Forecasting third-party mobile payments with implications for customer flow prediction

Shaohui Ma[a1],
Robert Fildes[b]
[a] School of Business, Nanjing Audit University, Nanjing, 211815, China
[b] Lancaster Centre for Marketing Analytics & Forecasting, Lancaster University, Lancaster, LA1 4YX, UK

Abstract

Forecasting customer flow is the key for retailers when making their daily operational decisions, but small retailers are often lack resources to obtain such forecasts. Instead of forecasting stores' total customer flows, this research utilizes the newly emerging third-party mobile payment data to provide participating stores with a value-added service by forecasting their share of daily customer flows. These customer transactions using mobile payments can then be further utilized to derive retailers' total customer flows indirectly, thereby overcoming the constraints small retailers face. We propose a third party mobile-payment platform centered daily mobile payments forecasting solution based on an extension of the newly developed Gradient Boosting Regression Tree (GBRT) method which can generate multi-step forecasting for many stores concurrently. Using empirical forecasting experiments with thousands of time series, we show that GBRT together with a strategy for multi-period ahead forecasting provides more accurate forecasts compared with established benchmarks. Pooling data from the platform across stores leads to benefits compared to analyzing the data individually, demonstrating the value of this machine learning application.

Keywords: Analytics; big data; customer flow forecasting; machine learning; forecasting many time series; multi-step ahead forecasting strategy

[1] E-mail address: shaohui.ma@nau.edu.cn (Shaohui Ma); r.fildes@lancaster.ac.uk (Robert Fildes).

1. Introduction

The increased availability of sources of 'big data' capturing consumer behaviour offers new opportunities for companies to understand, forecast and plan. Forecasting customer flow is one of the key factors to a successful retail business. With the aid of accurate customer flow forecasting, merchants can optimize their human resource and inventory planning, reduce operational cost and improve customer experience (Chapados et al., 2014; Mou et al., 2017). Large retail chains can employ professional forecasters, investing heavily in sophisticated customer flow monitoring and forecasting systems, and therefore have the ability to make accurate customer flow forecasting themselves. Small retailers, however, usually lack funds and expertise to obtain such forecasts and can only run their business by rule of thumb. Today, with the increased prevalence of third-party mobile payment services in China, it is now possible to help millions of small businesses improve their operations by providing professional customer flow forecasts based on third-party payment data with the potential added benefit arising from a shared data source.

As the mobile wallet ecosystem is maturing, in-store mobile payments have taken off and are enjoying rapid growth in China. The Gross Market Value (GMV) of China's mobile payment market reached 27 trillion Yuan (or 3.47 trillion Euro) in Q2 2017, soaring 95.4% from a year earlier and up 19.5% compared with the previous quarter[2], and it is expected to continue to grow quickly in the next few years. This research is based on the data provided by one of leading electronic payment platform, which has the largest mobile payment market share in China, and has attracted millions of retailers to using the platform, accumulating a huge amount of user payment data.

The third-party mobile payment platform collects data from both stores and customers (as shown in Fig.1). Stores who want to accept the mobile payment need to register their necessary information on the platform, e.g., location, product/service provided, and their licenses etc., and open an account on the platform which can connect with their bank accounts. Similarly, customers who want to use mobile payment also need to provide necessary personal information to open an account on the platform and download and install an application developed by the platform on their phone. Customers can use the application when seeking stores nearby, browsing store ratings and comments contributed by the other customers, as well as other store information. After choosing the target store, they visit the store for browsing and

[2] Data source is from iResearch, http://www.iresearchchina.com/content/details7_37999.html.

purchase, then making payments through a two-dimensional code generated by the application on their phone; stores then scan the code to confirm the transaction. After the purchasing experience, customers may share their own shopping experiences on the platform. The payments will first be transferred from the user's bank account to the user's platform account, and then to the merchant's account on the platform. The platform acts in the role of third party in the transaction, but effectively collects all the mobile transactions data from its users.

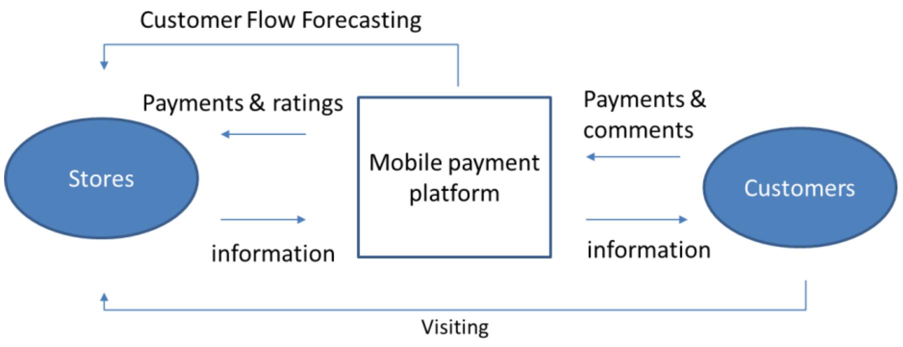
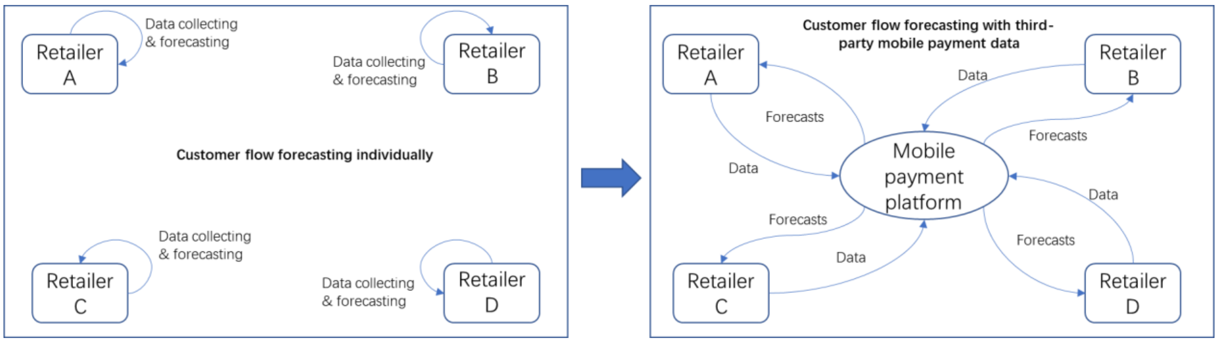


Figure 1. Mobile payment data collection process and customer flow forecasting

Using these store mobile payment data, this research aims to forecast the daily mobile payment customer flows for all those retailers sharing the platform. Though the customer flow forecasting provided by the platform here is limited to in-store customers using mobile payments, it is still expected to be an important reference point for small retailers to optimize their operations. Mobile transactions are forming an increasing proportion of payments and for stores where these transactions contribute a major share of the store totals they will typically be able to estimate total transactions accurately. In the past, forecasting customer flows was the retailer's own challenge, either using their historical data or, more likely, by simple rules of thumb. With third-party mobile payment data available from stores the problem can be transformed where professional mobile payment customer flow predictions are centrally generated to registered retailers, and then the total customer flow predictions are derived automatically given each retailers' own estimated share of their mobile payment customers, a solution which especially benefits the large number of small retailers (Fig. 2).



Customer flow forecasting individually
Retailer A
Retailer B
Retailer C
Retailer D
Data collecting & forecasting
Customer flow forecasting with third-party mobile payment data
Mobile payment platform
Data
Forecasts

Figure 2. Customer flow forecasting individually vs. customer flow forecasting with third-party mobile payment data

For the individual retailer, a number of alternative methods are available from simple rules of thumb (e.g. tomorrows sales are forecast to be the same as on the same day last week) to more complex univariate time series such as ARIMA. However, univariate methods do not capture the common effects from related series which share common drivers such as seasonality, common trends, weather effects and macroeconomic indicators (Arunraj and Ahrens, 2015;van Donselaar et al., 2016), and other complex characteristics, such as cross-sectional temporal and spatial interactions. Duncan et al. (2001) discuss the many situations where such pooling methods are likely to be beneficial. In addition to there being common features affecting retailers in the same town or development, two more recommendations apply directly: use pooling where there is high volatility or outliers.

In this research, we propose a novel pooled approach to capitalize on the commonalities in the data across the participating retailers: a Gradient Boosting Regression Tree (GBRT) based machine learning solution for customer flow forecasting with third-party mobile payment data. While machine learning (ML) methods have gained increasing currency, they have not been widely tested: where they have been tested their success has been moot. In the particular retail context we examine, the machine learning GBRT potentially provides evidence of its ability to outperform conventional univariate forecasting methods. Our solution provides multi-day ahead forecasts for a large number of stores concurrently. Its flexibility can include common drivers as well as coping with series of different lengths. We identify a set of important explanatory variables in forecasting customer flows through an efficient heuristic features selecting algorithm from a large candidate set, and propose a new strategy to generate multi-step ahead forecasts. To our knowledge, this is the first paper to introduce a scalable customer flows forecasting solution based on third party mobile payment data. We show this GBRT solution, when tailored to the problem context, provides improved accuracy over conventional univariate time series methods, as well as pooled regression and Random Forest based solutions, thereby adding a substantial piece of positive empirical evidence to the body of evidence on the forecasting accuracy of machine learning methods. It should therefore no longer be possible to claim that ML methods should not form a part of the armory of time series forecasting methods worth considering in any particular problem context.

