditional mean). Meinshausen (2006) generalizes the idea of random forests to generate consistent nonparametric estimates of conditional quantiles of the target variable, which is called quantile regression forests (QRF). The key difference between QRF and random forests is that random forests estimate the conditional mean and neglects all other information, while QRF approximate the full conditional distribution by keeping all observations in every leaf node. Therefore, we can estimate the empirical distribution for leaf nodes cross all trees of the forests.

To apply QRF in distributional forecasting, we first fit QRF to the training data with delivery time as the target variable. Then we estimate the conditional distribution $F(y|x)$ by the empirical distribution of the observations in all leaf nodes (of the forests) associated with x. The tuning parameters are the number of randomly sampled candidate predictors at each split (mtry), number of trees to grow (ntree), and the minimum size of terminal nodes (nodesize). The QRF is tuned by minimizing the average pinball loss using a three-fold cross validation approach, for different combinations of the three parameters: $2 \leq mtry \leq 9$ with an increment of 1; $500 \leq ntree \leq 1000$ with an increment of 500; $nodesize \in \{10, 15, 20, 30\}$.

5.3. Classification Decision Rule and Optimal Threshold

After we obtain the estimated conditional distribution $\hat{F}(y|x)$, we can transform it into the probabilities of delivery days: $\hat{p}(k|x)$ is the estimated probability that the order x will be delivered on the kth day after order placement (in our application the maximum possible number of delivery days is 5, so $k \in \mathcal{K} = \{1, 2, 3, 4, 5\}$). This can be done by discretizing $\hat{F}(y|x)$ properly based on the order placement time.

The estimated probabilities $\{\hat{p}(k|x), k \in \mathcal{K}\}$ form the basis of promised delivery time (in days) management. Setting the promised delivery time can be viewed as a classification problem since the output is a discrete label (e.g., 1-day or 2-day delivery). We can translate the estimated probabilities into a class label using a decision rule. For instance, the usual decision rule that tends to maximize the accuracy is "predict the class with the highest posterior probability", i.e., the classifier predicts that order x arrives on the kth day if $\hat{p}(k|x) > \hat{p}(j|x)$ for $j \neq k$. However, this decision rule is not appropriate when the probability estimates are noisy or different types of classification errors incur different costs to the system (Lachiche and Flach 2003).

In online retailing, incorporating different costs for different types of classification errors is particularly essential. For instance, consider the following three cases :

1. The promised delivery time is 1 day but the order arrives on the 3rd day.
2. The promised delivery time is 2 days but the order arrives on the 3rd day.
3. The promised delivery time is 2 days but the order arrives on the 1st day.

The promised delivery time does not match the actual delivery time in all three cases, so they all incur classification errors. However, the three types of errors bear different costs: the customers in the first case would be more disappointed than in case 2 because of the longer delay, while the customers in the 3rd case can be very satisfied and there is no loss of goodwill. Still, the 3rd type error is not desirable because it indicates the company could have provided a faster delivery promise to attract potentially more customers and stimulate more sales (i.e., there is an opportunity cost). Consequently, the online retailer needs to assign different costs to different types of classification errors and the objective is to minimize the expected cost given by:

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{K}:j\neq k} p(k) r_{kj} c_{kj}, \tag{2}$$

where $p(k)$ is the prior probability of class k, r_{kj} is the the proportion of instances of actual class k that are predicted as class j (also called confusion rates), and c_{kj} is the cost of misclassifying an instance of class k as j. The confusion rates and misclassification costs can be represented as a confusion matrix $\mathbf{M}$ and a cost matrix $\mathbf{C}$, as shown in Table 2. When running on a set of samples (e.g., validation set), the cost function (2) can be evaluated as $\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{K}:j\neq k} m_{kj}c_{kj} / \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{K}} m_{kj}$. A cost-sensitive classification framework aims at minimizing the cost function (2) by a specified classification decision rule (Kuhn et al. 2013, Bernard et al. 2016).

Table 2 Multi-class confusion matrix $\mathbf{M}$ and misclassification cost matrix $\mathbf{C}$: m_{kj} represents the number of instances of class k that are predicted as j and c_{kj} is the cost of classifying an instance of class k as j (we assume $c_{kk} = 0$ without loss of generality).

		Predicted class			
		1	2	...	K
	1	m_{11}, c_{11}	m_{12}, c_{12}	...	m_{2K}, c_{2K}
Actual	2	m_{21}, c_{21}	m_{22}, c_{22}	...	m_{2K}, c_{2K}
class	...	...	...	...	...
	K	m_{K1}, c_{K1}	m_{K2}, c_{K2}	...	m_{KK}, c_{KK}

When there are only two classes (e.g., positive and negative), the decision rule simply decides a threshold on the positive probability: predict a positive class when the positive probability is above the threshold. The optimal threshold in the cost-sensitive framework can be found efficiently by performing an ROC analysis (Received Operating Characteristic), as shown in Lachiche and Flach (2003), Hernández-Orallo et al. (2012), and Bernard et al. (2016). However, in the general multi-class scenario, finding the optimal decision rule is often difficult. The typical decision rule is to predict class according to a weighted posterior probability: $\arg\max_k\{\omega_k \cdot \hat{p}(k|x)\}$, where the weights $\{\omega_k, k \in \mathcal{K}\}$ must be learned to minimize the expected cost. Finding the optimal set of weights is intractable in most cases, and has been proven to be NP-complete (Bourke et al. 2008).

In this paper we propose to use a decision rule that applies a single threshold to the cumulative distribution function (also called accumulated posterior probabilities, see e.g., Ha 1997). Given an ordered set of classes $\mathcal{K}$, let $P_x(j)$ denote the accumulated posterior probability for sample x:

$$P_x(j) = P(y(x) \leq j) = \sum_{k=1}^{j} \hat{p}(k|x). \tag{3}$$

Then we predict a class as $\inf\{j : P_x(j) \geq \tau\}$, where $\tau \in [0,1]$ is a threshold that needs to be tuned. When there are only two classes, this decision rule reduces to the one based on the positive probability threshold. Under the proposed decision rule, we will promise a j-day delivery if j is the smallest integer such that the probability of being delivered within j days is no less than τ. As such, this decision rule has a similar structure to the newsvendor solution, and we can show that it results in the minimum expected cost when the misclassifications costs are linear and estimated posterior probabilities are perfect (the proof is presented in Appendix B):

Proposition 1. *When $c_{kj} = (j-k)c_1$ if $j > k$ and $(k-j)c_2$ otherwise, there exists a decision rule with $\inf\{j : P_x(j) \geq \tau\}$ that minimizes the expected misclassification cost (2) given that the probability estimates $\{\hat{p}(k|x)\}_{\forall k}$ are accurate.*

In practice, the estimated probabilities are noisy, so the threshold from the above proposition may not lead to the best out-of-sample performance, and we decide to learn the optimal threshold τ^* from the training set. In order to avoid overfitting, the last five days in the training set (from day 20 to day 25) are held back as the validation set. We fit QRF to the remaining training data (day 2 to day 20) and estimate the probabilities of delivery days by discretizing the estimated CDF. Then we perform a grid search in $[0,1]$ to find τ^* that minimizes the average cost on the validation set. Consequently, we can apply the proposed decision rule with τ^* to the classification task on the test set.

6. Results and Discussion

In this section we evaluate the forecasting performance of the proposed delivery time forecasting models, in comparison with common benchmarks. We further assess the the power of our proposed classification decision rule in promised delivery time management, and validate its superior performance as compared to the estimates from JD.com.

6.1. Forecasting Performance Comparison

We discuss two types of forecasting accuracy measures: distributional forecasting scoring rules and probabilistic scoring rules based on classification. We also discuss the importance of different predictors in delivery time forecasting.

6.1.1. Distributional Forecasting Accuracy We first consider measuring the distributional forecast accuracy from two scoring rules: pinball loss function and continuous ranked probability score (CRPS). The pinball loss function for q-quantile is defined as

$$\ell_i(y_i, \hat{y}_{i,q}) = \begin{cases} (1-q)(\hat{y}_{i,q} - y_i), & \text{if } y_i < \hat{y}_{i,q} \\ q(y_i - \hat{y}_{i,q}), & \text{otherwise} \end{cases} \tag{4}$$

where q is the reported quantile, y_i is the actual observed delivery time, and $\hat{y}_{i,q}$ is the delivery time forecast at the q-quantile. Pinball loss function is an asymmetric function and an error measure for quantile forecasts, which penalizes low-probability quantiles more for overestimation than underestimation (vice versa for high-probability quantiles). A lower score indicates a better forecasting performance. In this paper we report the pinball loss score for critical quantiles 0.05, 0.25, 0.50, 0.75, and 0.95 as well as average pinball loss across all targeted quantiles, so we can assess the overall distributional fit. The second scoring rule for distributional forecast evaluation is CRPS (Matheson and Winkler 1976), which quantifies the difference between the predicted CDF and the empirical CDF. CRPS measures the gap between the observation y, and $\hat{F}$, as follows:

$$CRPS(\hat{F}, y) = \int_{-\infty}^{\infty} (\hat{F}(z) - \mathbf{1}\{z \geq y\})^2 dz, \tag{5}$$

where $\mathbf{1}\{z \geq y\}$ is the indicator function that equals to 1 if $z \geq y$ and 0 otherwise. It is shown by Gneiting and Raftery (2007) that CRPS can be equivalently expressed as

$$CRPS(\hat{F}, y) = E(|Y - y|) - \frac{1}{2}E(|Y - Y'|), \tag{6}$$

where Y and Y' are two independent random variables with distribution $\hat{F}$, and $E(\cdot)$ denotes the expectation operator with respect to the distribution $\hat{F}$. The first term in (6) represents the absolute error and the second term measures the sharpness of F. Because the analytical form of CDF is not available, we draw 1000 samples to evaluate CRPS using equation (6). Both pinball loss function and CRPS have been widely used in distributional forecast evaluation, see, e.g., Guo et al. (2019) and Grushka-Cockayne et al. (2017).

Motivated by the literature and industry practice, we consider two benchmark distributional forecasting methods:

1. Online empirical forecasting. A simple and popular way to forecast service quality measures (such as wait time) in real time is to utilize the recent history information (e.g., hospitals use moving average for point estimates of wait time, as shown in Dong et al. 2019). In our setting, we can forecast the delivery time distribution by the empirical distribution of observed delivery time for most recent orders with a moving h-hour window. Instead of applying a fixed h-hour window to all orders placed during the day, we adjust the window length based on the order placement time (because we observe that the order density is heterogeneous): A 4-hour moving window is applied for orders placed from 9:00 AM to 11:59 PM (the moving window is extended to 5 hours when the observations are scarce); and for orders placed from 00:00 AM to 8:59 AM, a moving window is constructed with a fixed start time of 7:00 PM of the previous day (the start time is extended to 6:00 PM if the observations are insufficient).

2. Quantile regression. As introduced by Koenker and Bassett Jr (1978), quantile regression is a generalization of the least squares regression that estimates quantiles of the target variable by a set of predictors. It provides more information about the conditional distribution of the target variable and is more robust against outliers in the response measurement. Quantile regression is thus widely used for distribution modeling and has been applied to wait time forecasting in emergency departments (Sun et al. 2012). Because quantile regression estimates different quantiles independently, the estimated quantile curves can cross each other, which leads to an invalid distribution of the response. Consequently, we apply a restricted quantile regression model (with non-crossing constraints) to avoid crossing of quantile curves (please refer to R package "Qtools" for technical details). We then estimate the conditional CDF $\hat{F}(y \mid x) = \inf\{p : Q(p \mid x) \geq y\}$, where Q represents the estimated conditional quantile computed at the values $p \in \{0.01, 0.02, ..., 0.99\}$.

Table 3 summarizes the average out-of-sample pinball loss and CRPS of the considered models on the test set. We differentiate between four cases in terms of estimating the DC that will ship an order (dc-ori) containing a specific SKU, as discussed in Section 4. To reiterate, perfect information assumes the dc-ori is known exactly, and full-history estimation predicts the dc-ori as the one that fulfilled the most orders for the SKU and dc-des in the training set. Recent-history estimation predicts the dc-ori as the most frequent DC for the last 5 or 10 orders that have been processed most recently. We observe that quantile regression, regression tree, and QRF all achieve significantly better forecasting performance than online empirical forecasting. Among them, QRF is the best model as it yields the lowest average pinball loss and CRPS in all scenarios, reporting more than 40% error reductions from online empirical forecasting. Compared to the simple quantile regression, QRF is able to reduce the average pinball loss and CRPS by about 13% and 11%, respectively. Not surprisingly, the perfect-information results enjoy the highest accuracy for all regression-based models. Among the three DC estimation approaches, the recent-history estimation based on the most recent 5 orders outperforms the other two. Overall, the online retailer can improve the distributional forecasting accuracy by exploiting the relationship between predictors and delivery time with a more advanced machine learning method (QRF). The improvement is the largest when the online retailer can acquire the order fulfillment information in real time before an order checkout. Otherwise, utilizing the recent order history can still lead to a promising distributional forecasting model.