The outline of the paper is as follows. In Section two, we review related studies and introduce our innovations. In Section three we discuss associated methodological issues in our proposed solution. In Section four, we describe the data, introduce the experiments design and forecasting accuracy measures. We then present the empirical results in Section five. In the last section, we discuss the findings offering conclusions as to forecasting practice and further academic research.

2. Related research

This research builds on four topics in the forecasting literature: customer flow forecasting, forecasting for many time series and the contribution made by ML methods, forecasting with a Gradient Boosting Tree, and multi-step ahead forecasting strategies.

2.1. Customer flow forecasting

Customer flow is the number of customers visiting a store during a period of time. It is also named as store traffic or foot traffic in the existing literature. So far the subject of customer flow forecasting has received very limited research focus; this is mainly due to the difficulties in obtaining customer flow data (Abrishami et al., 2017). To collect large scale and reliable customer flow data, a comprehensive system with sensors is often required. In existing researches, security gates (Veeling, 2014), digital cameras linked with human facial recognition system (Cortez et al., 2016), and wireless access points (Abrishami et al., 2017) have been used to count people entering or exiting. While these systems can record customer flow accurately, they need extra investments and a bounded space, which limits the application for small retailers. Traditionally, the number of transactions collected by POS machines is often adopted as a proxy for store traffic (Walters and Mackenzie, 1988; Walters and Rinne, 1986; Williams et al., 2014), as POS machines have been widely adopted by retailers. Nowadays, the third-party mobile payment data provides a new way to track customer flows, which is costless and has higher availability compared with traditional approaches. Moreover, the data is collected from an online platform instead of from individual retailers; this makes it possible to provide millions of small retailers on the platform with professional customer flow forecasts without extra cost. The issue we resolve is whether stores could obtain more accurate customer flow forecasting from using the shared platform than from relying only on their own data and which methods are most effective.

2.2. Forecasting methods for many time series

Platform centered customer flow forecasting with third-party payment data is a many time series forecasting problem, a common problem for many organizational forecasters (Fildes and Petropoulos, 2015). The modeling strategies for many time series can be classified into two categories: modeling each time series individually or modeling the group of time series in a pooled fashion. Various forecasting approaches have been applied to this problem: linear regression (Chuang et al., 2016; Lam et al., 1998), univariate time series models (Cortez et al., 2016), and a number of ML methods such as Random Forest Regression (Abrishami et al., 2017), Support Vector Regression (Abrishami et al., 2017; Cortez et al., 2016) and Neural networks (Veeling, 2014).

Modeling at individual level can fully consider each time series' characteristics, such as seasonality and trend and in recent years many increasingly complex individual methods have been proposed under the heading machine learning, aiming to capture more complex patterns (Aye et al., 2015; Au et al., 2008; Wong & Guo, 2010). But such innovations have led some researcher to argue from their reading of the research literature that simple models often beat complex models in time series forecasting (Green & Armstrong, 2015). The famous M3 competition, now followed by M4, have reached the more specific conclusion that complex machine learning models cannot improve forecasting accuracy on their own (Makridakis & Hibon, 2000; Crone, Hibon, & Nikolopoulos, 2011; Makridakis et al., 2018a, 2018b, 2019). In Makridakis et al. (2018a) for example the authors considered 8 ML methods including neural nets and regression trees and argued that the results showed for the a sub-set of the M3 data, the ML methods all performed worse than the benchmark statistical methods. Our own reading of Makridakis et al. (2019) suggests that ML approaches can certainly do well, even when applied over a wide range of heterogeneous data series. Other studies, typically on more homogeneous data than those in the M-Competitions, have also been more positive with Gur Ali et al. (2009) finding that regression trees outperformed an exponential smoothing benchmark (for 168 retail series) but only in periods where the SKUs were promoted. One important reason for the mixed results is that data modelling at individual level usually faces the problem of scarce observations, where complex models are easy to overfit, leading to relatively poor forecasts, while simple models are not affected by this problem. Modeling time series individually through simple methods often works well. But such simple approaches cannot capture data features common in practice such as common seasonal patterns (Chen and Boylan, 2008; Zotteri et al., 2005), promotional effects (Gur Ali et al., 2009), nonlinear trends and other complex characteristics, such as cross-sectional temporal and spatial interactions: this is especially true for short time series with high noise (Duncan et al., 2001).

The alternative to modeling the time series individually is through pooling the many time series together: this clearly enhances the data available for modelling and has the potential to capture cross time series common patterns, thereby improving the robustness of the estimated parameters (Dekker et al., 2004; Zotteri and Kalchschmidt, 2007). When the number of time series are large, simple linear models would suffer the problem of underfitting in this situation, so pooled data can potentially benefit from more complex methods, both in terms of their inclusion of analogous time series and also the non-linear patterns included in machine learning approaches. It is therefore an open question whether pooled complex methods offer improvements in accuracy.

There is some evidence that machine learning methods, such as Neural Networks (NN) and Support Vector Machines (SVM), can generate more accurate forecasts with pooled data than individual forecasting methods (Gür Ali et al., 2009). But traditional NN or SVM models cannot yet fit big data efficiently, and have therefore only been applied on a small scale (Gür Ali and Yaman, 2013; Bandara et al., 2019). With the rapid development of machine learning algorithms, such as random forest, gradient boosting trees and deep learning neural networks, recent years have seen their application to many large-scale time series forecasting problems in areas, such as transportation (Yao et al., 2018), environment (Li et al., 2016), product demand forecasting and electricity prices (Hartmann et al., 2015; Lago et al., 2018). The results, have shown the superior forecasting power of complex models with pooled data.

Most of the existing research has aimed at forecasting the customer flow for a single store. An exception is Abrishami et al. (2017), who test flow forecasting models on over 100 stores, but they model customer flows for each store individually. Compared with the existing literature on forecasting multiple time series literature, this research concerns the customer flow data from stores in various categories and locations, and it focuses on multiple step ahead daily forecasting of data which contain more complex seasonal patterns than the yearly or monthly forecasting in the M4 competition. To consider both store specific characteristics and possible cross time series patterns, we have extracted a unique set of features and developed a heuristic algorithm for feature selection. Our research also provides further evidences that complex models working with pooled data observed in a common environment can deliver improved forecasts above individual forecasting approaches.

2.3. Forecasting with Gradient Boosting Regression Tree

Boosting theory was developed by Valiant (1984). The main idea behind this algorithm is that many weak classifiers though better than random guessing, could create a strong classifier when used in combination. Gradient boosting was first introduced by Friedman (2001) and was motivated as being a gradient descent method in function space, capable of fitting generic nonparametric predictive models. The Gradient Boosting Regression Tree (GBRT) is one of most popular gradient boosting algorithms, which uses a regression tree as the base weak learner.

GBRT has empirically proven itself to be highly effective for a vast array of classification, ranking and regression problems. It is one of the most preferred choices in data analytics competitions such as Kaggle and KDD Cup. Recent studies have applied the GBRT in the domain of insurance loss (Guelman, 2012; Xia et al., 2017), bankruptcy (Zięba et al., 2016), travel time (Zhang and Haghani, 2015 who consider only a few series), and protein solvent accessibility (Fan et al., 2016). But GBRT has not so far been applied in forecasting for many time series. This paper is the first research that systematically introduces a GBRT based solution to this problem. The solution is an integration of data processing procedures, including data cleaning, transformation, feature extraction and selection, hyper-parameter tuning, model training, and forecasts generation. There are various innovations in our solution tailored for customer flow forecasting, including the time series transformation methods for different forecasting strategies, a set of features specially designed for GBRT that summarize time series' local dynamics and global static characteristics, an efficient heuristic features selecting algorithm, and new strategies to generate multi-step ahead time series forecasts as we now discuss.