Table 3 Avaerge pinball loss and CRPS of different models on the test set

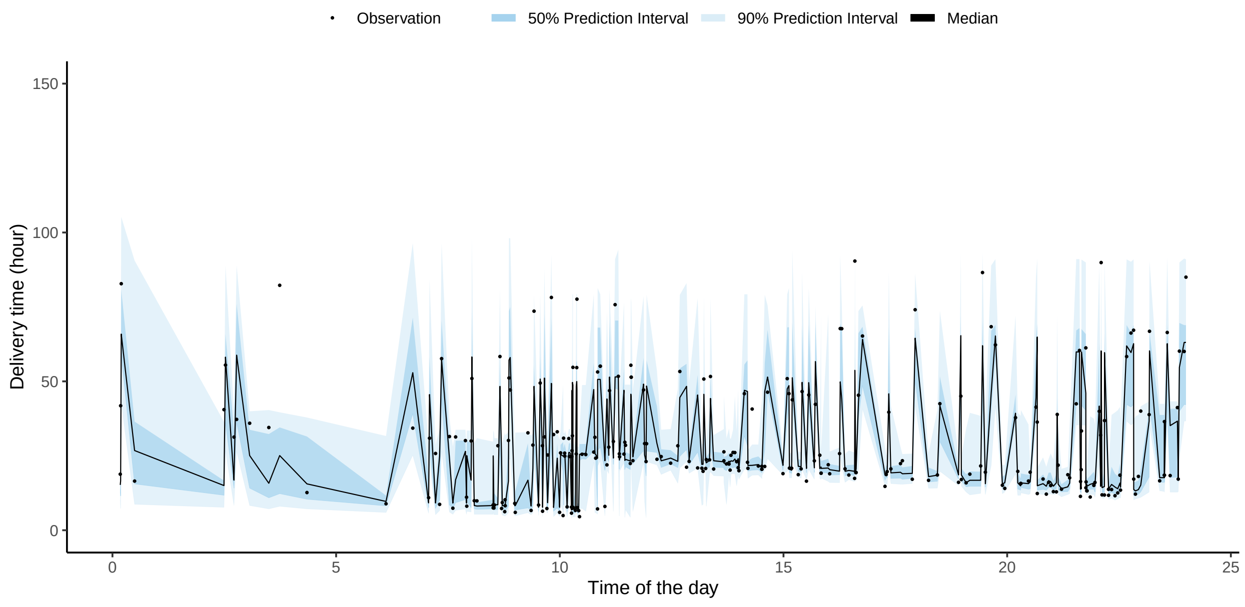
Model		Pinball loss						CRPS
		$Q_{0.05}$	$Q_{0.25}$	$Q_{0.5}$	$Q_{0.75}$	$Q_{0.95}$	Avg.	
Online empirical forecasting		1.585	4.765	7.182	7.707	3.409	4.930	11.027
Perfect-information	Quantile regression	1.236	3.279	4.538	4.331	2.459	3.169	6.905
	Regression tree	0.948	3.477	4.752	4.347	**1.973**	3.100	6.793
	QRF	**0.747**	**2.772**	**4.098**	**4.014**	1.974	**2.721**	**6.079**
Full-history estimation	Quantile regression	1.337	3.427	4.741	4.580	2.683	3.354	7.281
	Regression tree	1.057	3.656	4.983	4.595	2.106	3.279	7.120
	QRF	**0.818**	**2.980**	**4.376**	**4.271**	**2.076**	**2.904**	**6.492**
Recent-history estimation (5)	Quantile regression	1.273	3.345	4.617	4.403	2.506	3.229	7.033
	Regression tree	1.001	3.554	4.835	4.421	1.993	3.161	6.933
	QRF	**0.788**	**2.877**	**4.201**	**4.085**	**1.990**	**2.788**	**6.234**
Recent-history estimation (10)	Quantile regression	1.272	3.348	4.625	4.415	2.509	3.234	7.046
	Regression tree	1.003	3.560	4.846	4.432	**1.992**	3.166	6.948
	QRF	**0.790**	**2.887**	**4.220**	**4.110**	1.995	**2.800**	**6.261**

To visualize the forecasting results, Figure 3 displays the prediction intervals (50% and 90%) and the median point forecasts of QRF for orders arriving on a test day. Due to a large number of observations, we only present the results for two randomly selected destination DCs (dc-dest): DC 16 and DC 33. It is shown that most observations fall within our prediction intervals, and the median forecasts capture the variation of delivery time over different time periods. Furthermore, as we can see from Figure 3a, there is a small jump of delivery time around 11 am in DC 16. This is because many orders placed before 11 am were given same-day delivery services, while orders placed after 11 am would only be given next-day delivery services (i.e., orders placed after 11 am tend to have longer delivery time). Additionally, the delivery speed varies across different DCs: as shown in Figure 3b, the delivery time for DC 33 has a different temporal pattern than DC 16 as the same-day deliveries were less frequent in DC 33. Also, we observe that the delivery time shows relatively large fluctuations even for orders placed within the same hour. The main reason is that orders could be shipped from different DCs (dc-ori) as informed by the fulfillment algorithm. Since DCs are located in different physical areas, the variation in dc-ori would cause significant fluctuations in delivery time, which highlights the importance of capturing fulfillment operations in forecasting. When conditioning on dc-ori, the delivery time tends to have a smoother temporal distribution, as shown in Appendix C.

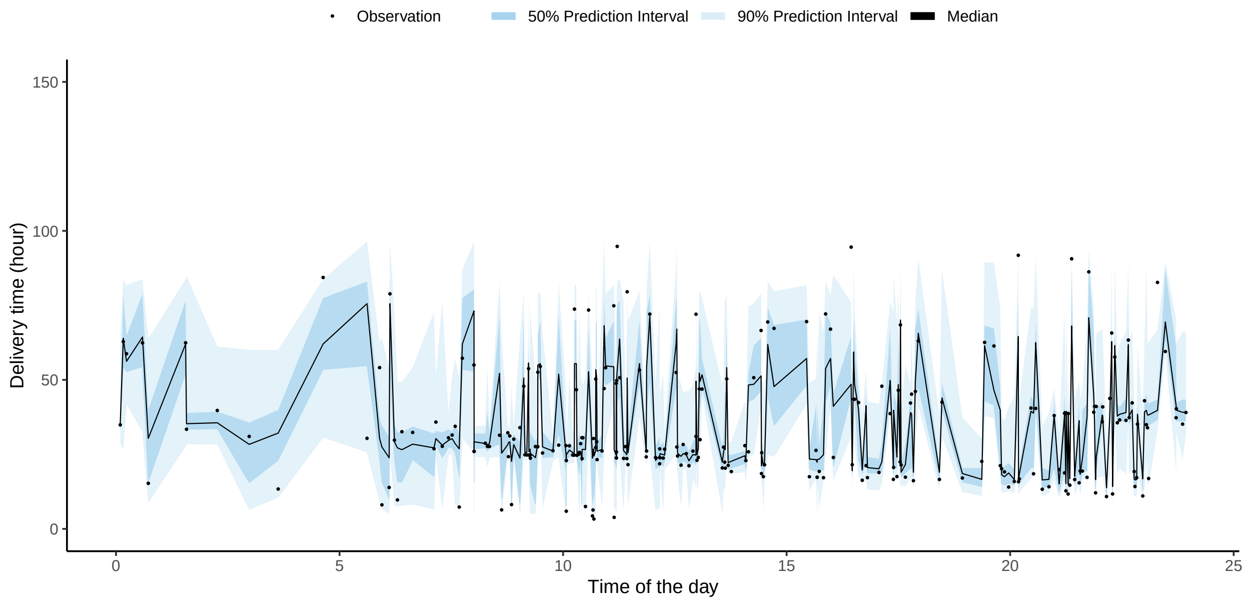
We also compare the distribution of actual observations versus our predictive posterior distribution in Figure 4, following the approach of Shang et al. (2017). The predictive distribution is drawn by sampling from the QRF forecasts on test data points. We find that the predictive distribution has an excellent fit to the actual data distribution, capturing both the asymmetric and multimodal features.

6.1.2. Probabilistic Forecasting Accuracy After deriving the estimated conditional distribution of delivery time, we can discretize it to obtain the estimated probabilities of delivery days ($\{\hat{p}(k|x)\}_{\forall k}$), as introduced in Section 5.3. These probabilistic estimates can then be utilized to set the promised delivery time to customers. Therefore, we also evaluate the performance of these probabilistic forecasts against the realized delivery days.