2.4. Multi-step ahead time series forecasting

An additional methodological issue is the requirement for multiple step ahead forecasting: this is more difficult than one-step, since it usually results in the accumulation of errors and increased uncertainty (Sorjamaa et al., 2007). So far three basic strategies have been proposed in the literature to tackle the multi-step ahead forecasting task, including the Recursive (or Iterative) strategy, the Direct (or Independent) strategy and the Multi-Input Multi-Output (MIMO) (or Joint) strategy. The Recursive strategy is the most intuitive strategy which iterates H times a one-step ahead forecasting model to obtain H-step ahead forecasts. After estimating the next (future) value in the series, the result is fed back as an input into the subsequent forecast. The Direct strategy estimates H forecasting models, each returning a forecast for the i-th (i=1...H) horizon but always relying on the real observed data as inputs. The MIMO strategy, originally proposed by Kline (2004), replaces the H models of the Direct approach with one multi-output model: this aims to preserve the stochastic dependency, between the predicted values, that characterizes the time series. Some variants have also been proposed by combining the basic strategies. For example, Sorjamaa and Lendasse (2006) proposed a combination of the Direct and Recursive strategies called DirRec, in which a different model is used at each step but the approximations from previous steps are introduced into the input set. Similarly, Ben Taieb et al. (2010) proposed another combination strategy, named DIRMO, which aims to preserve the most appealing aspects of both the Direct and MIMO strategies.

The empirical researches on Recursive and Direct strategies have shown contradictory results as to which strategy has better forecasting performance. For example, researches from Birattari et al. (1999) and Weigend et al. (1992) provide experimental evidences in favor of the Recursive strategy against the Direct strategy. In contrast, Sorjamaa and Lendasse (2006) and Hamza et al. (2009) find that the Direct strategy is better than the Recursive strategy. Concerning MIMO, several recent studies (Bontempi, 2008; Bontempi and Taieb, 2011; Taieb et al., 2012; Xiong et al., 2014) have consistently shown that MIMO outperforms both the Recursive and Direct strategies on the data used in artificial Neural Network and computational intelligence forecasting competitions, NN3 and NN5 .

One restriction of the MIMO strategy is it is only applicable with models which could generate multiple outputs. For example, the implementation modeling technique using MIMO in most existing studies is lazy learning, a K-Nearest Neighbor (KNN) based local modeling technique with flexibility when constructing the required multiple input-output modeling structure. But most of the existing computational intelligence methods including gradient boosting trees, cannot use the MIMO strategy straightforwardly for multi-step-ahead prediction due to their inherent single-output structure. In this research, we design a pseudo MIMO approach, which uses single output models to generate multi-step forecasts concurrently. In addition, this paper is the first providing an experimental comparison of the different multi-step ahead forecasting strategies with GBRT when forecasting many time series.

2.5 Summary

This research is innovative in four respects:

(1) It is a novel application in retailing to the practical problem of customer flow forecasting using newly emerging mobile payment 'big' data.

(2) It identifies a set of important predictors for forecasting daily mobile payment customer flows, and explores the benefit of complex models using data pooling for forecasting many time series.

(3) It develops a general solution for forecasting many time series based on GBRT.

(4) It proposes a new strategy to generate multi-step ahead forecasts and provides experimental comparisons on various forecasting strategies for generating multiple steps ahead forecasts when using complex models with pooled data.

In addition it provided empirical evidence of the strong performance of a machine learning algorithm in demand forecasting, when evaluated over multiple series based on extensive out-of-sample data.

3. Methodology

We propose a scalable solution based on GBRT to predict mobile payment customer flows for a large number of stores concurrently. The main modules of our solution are shown in Figure 3.

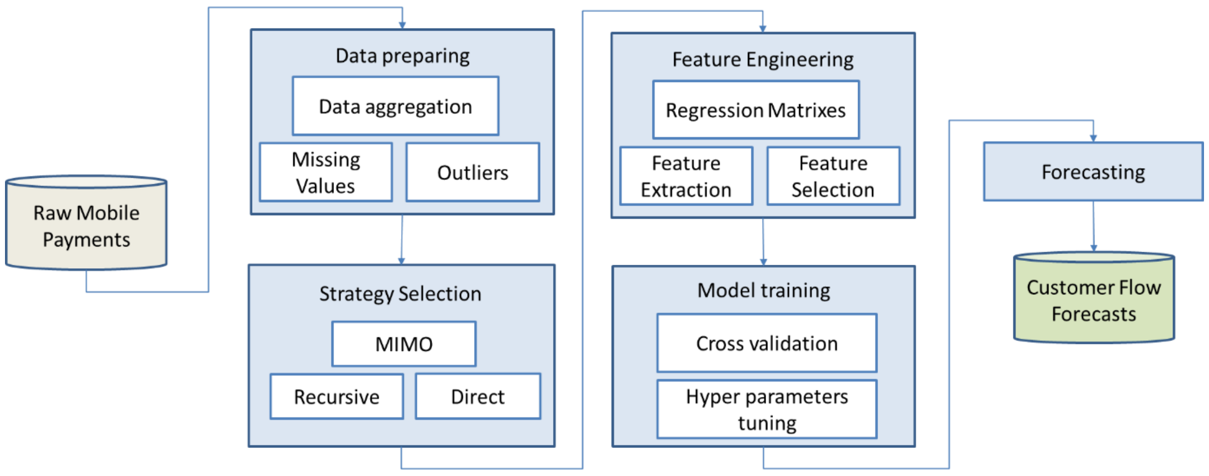


Figure 3. Flowchart for customer flow forecasting

Raw customer mobile payments are first aggregated into store specific daily mobile payment customer flow time series. After removing outliers and imputing missing values, the resulting time series are then transformed into a pool of forecasting strategy specific regression matrixes. Global and local features are extracted from the matrix and then important features are selected. After a parameter tuning procedure, GBRT models are trained to generate multi-day ahead forecasts. In the following subsections we focus on transforming the time series into a regression matrix format to make them suitable for applying our regression (GBRT) approach, the proposed method of feature extraction and the training module implementation.

3.1. Time series transformation for one-step ahead forecasting

Considering a pool of n time series $\aleph = (\mathbf{X}_1, \cdots, \mathbf{X}_n)^{\mathrm{T}}, \mathbf{X}_i = \left(x_{iS_i}, ..., x_{iT}\right),\ i \in [1, \cdots, n]$ where all the time series in the pool have the same end time, T, but not necessary have the same start time S. To forecast $x_{i,T+1}$ for all the time series, we need first train a GBRT model based on previous observations in $\aleph$. As GBRT is inherently a regression model, we need transform $\aleph$ into a regression matrix and define dependent and independent variables. For different forecasting tasks and strategies, we need to transform original customer flow time series into regression matrixes with different structures to facilitate the following training and forecasting processes.

$$
\mathbf{X}_i = \begin{array}{cccc|c|c}
\multicolumn{4}{c|}{\text{Predictors}} & \text{Dependent} & \\
x_{iS_i} & x_{i,S_i+1} & \cdots & x_{i,S_i+W\text{-}2} & x_{i,S_i+W-1} & \\
x_{i,S_i+1} & x_{i,S_i+2} & \cdots & x_{i,S_i+W-1} & x_{i,S_i+W} & \\
x_{i,S_i+2} & x_{i,S_i+3} & \cdots & x_{i,S_i+W} & x_{i,S_i+W+1} & \text{Training} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
x_{iK_i-1} & x_{iK_i} & \cdots & x_{i,T-1} & x_{i,T} & \\
\hline
x_{iK_i} & x_{i,K_i+1} & \cdots & x_{iT} & x_{iT+1} & \text{Forecasting}
\end{array}
$$

Figure 4. Regression matrix for one-step ahead forecasting

To transform a one-step ahead time series forecasting problem into a regression problem, it is straightforward to transform a time series into a Hankel matrix (Fig. 4), which has W columns and K_i=T-W-S_i+2 rows. The last column of $\mathbf{X}$ on the right is used as predicted variables for training and the other columns are used to construct predictors. The last row of the **X** is used to generate the one-step ahead forecast. As customer flows usually exhibit weekly cycles, we set the window width *W* to be multiples of seven days, and the optimal *W* is selected by our feature selection algorithm. We pool the matrixes from all the time series together to train the forecasting model, not training the *n* models for each time series individually. Pooling addresses the data availability issue by leveraging analogous customer flow time series to learn common patterns (Frees and Miller, 2004). Pooling observations across a group of similar time series is expected to lead to smaller variance than the individual forecasts, with fewer parameters to be estimated, adapting more rapidly to any structural changes in time series with robustness in the presence of outlier observations (Duncan et al., 2001).

One problem of constructing Hankel matrix **X** is the time series length T-S_i+1 need to be larger than window width *W*, so that we can obtain *K* rows of observation from this time series. When the length of one time series is less than *W*, we supplement it to length *W* with its mean of observed values.

3.2. Time series transformation for H-step ahead forecasting

In this research, we focus on three strategies to generate H-step ahead forecasts.

Recursive strategy

The Recursive strategy trains first a one-step model *f*

$$x_{t+1} = f(x_t, \dots, x_{t-L-1}) + \varepsilon_{t+1},$$

where *L* is the number of lags used as predictors. After generating one-step ahead forecasts, it constructs explanatory features for the next step with the newly generated forecasts, and then generates the next step of forecasts with the same model, until all *H* steps of forecasts are generated. The forecasts for horizon *h* can be expressed by

$$\hat{x}_{T+h} = \begin{cases} \hat{f}(x_T, \dots, x_{T-L-1}) & h = 1 \\ \hat{f}(\hat{x}_{T+h-1}, \dots, \hat{x}_{T+1}, x_T, \dots, x_{T-L-1}) & h \in \{2, \dots, L\} \\ \hat{f}(\hat{x}_{T+h-1}, \dots, \hat{x}_{T+h-L}) & h \in \{L+1, \dots, H\} \end{cases} \quad (1)$$

Therefore the regression matrix structure for the recursive strategy is the same as that for one step forecasting.

Direct strategy

The Direct strategy needs to build an individual model for each *h, h=1...H*, forecasting horizon; the prediction for horizon h is given by

$$\hat{x}_{T+h} = f_h(x_T, \ldots, x_{T-L-1}).$$

For horizon h, only inputs at least h steps before the target time period can be used to construct explanatory features. The regression matrix designed for the direct strategy is shown in Fig 5. The last H columns are used as predicted variables, and the last row is used to generate forecast for horizon 1 to H. To forecast $\hat{x}_{T+h}$, we need first train a model with column W+h-1 as the dependent variable. This strategy needs to learn H models to generate 1 to H ahead forecasts, therefore demands more computational time.

$$\mathbf{X}_i = \begin{array}{c|c}
\overbrace{\begin{array}{cccc} \phantom{x_{i,S_i+W-2}} & \text{Predictors} & & \end{array}} & \overbrace{\begin{array}{ccc} & \text{Dependents} & \end{array}} & \\
\end{array}$$

$$\mathbf{X}_i = \left[\begin{array}{cccc:ccc}
x_{iS_i} & x_{i,S_i+1} & \cdots & x_{i,S_i+W\text{-}2} & x_{i,S_i+W-1} & \cdots & x_{i,S_i+W+H\text{-}2} \\
x_{i,S_i+1} & x_{i,S_i+2} & \cdots & x_{i,S_i+W\text{-}1} & x_{i,S_i+W} & \cdots & x_{i,S_i+W+H-1} \\
x_{i,S_i+2} & x_{i,S_i+3} & \cdots & x_{i,S_i+W} & x_{i,S_i+W+1} & \cdots & x_{i,S_i+W+H} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
x_{iK_i-H} & x_{iK_i-H+1} & \cdots & x_{i,T-H} & x_{i,T-H+1} & \cdots & x_{i,T} \\
\hdashline
x_{iK_i} & x_{i,K_i+1} & \cdots & x_{iT} & \hat{x}_{i,T+1} & \cdots & \hat{x}_{i,T+H}
\end{array}\right]
\begin{array}{l}
\left.\begin{array}{c} \\ \\ \\ \\ \\ \end{array}\right\} \text{Training} \\
\left.\begin{array}{c} \\ \end{array}\right\} \text{Forecasting}
\end{array}$$

Figure 5. Regression matrix for Direct strategy

Pseudo MIMO strategy

MIMO strategy was initially proposed by Kline (2004), which has so far been limited to models that could generate multiple outputs, e.g. lazy learning (Taieb et al., 2012), multi-output Support Vector Regression (Bao et al., 2014), multi-output Neural Networks (Kline, 2004). But the existing GBRT algorithms do not support multiple outputs (we leave this as future work). Instead, we design a pseudo MIMO strategy which can use single-output models to generate multi-output forecasts. To do so, we transform a time series into a special regression matrix structure, which is shown in Fig.6. Similar to the regression matrices designed for one-step forecasting, the last column of the matrix is used as the predicted dependent variable(s), with the last H rows used to generate forecasts for horizon 1 to H concurrently, while the other rows are used for training. The matrix is separated into n= (T-S_i-W+2)/H slots with the same width H. In each slot, there are H predicted variables representing horizon 1 to H. Those H predicted variables have exactly the same candidate predictors, but they may have different seasonal and calendar event indicators that will be introduced in the next subsection. In this way, we can train one model but this generates H-step ahead forecasts without requiring iterations.

Pseudo MIMO does not accumulate errors across the forecasting horizon which is the main problem arising from using the Recursive strategy, and it can learn the commonality over the *H* forecasting horizons which Direct strategy cannot. Also, pseudo MIMO is more computational efficient compared with the other two strategies, as the size of its training set is nearly the same as that of Recursive strategy but it generate *H* forecasts concurrently without iterations.

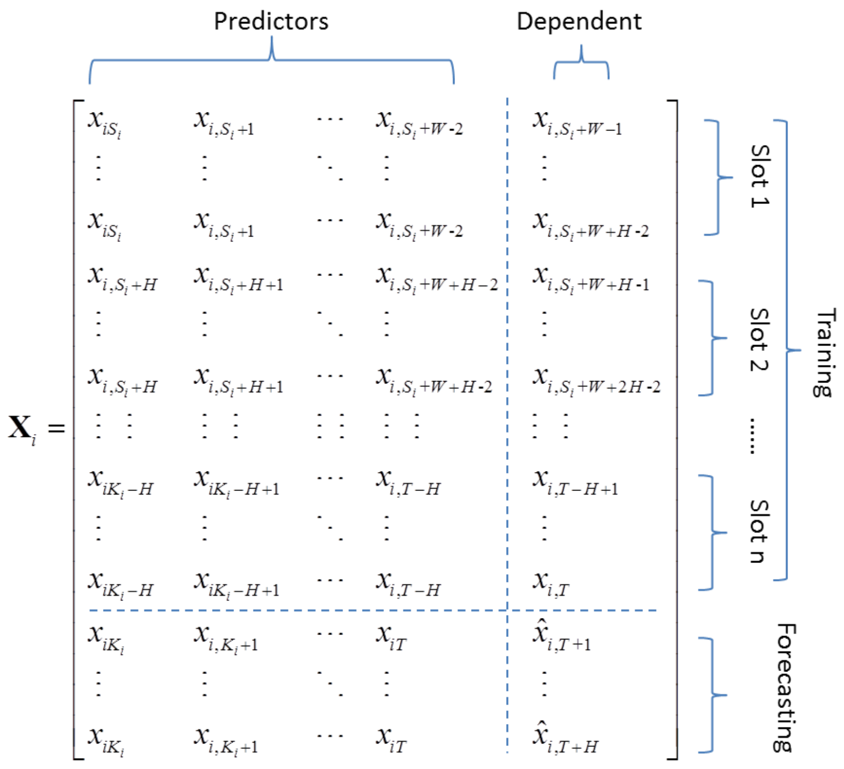


Figure 6. Regression matrix for MIMO strategy

3.3. Feature extraction

After the transformation, the next step is to extract a set of features from the regression matrix. The data enriched environment by pooling enables us to consider a large set of features capturing the dynamic characteristics of the time series in both the short and long term. First, a number of the lags in each row of the **X** are used as candidate predictors directly, these lags reflect the dynamics of the most recent customer flows. Second, a set of indicators which summarize time series' local dynamics and global static characteristics are also calculated to be candidates. The details of these candidate features are summarized in the Table 1. MM_1~MM_w are moving medians over last 1 to *w* weeks, which may capture the short term temporal trends in recent weeks. Similarly, M20_1~M20_w (or M80_1~M80_w) are moving 20 (or 80) percentiles over last 1 to *w* weeks, different measures which also aim

Table 1. Candidate features for mobile payment customer flow forecasting

Features	Description	Type	Characteristics
Lag_1~Lag_L	Lags 1 to L (L=min(21,W))	Numeric	Lags
MM_1~MM_w	Moving medians over last 1 to w=W/7 weeks	Numeric	Local dynamics
M20_1~M20_w	Moving 20 percentile over last 1 to w=W/7 weeks	Numeric	Local dynamics
M80_1~M80_w	Moving 80 percentile over last 1 to w=W/7 weeks	Numeric	Local dynamics
MSD_1~MSD_w	Moving standard deviation over last 1 to w=W/7 weeks	Numeric	Local dynamics
DW_1~DW_6	The ratio between the mean of the flow in the day of the week (Monday to Saturday) and the global mean	Numeric	Global summaries
City	The city of the store located	Categorical	Store specific
Category	The category in which the store belongs (e.g., grocery, fast food, bakers, etc.)	Categorical	Store specific
Store ID	Store identification	Categorical	Store specific
Comments	Number of comments towards the store on the platform	Numeric	Store specific
Avr-payments	Average payments for each purchase	Numeric	Store specific
Day of Week	From Monday to Sunday	Categorical	Seasonality
Holiday	Including New Year, Spring festival, Qingming day, Labor's day, Mid-autumn day, National day	Binary	Calendar events
Holiday_eve	The day before a holiday	Binary	Calendar events
Holiday_after	The day after a holiday	Binary	Calendar events
Temperature	The average temperature during the day	Numeric	Weather
Wind strength	The maximum wind strength during the day	Numeric	Weather
Precipitation	Five levels, from rainstorm (or blizzard) to drizzle (or slight snow)	Categorical	Weather

to capture any short term temporal trends. We use moving standard deviation over last 1 to *w* weeks, named as MSD_1~MSD_w, to capture the extent of variations in recent customer flows. DW_1~DW_6 are indicators of store specific day of week effects, calculated by the ratios between the means in each day of the week (Monday to Saturday) and the global mean of the store cross time. These global summaries are specially designed for GBRT, as they should help the tree splitting process to pool time series with similar characteristics together. In addition, as customer flows are often affected by seasons, calendar events, weather, and other store specific characteristics (e.g., location, category, and customer rating): we also include a set of features capturing these factors.