To measure the probabilistic forecasting accuracy, we adopt two common scoring rules: Logistic loss function (log loss) and Brier score. The log loss, as known as cross-entropy loss, is defined as $-\sum_{k\in\mathcal{K}} y_k(x) log(\hat{p}(k|x))$, where $y_k(x)$ is a binary label that equals to 1 if the sample x belongs



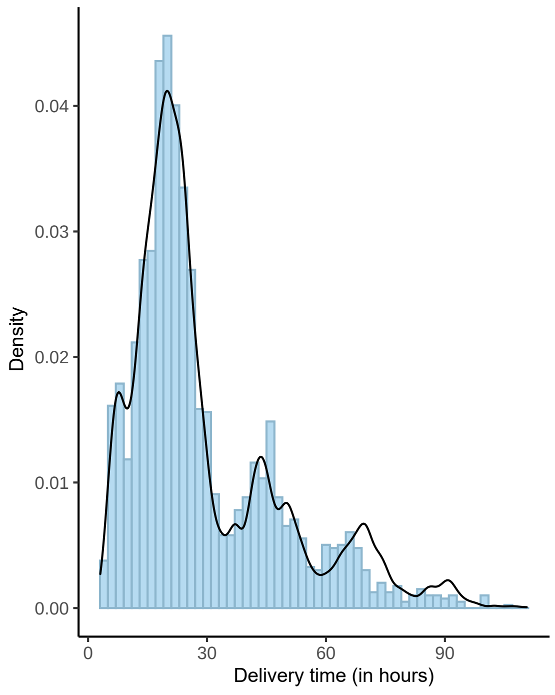
(a) DC 16



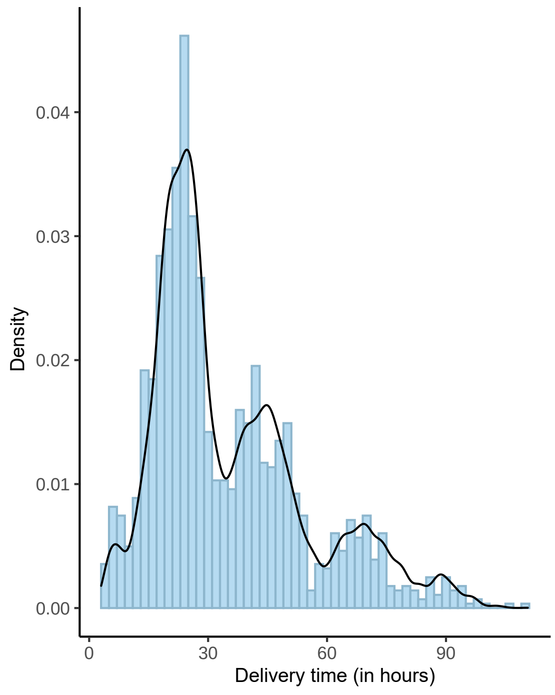
(b) DC 33

Figure 3 Actual observations versus forecasts (median and prediction intervals) on a test day

to class k and 0 otherwise. This function has been used in training classification models such as logistics regression. The second score function, Brier score, is defined as $\sum_{k\in\mathcal{K}}(y_k(x) - \hat{p}(k|x))^2$ (Brier 1950). The best possible Brier score is 0 when we predicts a probability of 1 for the correct class and 0 for the remaining classes.



(a) DC 16



(b) DC 33

Figure 4 Histogram of actual observations (in blue color) versus predictive density from QRF forecasts (black solid line)

Instead of leveraging distributional forecasts for delivery day estimation, we can also directly apply classification models from machine learning in this classification context. Hence we consider two additional machine learning benchmarks: multinomial logistics regression and random forest classification. Multinomial logistics regression extends the linear regression to classification tasks by modeling the log-odds as linear functions of predictors. As a result, it ensures the posterior probabilities sum up to one. Similar to QRF, random forest classification builds a set of trees by a combination of bagging and feature sampling. For classification purposes, trees are grown with splitting rules that minimize classification cost measures (i.e., node impurity measures such as Gini index). We also performed a three-fold cross-validation to tune the random forest classification models. Both multinomial logistics regression and random forest classification are popular in a wide range of applications and more details can be found in Friedman et al. (2001).

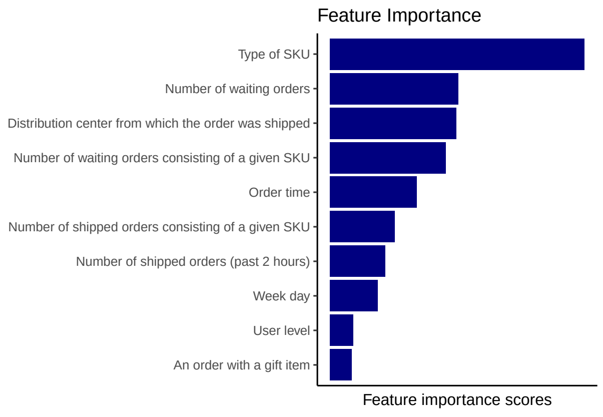
Table 4 presents the average log loss and Brier score of all models along with the classification benchmarks. Similar to the previous discussion, incorporating the constructed predictors leads to significant improvements in forecasting accuracy as all regression methods (and the random forest

Table 4 Average log loss and brier score of different models on the test set

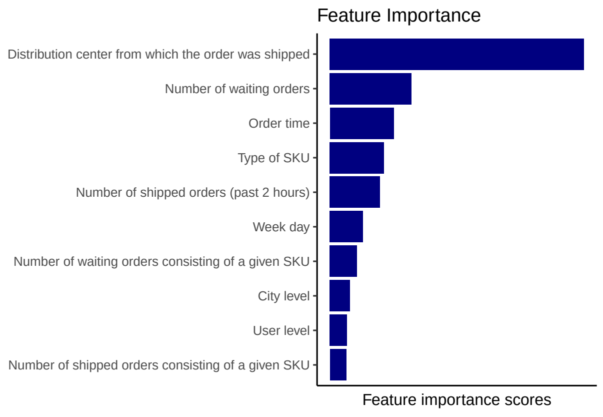
Model		Log loss	Brier score
Online empirical forecasting		1.757	0.530
Perfect-information	Multinomial logistic regression	0.586	0.289
	Random forest classification	0.682	0.285
	Quantile regression	0.926	0.308
	Regression tree	**0.572**	0.294
	QRF	0.576	**0.272**
Full-history estimation	Multinomial logistic regression	0.641	0.308
	Random forest classification	0.721	0.303
	Quantile regression	1.076	0.325
	Regression tree	**0.607**	0.311
	QRF	0.609	**0.290**
Recent-history estimation (5)	Multinomial logistic regression	0.600	0.295
	Random forest classification	0.679	0.291
	Quantile regression	0.954	0.313
	Regression tree	**0.582**	0.299
	QRF	**0.582**	**0.279**
Recent-history estimation (10)	Multinomial logistic regression	0.600	0.296
	Random forest classification	0.680	0.293
	Quantile regression	0.953	0.314
	Regression tree	**0.581**	0.300
	QRF	0.583	**0.281**

classification model) achieve considerably lower errors than online empirical forecasting. Since we use probabilistic scoring functions, quantile regression gives higher errors than multinomial logistic regression and random forest classification (note that multinomial logistic regression is fitted to minimize the log loss function). Notably, the two tree-based distributional forecasting models deliver favorable performance: the regression tree reports the lowest log loss while QRF obtains the best Brier score. Moreover, QRF consistently outperforms the two classification benchmarks in both log loss and Brier score. Observing that the gap in log loss between regression tree and QRF is relatively small, QRF is a more promising approach based on its dominating forecasting performance in Section 6.1.1. Also, the recent-history estimation with last 5 observations outperforms the other two estimation policies in most cases, which is consistent with our previous findings.

6.1.3. Critical Predictors In addition to forecasting performance evaluation, it is also important to understand which predictors are most helpful for delivery time forecasting. QRF is able to score feature importance based on the contribution of a predictor in node splitting. Figure 5 presents the feature importance scores of the 10 most critical predictors from QRF for two DCs. It can be seen that the ranking is different for the two selected DCs. The type of SKU is the most important feature for DC 5, while the distribution center from which the order was shipped (dc-ori) is ranked first for DC 33. Nevertheless, we observe that order time and DC related features are generally considered important by QRF, including the number of waiting/shipped orders (of a given SKU or not). This validates our observations from Section 6.1.1 that order time and dc-ori are the two main contributing factors to the variation of delivery time.