3.4. Feature selection

Considering the scale of the daily mobile payments data and the high dimensional of candidate feature space, it is computational infeasible for searching the best subset of features. Instead we design a sub-optimal heuristic algorithm for feature selection which mainly consists of three parts. First, (i) with a pre-set of core features included (such as Day of Week and calendar events) and a maximum estimation window *W*, the optimal number of lags (*L*) is determined; (ii) then the optimal window width *w* is determined to construct local indicators capturing the local dynamics; and (iii) finally, the best subset is selected from outside factors. The first two parts are based on a forward selection process, and the third part is based on an individual evaluation. The detail of the selection process is described in Algorithm 1.

3.5. Model training

The forecasting model selected in our solution is Gradient Boosting Regression Tree (GBRT) which has empirically proven to be highly effective in a number of applications as we discussed in section 2. The GBRT algorithm iteratively constructs *K* different weak learners which consist of regression trees of fixed size from the training set

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i),\ f_k \in \Omega \tag{2}$$

where $\Omega = \{ f(\mathbf{x}) \}$ is the space of regression trees. Each tree contains a continuous score on each of the leaves. For a given example, it uses the decision rules in the trees to classify it into the leaves and calculates the final prediction by summing up the score in the corresponding leaves. To learn the set of functions used in the model, GBRT minimizes the following regularized objective,

$$L(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Psi(f_k)\,. \qquad (3).$$

Here l is a differentiable convex loss function that measures the difference between the prediction and the target. The second term $\Psi\left(f_k\right)$ penalize the complexity of the model (i.e., the regression tree functions), which helps to smooth the final learnt weights to avoid over-fitting. The tree ensemble model in Eq. (2) includes functions as parameters and cannot be optimized using traditional optimization methods in Euclidean space. Instead, the model is

Algorithm 1.

```
Input: training dataset; candidate variables
Output: selected features
    1) Build candidate feature set with the maximum window width $W$=8 weeks;
    2) Extract last four weeks of data from training set as an evaluation set for feature
       selection;
    3) Manual selection of a core feature set, which includes global summaries
       (DW_1~DW_6), seasonality (Day of Week), and calendar events (Holiday,
       Holiday_eve, and Holiday_after).
    4) Select the optimal number of lags $L$ by one-dimensional searching in the range
       of 1 to 21. Specifically, train 21 GBRT models: $j$th model uses features including
       1 to $j$ lags together with features in the core feature set, optimal $L$ is given by the
       model provide most accurate forecasts on the evaluation set. Updating core
       feature set by adding the optimal number of lags.
    5) Selecting the optimal window width $w$ for building local indicators (row 2-5 in
       Table 1) by one-dimensional searching in the range of 1 to 8 weeks. Specifically,
       training 8 GBRT models, $j$th model uses 1 to $j$ weeks of flows to build local
       indicators, training the model together with features in the core feature set;
       optimal $w$ is given by the model providing most accurate forecasts on the
       evaluation set. Update core feature set by adding the local indicators built with
       the optimal window width.
    6) Evaluating the value of the other candidate features one by one by comparing the
       forecasting accuracy on evaluation set when using or discarding the target
       features, and adding the feature into core feature set if it improves the forecasts.
    7) Return the core feature set.
```

trained in an additive manner. Formally, let $\hat{y}_i^{(t)}$ be the prediction of the i-th instance at the t-th iteration, f_t is then added to minimize the following objective

$$L^{(t)} = \sum_i l\left(y_i, \hat{y}_i^{(t-1)} + f_t\left(\mathbf{x}_i\right)\right) + \Psi\left(f_t\right). \qquad (4).$$

This means f_t is added to improve the model most according to Eq. (4). Boosting algorithms vary in how they quantify lack of fit and select settings for the next iteration. In this research, we adopt a second-order approximation boosting algorithm, named as XGboost, which can quickly optimize the objective in the general setting (Chen and Guestrin, 2016),

$$L^{(t)} \simeq \sum_{i}\left[l\left(y_i, \hat{y}_i^{(t-1)}\right)+g_i f_t(x_i)+\frac{1}{2}h_i f_t^2(x_i)\right]+\Psi\left(f_t\right) \tag{5}$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$ are the first and second order gradient statistics of the loss function. XGBoost is available as an open source package which has been widely recognized in a number of machine learning and data mining challenges.

XGBoost has more than twenty tunable parameters that affect its learning performance. We follow the parameter tuning guide in the XGBoost manual and select six parameters for tuning.

Max depth: the maximum depth of a tree;

Gamma: the minimum loss reduction required to make a split;

Subsample: denotes the fraction of observations to be randomly samples for each tree;

Colsample by tree: the fraction of columns to be randomly samples for each tree;

Lambda: L2 regularization term on weights (analogous to Ridge regression);

Alpha: L1 regularization term on weights (analogous to Lasso regression).

Similar to our feature selection algorithm, we extract the last four weeks of data from the training dataset as an evaluation set for parameter tuning. The steps to be performed are: choosing a relatively high learning rate; and determining the optimum number of trees for this learning rate; then tuning tree-specific parameters (max depth, gamma, subsample, colsample by tree) for decided the learning rate and the number of trees, and then tuning the regularization parameters (lambda, alpha). In the last step we lower the learning rate and decide the optimal parameters.

4. Experiments design

4.1. Data

To investigate the forecasting accuracy of different methods on daily mobile payment customer flows, we conduct a series numerical experiments on a randomly selected 2000 stores sample including 19.6 million platform payments log from July 2015 to October 2016. In addition, we also obtain descriptive information on each store, e.g. store ID, location, category, and rating. We aggregate user payments according to store ID and date to form store specific daily customer flows. Table 2 presents the statistical summaries on daily customer flows in various store categories.

Table 2. Descriptive statistics of the daily mobile payment customer flows

No.	Category	Number of stores	Average length of history (days)	Mean of daily payments	Median of daily payments
1	Convenience Store	207	284.88	98.88	72
2	Supermarket	372	266.34	187.49	120
3	Pharmacy	4	161.75	60.12	57
4	Baking	272	296.30	87.43	73
5	Restaurant	127	264.09	73.86	63
6	Fast Food	839	333.42	105.93	81
7	Cafe	179	276.97	92.41	75

Figure 7 depicts the average daily mobile payment customer flows over the selected stores. It is easy to observe that the average customer flow has an upwards trend: this may be mainly due to the increasing number of users making payments through their mobiles. We can also observe that customer flow exhibits strong weekly cycles, and has different patterns during holidays (vertical shadows on the Figure 7).

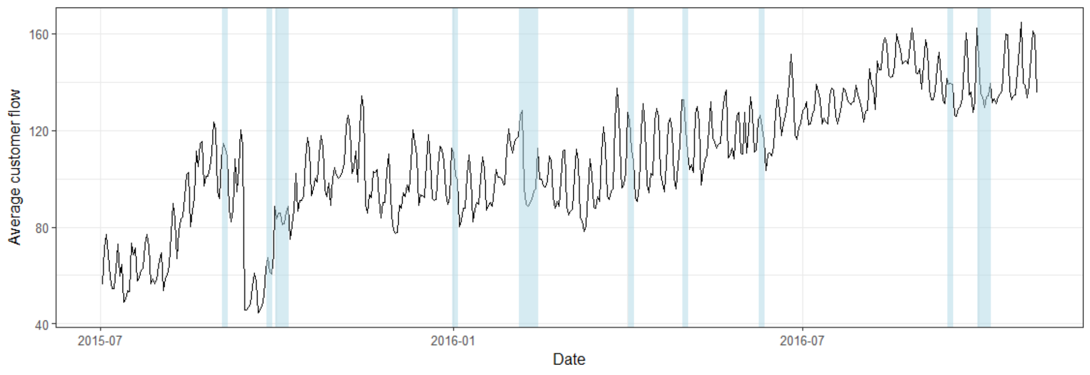


Figure 7. Average daily mobile payment customer flow over 2000 stores

Figure 8 shows a sample of daily mobile payment customer flows at individual store level. Customer flows are in general non-stationary, with different weekly cycles and length of history. Some of them exhibit trend (of various forms), have very limited historical information, and contain outliers and breaks. We identify and remove outliers with the ‘tsoutliers’ function in the ‘forecast’ R package v.8.4 (Hyndman and Khandakar, 2008). Breaks are treated as missing values that are imputed with a self-designed K nearest neighbor (KNN) algorithm considering both day of week effects and local trends (see Appendix A for more detail). Those imputed values are only used to construct predictors, and are neglected when appearing as dependent variables in the following training and test processes.

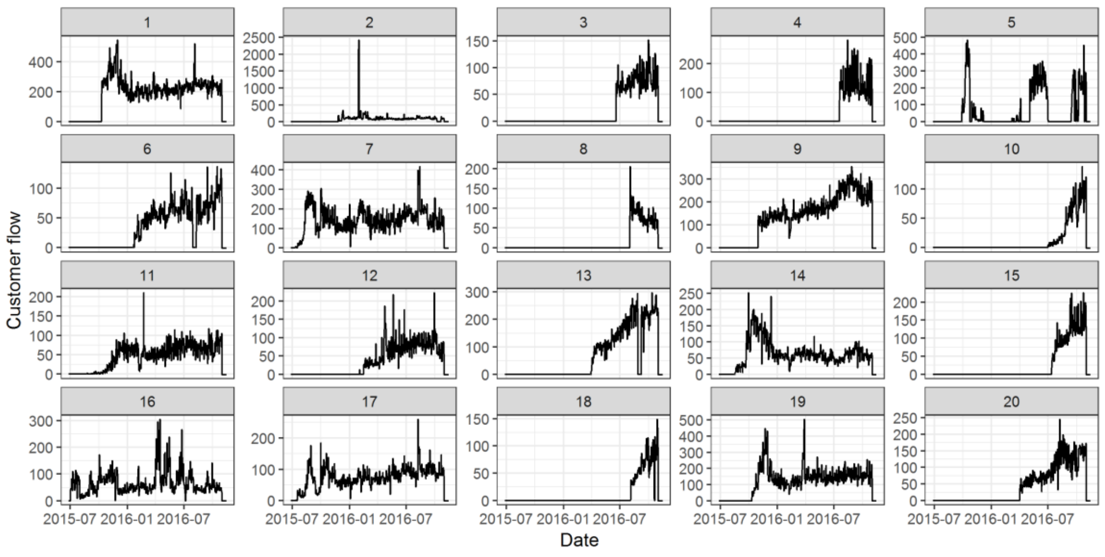


Figure 8. A sample of customer flows over time

4.2 Training and test set definition

We separate the data into three subsets, one for model training and two for forecasting accuracy evaluation. The training set spans 401 days, from July 1, 2015 to August 8, 2016. Two test sets each consists of six weeks of customer flow data. The first spans from August 9, 2016 to September 19, 2016, and the second from September 20 to October 31, 2016 which contains seven days of holiday (October 1 to 7 are the Chinese National days). We estimate the models and generate forecasting with a rolling scheme. For every two weeks, we extend the estimation window forward throughout the remaining sample period.

4.3. Forecasting evaluation metrics

We use four error measures to compare the forecasting performance of the models. The first is sMAPE, symmetric Mean Absolute Percentage Error, which measures the difference between the prediction and the actual outcome. sMAPE has also been adopted by NN3, NN5 competitions; it is here defined as:

$$\text{sMAPE} = \frac{1}{nH}\sum_{i=1}^{n}\sum_{h=T+1}^{T+H}\left|\frac{\hat{x}_{ih} - x_{ih}}{\hat{x}_{ih} + x_{ih}}\right|,$$

where $\hat{x}_{ih}$ is the forecasts customer flow of store *i* in horizon *h*, and x_{ih} is the observed customer flow of store *i* in day *h*. The second is Median Absolute Percentage Error (MdAPE), which is here defined as:

$$\text{MdAPE} = \underset{\substack{i \in [1,\cdots n] \\ h \in T+[1,\cdots,H]}}{median}\left|\frac{\hat{x}_{ih} - x_{ih}}{\hat{x}_{ih} + x_{ih}}\right|.$$

Compared with sMAPE, MdAPE is more robust to potential outliers. The last criterion we used is based on relative errors. The Average Relative Mean Absolute Error (AvgRelMAE) is proposed by Davydenko and Fildes (2013) for measuring forecasting accuracy at a disaggregate level (e.g. Store sales, SKU-level demand). It is a geometric mean of the ratio of the MAE between the candidate model and the benchmark model.

$$AvgRelMAE = \left(\prod_{i=1}^{n}\frac{\sum_{h=\mathrm{T}+1}^{T+H}\left|\hat{x}_{ih}^{f} - x_{ih}\right|}{\sum_{h=\mathrm{T}+1}^{T+H}\left|\hat{x}_{ih}^{b} - x_{ih}\right|}\right)^{\frac{1}{n}}$$

where *N* is the number of stores in the sample, $\hat{x}_{ih}^{b}$ is the baseline statistical forecast for series *i*, $\hat{x}_{ih}^{f}$ is the candidate model *f* evaluated for series *i*. The AvgRelMAE has the advantages of being scale independent and robust to outliers, with a straightforward interpretation: a value smaller than one indicates an improvement by the candidate model over the benchmark.

In order to measure the forecasting error bias, the Mean Percentage Error (MPE), that is defined here as the mean of ratios of total error to total customer flows in the test periods per store, i.e.,

$$\text{MPE} = \frac{100}{n}\sum_{i=1}^{n}\left(\frac{\sum_{h=T+1}^{T+H}\left(\hat{x}_{ih} - x_{ih}\right)}{\sum_{h=T+1}^{T+H} x_{ih}}\right).$$

is used as the final criterion.

To evaluate whether the forecast error differences in models that may appear are due to randomness, we employ the non-parametric Friedman test and the post-hoc Nemenyi test (Demšar, 2006; Koning et al., 2005). We use the implementation of the tests available in the tsutils R package (Kourentzes, 2019).

4.4. Benchmark models

To compare the forecasting accuracy of our proposed methods, seven models have been selected to be benchmarks. We include both simple methods such as 'naïve' random walk and more complicated methods that incorporate some of the features in our novel approach. The benchmarks include some that we would expect be hard to beat as well as some that would be easy to employ in practice. These models are explained in detail as the following.

(1) Last Week. The forecasts are the observed values for the same day of week in the last observable week.

(2) Naive. The forecasts are the observed values for the last observable day.

(3) ETS. An automated ETS algorithm of Hyndman & Khandakar (2008), using ets functions in the 'Forecast' R package v.8.4 using its default settings.

(4) Theta. A decomposition technique proposed by Assimakopoulos and Nikolopoulos (2000), which was the top performer in M3-Competition. It is implemented with the 'thetaf' function in the 'Forecast' R package using its default settings.

(5) ARIMA. The automatic ARIMA function ('auto.arima') implemented in the 'Forecast' R package is used to identify and estimate the model using its default settings.

(6) Pooled regression. Using the same predictors selected by GBRT, estimated with penalized L-1 regression which is named Lasso by Tibshirani (2011). It is implemented using the 'glmnet' package v.2.0-16 in R. The optimal value of the penalty parameter is determined by 10-fold cross-validation.

(7) Random Forest (RF): first proposed by Breiman (2001). Random Forest is based on decision trees and combined with aggregation and bootstrap ideas, first proposed by Breiman (2001). Random Forest has widely been used in applications, see Ziegler and König (2014) for recent surveys. In this research, we adopt a fast implementation of random forest, the 'Ranger' package v.0.11.2 in R (Wright and Ziegler, 2017), which is particularly suited for high dimensional data.

(8) GBRT. Implemented with the 'Xgboost' package v. 0.82.1 in R (Chen and Guestrin, 2016).

5. Results

The training set contains 543888 observations and more than one hundred features. Both feature selection and hyper-parameter tuning processes require significant computing capability. To facilitate our experiments, we run all the algorithms programmed with R on a workstation with two Intel Xeon E5-2630 processors and 128 Gigabyte of memory.

5.1. Feature selection

As described in the Algorithm 1, we extract the last eight weeks of data from the training set as the evaluation set for feature selection using Xgboost. We first search for the optimum number of lags in a range of 1 to 21 days. We use one-step ahead forecasts, evaluated by sMAPE, as the criterion for the selection. The Nemenyi test results for the rank of models with various lags are reported on the left side of Figure 9. It shows that the lag of 15 days has the lowest value of mean rank over 86772 observations[3], but no difference is seen with that of 7 days lag at the 5% significance level. To keep the model as simple as possible, we select 7 days as the optimum number of lags for the following experiments. Then we build local dynamics using window width from 1 to 8 weeks, and search the optimum window width in the similar way. We finally select 28 days (4 weeks) of window width for building local summaries of the dynamics (the right side of the Fig. 9).