(a) DC 5



(b) DC 33

Figure 5 Feature importance scores from QRF (only the top 10 features are listed)

6.2. Setting Cost-Effective Promised Delivery Time

In this subsection we test the performance of our cost-effective delivery time promising strategy based on the accumulated posterior probability threshold. Because QRF is the best-performing distributional forecasting model, we apply it to generate the posterior CDF of delivery time and the estimated probabilities of delivery days. Based on the existing empirical studies and our discussion in Section 5.3, delivery delays (resulted from promising faster deliveries) can cause customer dissatisfaction and fewer future purchases, while promising slower deliveries can make customers abandon their shopping carts and reduce sales (Rao et al. 2011, Cui et al. 2020). Hence we associate positive classification costs for both late deliveries and early deliveries, and we assume late deliveries are more costly (to prioritize customer satisfaction in a highly competitive market). More specifically, we set $c_{kj} = 3(k - j)$ for $j < k$ (late delivery cost) and $c_{kj} = j - k$ for $j > k$ (early delivery cost). We also test other cost matrix structures (e.g., affine and quadratic) and have similalr observations.

In addition to the proposed decision rule based on minimizing the expected cost (MC), we consider two alternative classification rules: the naive Bayes rule (NB) and maximum-accuracy rule (MA). The NB rule simply selects the class with the highest estimated probability and the MA rule finds the optimal accumulated posterior probability threshold by maximizing the classification accuracy (as opposed to cost minimization). We report three classification metrics: average cost per order, accuracy, and F_1 score on the test set. Accuracy is the fraction of correct classified instances and F_1 score is calculated as the harmonic mean of precision and recall:

$$F_1 \text{ score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}}, \tag{7}$$

where TP, FP, and FN are the number of true positives, false positives, and false negatives, respectively. F_1 score has been widely used in multi-class classification, of which a higher value is more desirable (Lewis et al. 2004, Tsoumakas et al. 2010).

Table 5 compares the classification performance of QRF with the three decision rules, using JD.com's estimation as the benchmark (the promised delivery time recorded in the data set). In addition to the perfect-information scenario, we only present the recent-history estimation approach using the last 5 observations because of its high accuracy. We find that regardless of the decision rule, QRF outperforms JD.com in accuracy and F_1 score. NB leads to the highest F_1 score among the three decision rules. MA gives the highest accuracy, and MC achieves the lowest average cost. While QRF-NB fails to reduce the average cost and QRF-MA only reports small cost improvements, QRF-MC delivers more than 15% cost reductions from JD.com. This highlights the effectiveness of our cost-sensitive decision rule in delivery time promising. As compared to JD.com, QRF-MC obtains consistent improvements in the three metrics across all DCs, as shown in Figure 6. Hence our framework not only improves the reliability of promised delivery time, but also attains a better trade-off between customer satisfaction and sales (as indicated by the minimum cost). Interestingly, employing the most recent 5 observations for estimating DC does not cause a significant loss in all classification metrics when comparing to the perfect-information result.

Table 5 Classification performance on the test set (QRF-NB: QRF using the naive Bayes decision rule, QRF-MA: QRF using the maximum-accuracy decision rule, QRF-MC: QRF using the minimum-cost decision rule)

Model		Average cost/order	Accuracy	F_1 score
Perfect-information	QRF-NB	0.539	0.823	**0.429**
	QRF-MA	0.585	**0.827**	0.413
	QRF-MC	**0.465**	0.789	0.413
Recent-history estimation (5)	QRF-NB	0.546	0.820	**0.426**
	QRF-MA	0.593	**0.824**	0.410
	QRF-MC	**0.471**	0.785	0.411
JD.com		0.556	0.618	0.337

6.3. Potential Implementations and Other Applications

In order to implement the proposed framework, predictor variables should be available in real time, i.e., they can be pulled from real-time data streams of the online retailer (potentially from the cloud server). As presented in Section 4, predictors such as time, order characteristics, and user characteristics are readily available to the retailer at order checkout. The only predictors that may require additional IT work are variables related to DC operations. Based on our construction, we track the order status to compute the real-time workload and processing rate at each DC, for which the information should be updated timely (e.g., whether the order has been assigned to/shipped by a DC). Note that the DC processing rate can also be estimated by the staff level (such as packers), and the online retailer can make such information available in real-time data streams. For DCs that mainly rely on automation technologies (examples include the automated warehouse from JD.com

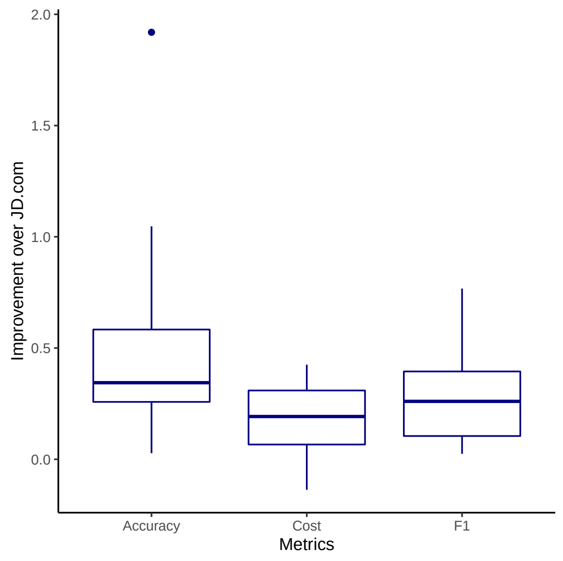


Figure 6 **Improvement of QRF-MC (recent-history estimation) over JD.com across different DCs**

and the Kiva systems adopted by Amazon, see CNBC 2018), the processing rate may be evaluated precisely to improve the online forecasting.

In addition to the promised delivery time management, our forecasting model can be applied to assist operational decisions. For instance, one may use the delivery station arrival time as the target variable so we can forecast the distribution of station arrival time of new orders (a delivery station refers to the last-mile facility that is the last stop of order delivery). As such, the time forecasts can be aggregated properly to predict the future order arrivals in different delivery stations, which can help with the labor planning at these facilities.

Furthermore, although the presented case study is focused on promising delivery days, the proposed framework can be easily adapted to contexts where a more stringent delivery target is expected. In grocery and meal delivery, it is common that service providers give customers estimated delivery time in hours or minutes (e.g., Uber Eats). Our distributional forecasting framework is able to provide such estimates in real time. In fact, some companies such as Grubhub choose to provide an estimated range of delivery time (e.g., 60-70 minutes) instead of a point estimate. The proposed distributional forecasting model presents an immediate solution to such estimation requirements.

7. Concluding Remarks

Delivery time forecasting and promising plays an essential role in managing customer expectations and improving customer satisfaction for online retailers. Albeit its practical significance, the relevant research is rather limited. We are the first to develop a data-driven framework to generate real-time distributional forecasts of delivery time and decide the promised delivery time for new customer orders. Our forecasting model builds on advanced machine learning approaches that can effectively capture the complicated relationship between delivery time and time-varying predictors. Specifically, we adapt the regression tree and QRF to estimate nonparametric conditional distributions. We borrow the idea from service operations literature to construct queue-length predictors by tracking the number of waiting/shipped orders in different DCs, which are shown to be critical to forecasting performance.

By treating the delivery time promising problem as a classification task, we propose a tractable decision rule that transforms the distributional forecasts to promised delivery days. The decision rule considers different misclassification costs associated with late deliveries and early deliveries, and is able to set promised delivery time in a cost-effective way. We test our framework on a comprehensive data set from JD.com. Based on both quantitative and visual evaluations, QRF provides superior forecasting performance. Compared to the promised time provided by JD.com, our framework has the potential to reduce the misclassification cost and enhance the reliability of promised delivery time.

We envision several possible lines of future research. First of all, when the exact fulfillment information is not available, it is possible to build a machine learning model (classification model) to estimate the DC using relevant features such as workload and inventory levels in different DCs. The learned model will thus be able to predict the solution of the fulfillment algorithm from historical data (a similar paradigm can be found in Hildebrandt and Ulmer 2020). Second, it would be important to estimate the misclassification costs related to late deliveries and early deliveries. Existing papers have explored the direct impact of delivery speed and reliability on sales, which can be extended to inform the cost estimation. Due to the limitation of our data (the time span is only one month), we plan to pursue this in the future. Lastly, implementing a new delivery time promising policy can bring disruptions to the system. For instance, it may generate more sales and increase the system workload, which can in turn slow the delivery operations. The proposed models may be re-trained to capture this new dynamic and it is worthwhile to further investigate the interaction between delivery time promising and fulfillment algorithm from a systematic point of view.

References

Acimovic, Jason, Stephen C Graves. 2015. Making better fulfillment decisions on the fly in an online retail environment. *Manufacturing & Service Operations Management* **17**(1) 34–51.

Ang, Erjie, Sara Kwasnick, Mohsen Bayati, Erica L Plambeck, Michael Aratow. 2016. Accurate emergency department wait time prediction. *Manufacturing & Service Operations Management* **18**(1) 141–156.

Arora, Siddharth, James W Taylor, Ho-Yin Mak. 2020. Probabilistic forecasting of patient waiting times in an emergency department. arXiv preprint arXiv:2006.00335.

Bernard, Simon, Clément Chatelain, Sébastien Adam, Robert Sabourin. 2016. The multiclass roc front method for cost-sensitive classification. *Pattern Recognition* **52** 46–60.

Bourke, Chris, Kun Deng, Stephen D Scott, Robert E Schapire, NV Vinodchandran. 2008. On reoptimizing multi-class classifiers. *Machine Learning* **71**(2-3) 219–242.

Bray, Robert. 2019. Operational transparency: Showing when work gets done. *Available at SSRN 3215560* .

Breiman, Leo. 2001. Random forests. *Machine learning* **45**(1) 5–32.

Brier, Glenn W. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review* **78**(1) 1–3.

Chan, Tat, Zekun Liu, Weiqing Zhang. 2018. Delivery service, customer satisfaction and repurchase: Evidence from an online retail platform. Available at SSRN 3258106.

CNBC. 2018. The world's first humanless warehouse is run only by robots and is a model for the future. https://www.cnbc.com/2018/10/30/the-worlds-first-humanless-warehouse-is-run-only-by-robots.html. Accessed on 2020-08-15.

Cui, Ruomeng, Meng Li, Qiang Li. 2019. Value of high-quality logistics: Evidence from a clash between sf express and alibaba. *Management Science* .

Cui, Ruomeng, Tianshu Sun, Zhikun Lu, Joseph Golden. 2020. Sooner or later? promising delivery speed in online retail. Availabe at SSRN 3563404.

Dong, Jing, Elad Yom-Tov, Galit B Yom-Tov. 2019. The impact of delay announcements on hospital network coordination and waiting times. *Management Science* **65**(5) 1969–1994.

Fisher, Marshall L, Santiago Gallino, Joseph Jiaqi Xu. 2019. The value of rapid delivery in omnichannel retailing. *Journal of Marketing Research* **56**(5) 732–748.

Friedman, Jerome, Trevor Hastie, Robert Tibshirani. 2001. *The elements of statistical learning*, vol. 1. Springer series in statistics New York.

Gneiting, Tilmann, Adrian E Raftery. 2007. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association* **102**(477) 359–378.

Grushka-Cockayne, Yael, Kenneth C Lichtendahl Jr, Victor Richmond R Jose, Robert L Winkler. 2017. Quantile evaluation, sensitivity to bracketing, and sharing business payoffs. *Operations Research* **65**(3) 712–728.

Guo, Xiaojia, Yael Grushka-Cockayne, Bert De Reyck. 2019. Forecasting airport transfer passenger flow using real-time data and machine learning. Available at SSRN 3245609.

Ha, Thien M. 1997. The optimum class-selective rejection rule. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(6) 608–615.

Hernández-Orallo, José, Peter Flach, Cèsar Ferri. 2012. A unified view of performance metrics: translating threshold choice into expected classification loss. *The Journal of Machine Learning Research* **13**(1) 2813–2869.

Hildebrandt, Florentin D, Marlin W Ulmer. 2020. Supervised learning for arrival time estimations in restaurant meal delivery. Working paper.

Hsu, Abby PT, Charles V Trappey, Amy JC Trappey, et al. 2015. Using ontology-based patent informatics to describe the intellectual property portfolio of an e-commerce order fulfillment process. *ISPE CE*, vol. 2015. 62–70.

Ibrahim, Rouba, Ward Whitt. 2009a. Real-time delay estimation based on delay history. *Manufacturing & Service Operations Management* **11**(3) 397–415.

Ibrahim, Rouba, Ward Whitt. 2009b. Real-time delay estimation in overloaded multiserver queues with abandonments. *Management Science* **55**(10) 1729–1742.

Ibrahim, Rouba, Ward Whitt. 2011. Wait-time predictors for customer service systems with time-varying demand and capacity. *Operations research* **59**(5) 1106–1118.

Koenker, Roger, Gilbert Bassett Jr. 1978. Regression quantiles. *Econometrica: journal of the Econometric Society* 33–50.

Kuhn, Max, Kjell Johnson, et al. 2013. *Applied predictive modeling*, vol. 26. Springer.

Lachiche, Nicolas, Peter A Flach. 2003. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 416–423.

Lauren Freedman. 2019. Why logistics have never been more important to online retailers' success. https://www.digitalcommerce360.com/2019/09/30/why-logistics-have-never-been-more-important-to-online-retailers-success/. Accessed on 2020-08-15.

Lewis, David D, Yiming Yang, Tony G Rose, Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research* **5**(Apr) 361–397.

Matheson, James E, Robert L Winkler. 1976. Scoring rules for continuous probability distributions. *Management science* **22**(10) 1087–1096.

Meinshausen, Nicolai. 2006. Quantile regression forests. *Journal of Machine Learning Research* **7**(Jun) 983–999.

O'brien, Deirdre B, Robert M Gray. 2005. Improving classification performance by exploring the role of cost matrices in partitioning the estimated class probability space. *Proceedings of the ICML Workshop on ROC Analysis in Machine Learning*. Citeseer, 79–86.

Rao, Shashank, Stanley E Griffis, Thomas J Goldsby. 2011. Failure to deliver? linking online order fulfillment glitches with future purchase behavior. *Journal of Operations Management* **29**(7-8) 692–703.

Shang, Yan, David Dunson, Jing-Sheng Song. 2017. Exploiting big data in logistics risk assessment via bayesian nonparametrics. *Operations Research* **65**(6) 1574–1588.