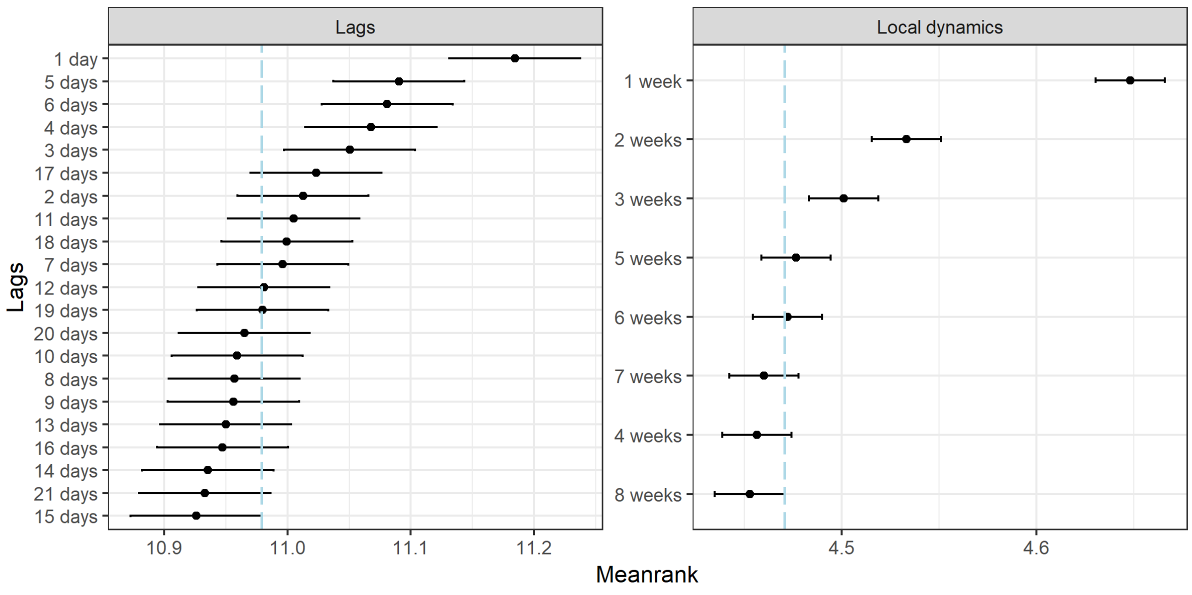


Figure 9. Nemenyi test results for rank of models with various lags and window widths for constructing local summaries at 5% significance level. The critical distance for the Nemenyi test is 0.106 and 0.036 respectively.

After the selection of lags and the window width for building local summaries, we then evaluate the value of the remaining candidate features one by one. Figure 10 depicts the

[3] We did not evaluate the rank at store level because of the unbalanced observations for each store in the validation periods.

Nemenyi test results on the models containing different feature sets, each feature sets consist of the core feature set, selected lags and dynamic lag summaries, and the feature under tested. The features which could significantly improve the forecasts over the baseline, which do not include any features under selected, are retained in the model. We find that only 'Temperature' decreases the sMAPE significantly, so all the other features in Figure 10 are dropped.

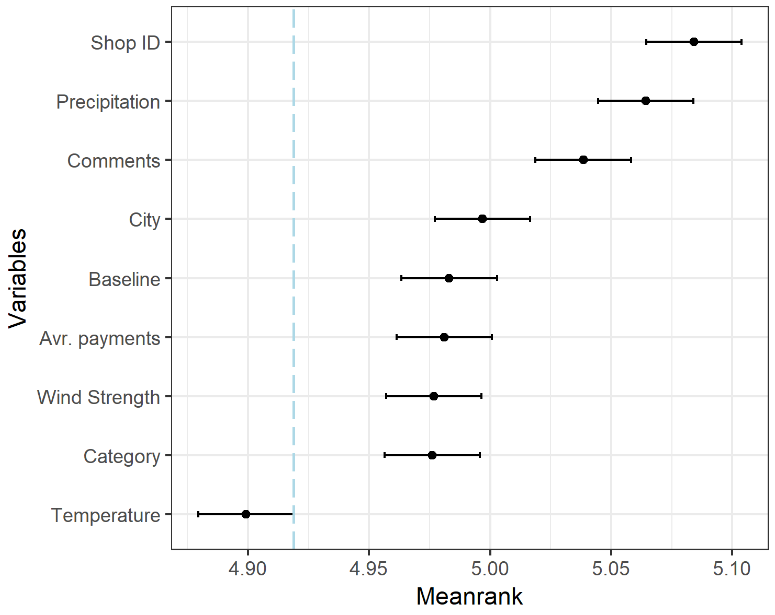


Figure 10. Nemenyi test results for rank of models using different feature sets at the 5% significance level. The critical distance is 0.039.

XGboost can provide estimates of feature importance from a trained model. Importance is calculated for a single decision tree by the amount that each attribute split point improves the performance measure, weighted by the number of observations the node is responsible for. The feature importance is then averaged across all of the decision trees within the model[4]. We report the feature importance ranking from the model with selected features on the training set in Figure 11. We can find that the most important features are Lag 1, the median/ 20 percentile 20/ 80 percentile of flows in last week, the global day of week ratios, and day of week indicators. Perhaps surprisingly, the only weather variable turns out to have limited impact on improving accuracy. This may be due to the nature of the sample of stores which is mainly composed of commodity stores and restaurants: their customer flows are less affected by the weather in Chinese cities.

[4] Though the feature importance could also be used to select features, it is easy to be affected by overfitting.

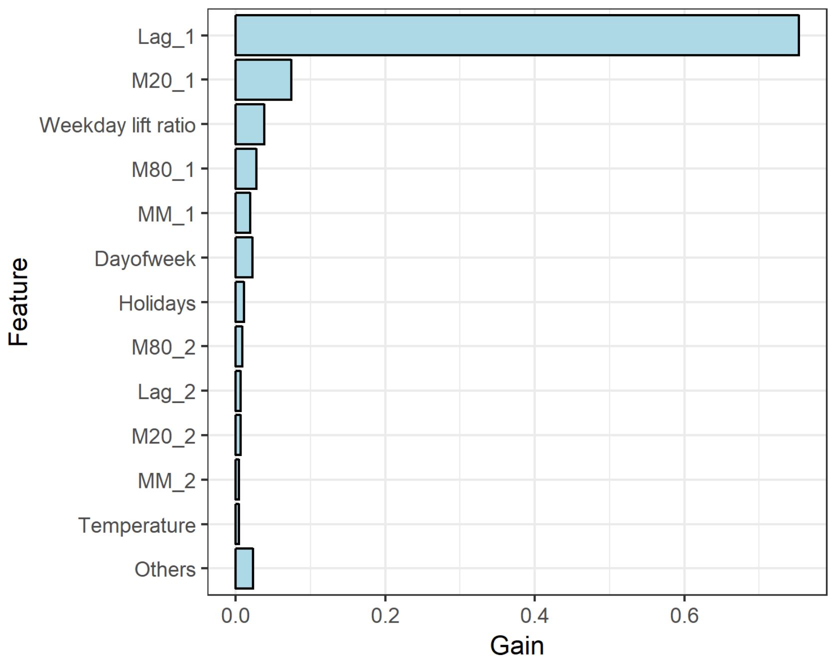


Figure 11. Rank of feature importance on selected features

5.2. Parameter tuning

The parameters tuning results for XGboost are depicted in Figure 12. We first choose 0.02 as the learning rate, and 1000 as the rounds of training. Except for max-depth, the out-of-sample forecasts are in general not sensitive to the value of these hyper-parameters. The optimal value for alpha, colsample, depth of trees, gamma, lambda, and subsample are selected as 0, 0.5, 10, 0.2, 0.7, and 0.8 respectively for the following experiments. After the tuning, we then reduce the learning rate to be 0.01, and find that the optimal rounds of training are around 2500.

Random forests implemented in the Ranger package for time series regression has four tunable parameters, i.e., number of trees, number of variables to possibly split at each node, minimal node size, and fraction of observations to sample. Similar to the tuning results on XGboost, we also use a grid searching algorithm for tuning random forests, and the last four weeks of training data is used for evaluation. The optimal values for the four parameters are 600 (for the number of trees), 10 (the number of variables to split at each node), 5 (minimal node size) and 0.9 (the fraction of observations to sample).