Shen, Zuo-Jun Max, Christopher S Tang, Di Wu, Rong Yuan, Wei Zhou. 2019. Jd. com: Transaction level data for the 2020 msom data driven research challenge. *Available at SSRN 3511861* .

State Post Bureau of the People's Republic of China. 2020. Notice on the 2019 delivery service satisfaction survey results. `http://www.spb.gov.cn/zy/xxgg/202003/t20200312_2056370.html`. Accessed on 2020-08-15.

Sun, Yan, Kiok Liang Teow, Bee Hoon Heng, Chee Kheong Ooi, Seow Yian Tay. 2012. Real-time prediction of waiting time in the emergency department, using quantile regression. *Annals of emergency medicine* **60**(3) 299–308.

Tsoumakas, Grigorios, Ioannis Katakis, Ioannis Vlahavas. 2010. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering* **23**(7) 1079–1089.

Appendix A: Description of Predictor Variables

Table 6 A detailed description of predictor variables used in the study

Features	Description
Time effect:	
1) Order hour	The hour of the day when the order is placed.
2) Week day	The day of the week when the order is placed.
DC operations:	
1) Distribution center from which the order was shipped	Warehouse where the package is shipped from. It can be different from the warehouse that is nearest to the customer's designated shipping address.
2) Number of shipped orders	Number of orders that have been shipped from a distribution center within the last 2 hours. This feature represents the processing rate of the distribution center within the last 2 hours.
3) Number of waiting orders	Number of orders in the distribution center waiting to be shipped. This feature represents the waiting line for the distribution center which is updated for every order.
4) Number of waiting orders consisting of a given SKU	The waiting line for a specific SKU and a distribution center is approximated by the number of orders (for a given SKU in a distribution center) waiting to be shipped. For a multi-item order, this feature is obtained for every SKU presented in the order, and the SKU with the maximum number of waiting orders is selected as the main observation.
5) Number of shipped orders consisting of a given SKU	The processing rate for a specific SKU in a distribution center is approximated by the number of shipped SKUs within the last 2 hours. For a multi-item order, this feature corresponds to the main observation.
Order effect:	
1) Number of different SKUs in one order	Total number of different SKUs in one order. This feature is greater than 1 for multi-item orders (except gift item) and equal to 1 for single item orders.
2) Bundle discount	This variable defines if a customer receives bundle discount by buying a prespecified bundle of SKUs within an order. Bundle discount shows the amount of discount received by the customer, however we consider it as a binary variable in our study (it is equal to 1 if the customer receives the bundle discount and 0 otherwise).
3) Coupon discount	This variable represents the customer coupon, which applies if the customer claims the coupon before making the purchase (it is a a binary variable in our study).
4) Type of SKU	Type of SKU representing 1P or 3P SKUs.
5) An order with a gift item	The seller may offer an SKU as a free gift when the customer purchases a prespecified set of SKUs with the final unit price of the gift item equal to 0. If an order receives a gift for any of the SKUs in the order, the value for this feature is 1 (and 0 otherwise).
6) Total quantity	Total quantity of items in one order.
User effect:	
1) User level	Customers are classified according to their past purchases, and user level defines the total purchase value of the customer in the past. User level takes on a value of $\{-1, 0, 1, 2, 3, 4, 10\}$, where value -1 defines a new customer (first-time purchaser), value 10 defines enterprise users (e.g., small shops in rural areas or small businesses), for values in $\{0, 1, 2, 3, 4\}$ a higher user level is associated with a higher total purchase value in the past.
2) City level	Actual information about the most commonly used shipping address for each repeated customer. City level takes on values in $\{-1, 1, 2, 3, 4, 5\}$, where level 1 corresponds to highly industrialized cities (e.g. Beijing), level 2 cities correspond to provincial capitals, levels 3-5 correspond to smaller cities, and -1 corresponds to unknown city level.
3) Plus member	This is a binary variable which defines a user with a PLUS membership on Feb 28, 2018 (it is equal to 1 when the corresponding user is an existing PLUS member). The processing time and delivery duration of orders for PLUS members can be different from regular members.

There are both single-item (1 SKU) and multi-item (several SKUs) orders in the data set. Every observation in the dataset corresponds to one unique item included in the order. This implies that for a multi-item order, it can be mapped multiple observations in the data set (depending on the number of SKUs). However, only one observation per order is required for training. For some of the above features (such as type of SKU), the feature value can vary across SKUs in a multi-item order. To calculate an aggregated value of those features (for multi-item orders), we chose one SKU observation as the main observation and excluded the remaining observations for training purposes: the SKU with the maximum number of waiting orders was selected to be the main SKU observation for feature value aggregation.

Appendix B: Proof of Proposition 1

Proof First, consider the so-called local risk decision rule:

$$\arg\min_j \sum_{k\in\mathcal{K}} c_{kj}\hat{p}(k|x), \tag{8}$$

where the predicted class is chosen as to minimize the expected local misclassification cost function $\hat{C}(j,x)$ (at sample x). It is known that if the probability estimates $\{\hat{p}(k|x)\}_{\forall k}$ are accurate, then this rule is also global optimal in minimizing the expected misclassification cost (O'brien and Gray 2005). Now assuming the misclassification cost has the stated linear structure, we have

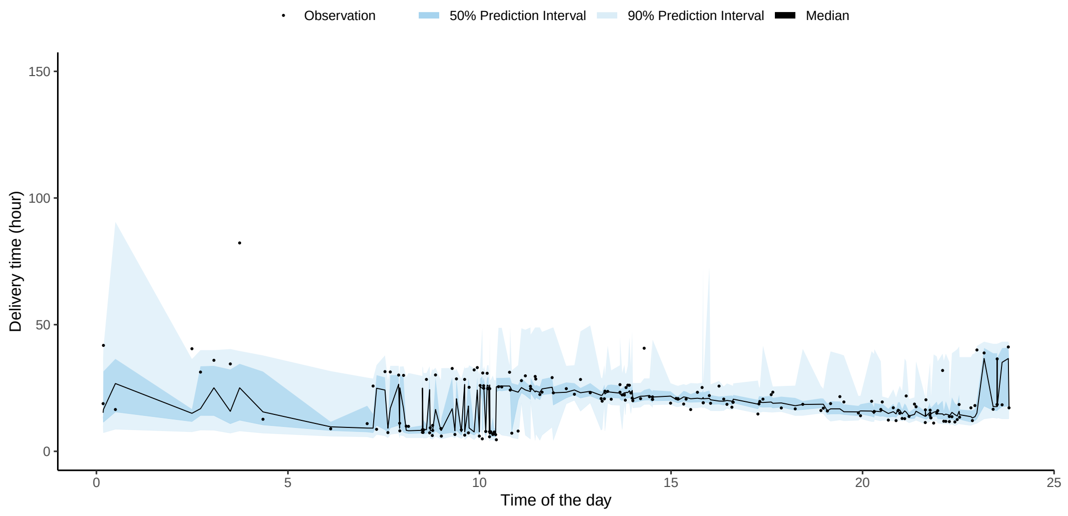
$$\begin{aligned}\hat{C}(j+1,x)-\hat{C}(j,x) =& \sum_{k\le j}(j+1-k)c_1\hat{p}(k|x) + \sum_{k\ge j+1}(k-j-1)c_2\hat{p}(k|x)\\ &-\sum_{k\le j-1}(j-k)c_1\hat{p}(k|x) - \sum_{k\ge j}(k-j)c_2\hat{p}(k|x)\\ =& c_1\sum_{k\le j}\hat{p}(k|x) - c_2\sum_{k\ge j+1}\hat{p}(k|x)\\ =& c_1\sum_{k\le j}\hat{p}(k|x) - c_2(1-\sum_{k\le j}\hat{p}(k|x))\\ =& (c_1+c_2)\sum_{k\le j}\hat{p}(k|x) - c_2.\end{aligned}$$

So we know that $\hat{C}(j+1,x) < \hat{C}(j,x)$ if and only if $(c_1+c_2)\sum_{k\le j}\hat{p}(k|x) - c_2 < 0$, i.e., $\sum_{k\le j}\hat{p}(k|x) < \frac{c_2}{c_1+c_2}$. This implies that we can increase j to reduce the cost when $\sum_{k\le j}\hat{p}(k|x) < \frac{c_2}{c_1+c_2}$. The optimal j^* will be the smallest integer such that $\sum_{k\le j}\hat{p}(k|x) \ge \frac{c_2}{c_1+c_2}$, which is the decision rule $\inf\{j : P_x(j) \ge \tau\}$ with $\tau = \frac{c_2}{c_1+c_2}$. This is actually the optimal newsvendor solution for a discrete demand.

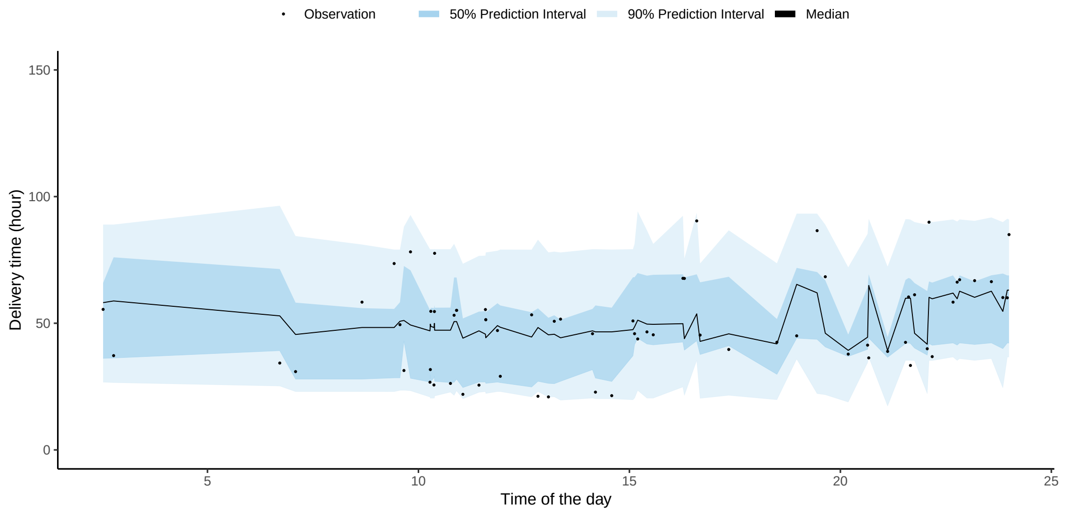
Hence, the local risk decision rule is the same as the accumulated posterior probability rule when the misclassification cost is linear (this may only be valid when the classes are ordered). When the probability estimation is accurate, both rules are optimal. $\square$

Appendix C: Additional Prediction Interval Figures

We present here the prediction interval figures for DC 16 and DC 33 by conditioning on the DC from which the order was shipped (dc-ori), in Figure 7 and 8, respectively. We find that after conditioning on dc-ori, the delivery time has less fluctuations. Still, our forecasts exhibit similar patterns to actual observations and provide good-quality prediction intervals.

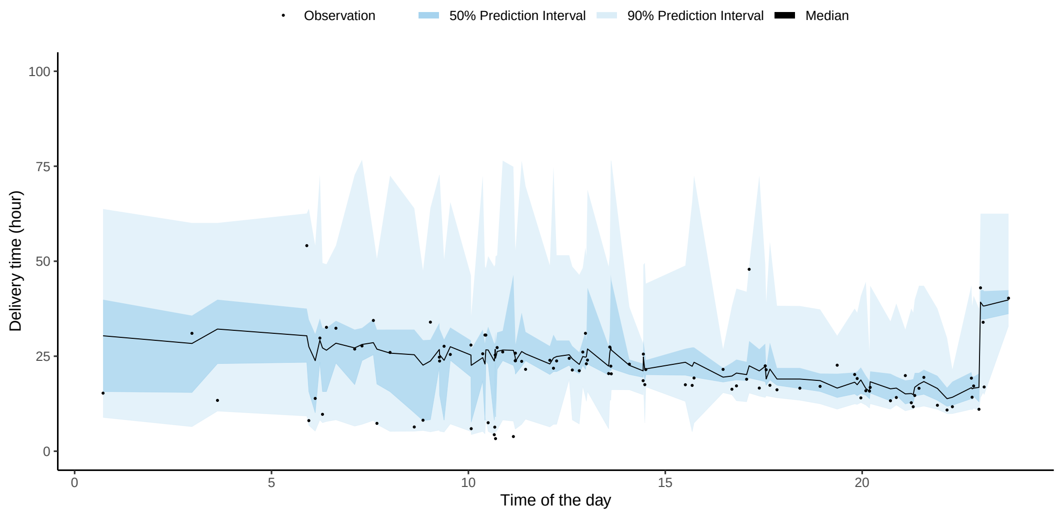


(a) dc-ori = 3

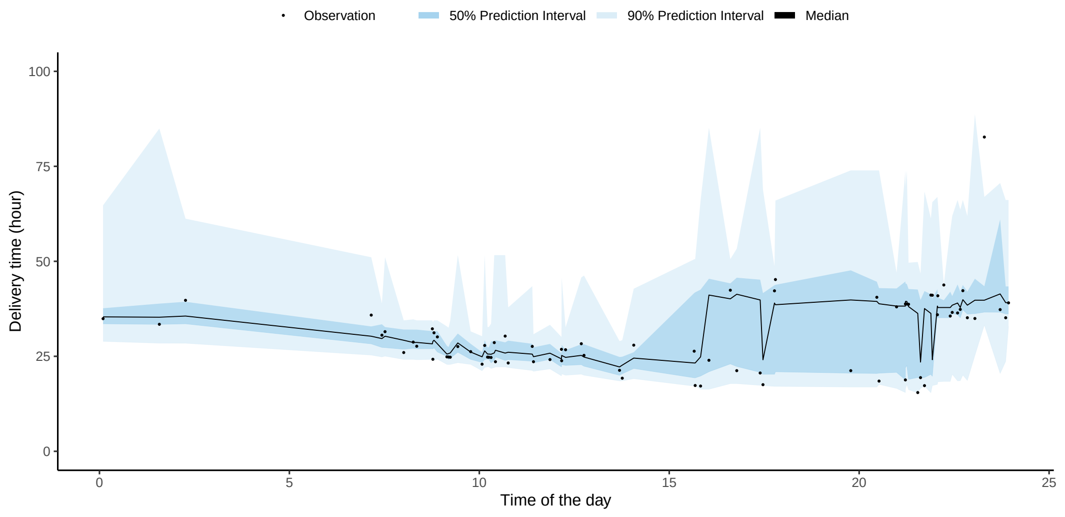


(b) dc-ori = 5

Figure 7 **Actual observations versus forecasts (median and prediction intervals) on a test day for DC 16 conditioning on dc-ori**



(a) dc-ori = 33



(b) dc-ori = 3

Figure 8 **Actual observations versus forecasts (median and prediction intervals) on a test day for DC 33 conditioning on dc-ori**