As Lasso is a much more efficient algorithm than XGboost and Ranger. For each rolling periods, we determine the optimal values of the penalty parameter by 10-fold cross-validation separately. The Cross-validation plot for the first rolling period is illustrated in Fig.B1 in Appendix B

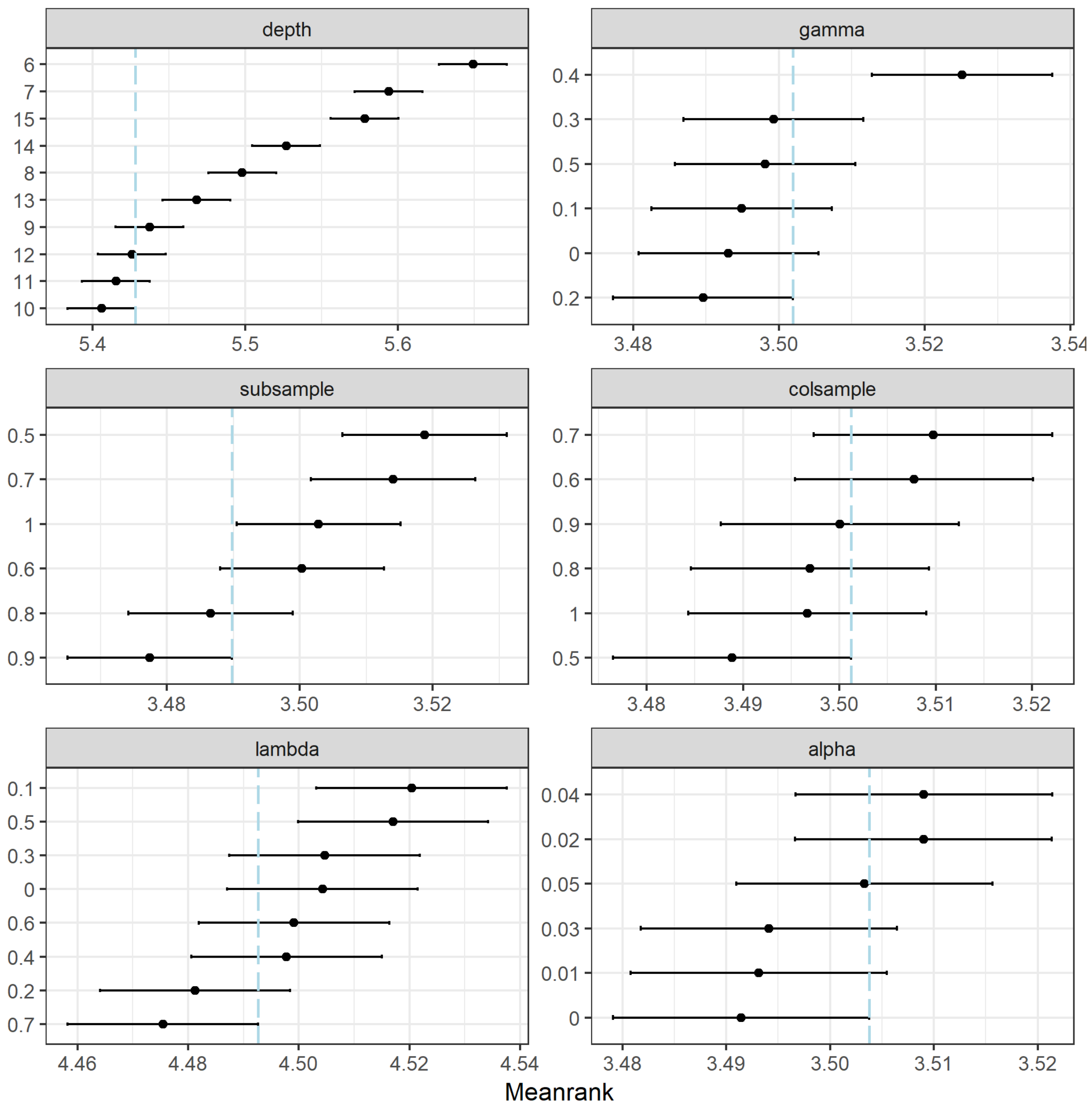


Figure 12. Nemenyi test results for the rank of models trained with different value of hyper-parameters at 5% significance level on. The critical distance is 0.044, 0.025, 0.025, 0.025, 0.034, and 0.025 for max-depth, gamma, subsample, colsample, lambda, and alpha respectively.

5.3. One-day ahead forecasts

The one-day ahead forecasting accuracies from all the benchmark models are shown in the Table 3. The ‘Last Week’ forecasts are used as the baseline for calculating AvgRealMAE. In both test sets, time series models which model customer flows for each store individually perform worse than regression tree based pooled models, i.e., RF and GBRT. The GBRT realized by the XGboost algorithm provides the most accurate forecasts on both test sets and all three accuracy metrics. But pooled regressions are in general inferior to several time series models, e.g., ETS and ARIMA. The results provide the evidence that data pooling can provide more accurate forecasts than the time series models, but here only when using more complex methods. The MPEs of the tree models are both negative but the values are small, indicating that the overall forecasts from these models are only slightly biased towards the pessimistic.

Table 3. One-day ahead forecasts

	Test set 1				Test set 2				Test set 1&2			
	sMAPE	MdAPE	AvgRel MAE	MPE	sMAPE	MdAPE	AvgRel MAE	MPE	sMAPE	MdAPE	AvgRel MAE	MPE
Last Week	0.130	0.082	1.000	6.330	0.150	0.103	1.000	-0.781	0.138	0.091	1.000	-0.973
Naive	0.120	0.080	0.943	4.322	0.112	0.077	0.776	0.062	0.115	0.079	0.833	0.033
ETS	0.103	0.066	0.789	2.309	0.099	0.069	0.674	-2.734	0.099	0.067	0.713	-2.360
Theta	0.106	0.069	0.831	5.093	0.100	0.069	0.689	-0.838	0.102	0.068	0.741	-0.186
ARIMA	0.105	0.071	0.819	2.324	0.108	0.078	0.731	-2.788	0.105	0.074	0.758	-2.457
Lasso	0.114	0.076	0.865	0.678	0.109	0.078	0.726	-0.047	0.110	0.077	0.777	0.083
RF	0.104	0.064	0.745	-5.463	0.096	0.067	0.637	-5.684	0.097	0.065	0.674	-5.326
GBRT	0.097	0.057	0.676	-4.314	0.088	0.061	0.583	-3.826	0.090	0.058	0.614	-3.713

N.B. The best performance is shaded.

In Fig. 13, the Nemenyi test intervals for all eight models on store level accuracy measures over the whole test periods are displayed. For all the measures, the intervals of the GBRT has no any overlap with other models, and hence, GBRT performs significantly better than the others.

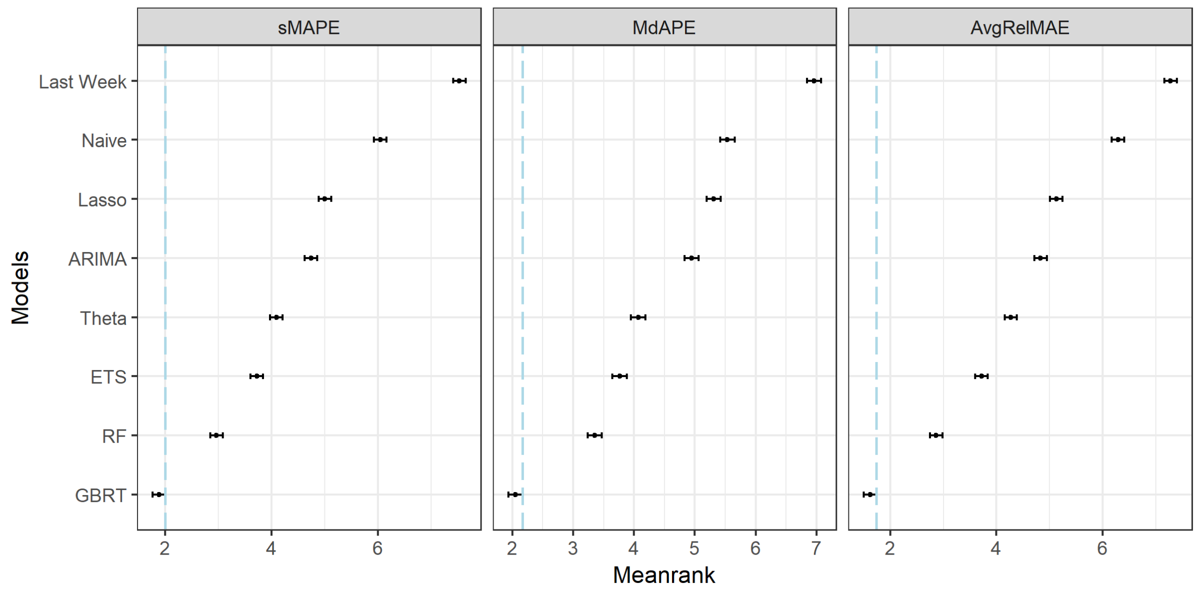


Figure 13. Nemenyi test results at 5% significance level on store level one-step ahead forecasts over the whole test periods. The critical distance for the Nemenyi test is 0.234.

Fig. 14 shows the boxplots on the store level accuracy differences between GBRT and time series models. We can find that more than three quarters of stores could have obtained more accurate forecasts using the GBRT compared to the time series models. This result show that most of the stores could benefit from the platform customer flow forecasting as pooling customer flow information from many stores improves their individual forecasting accuracy and robustness.

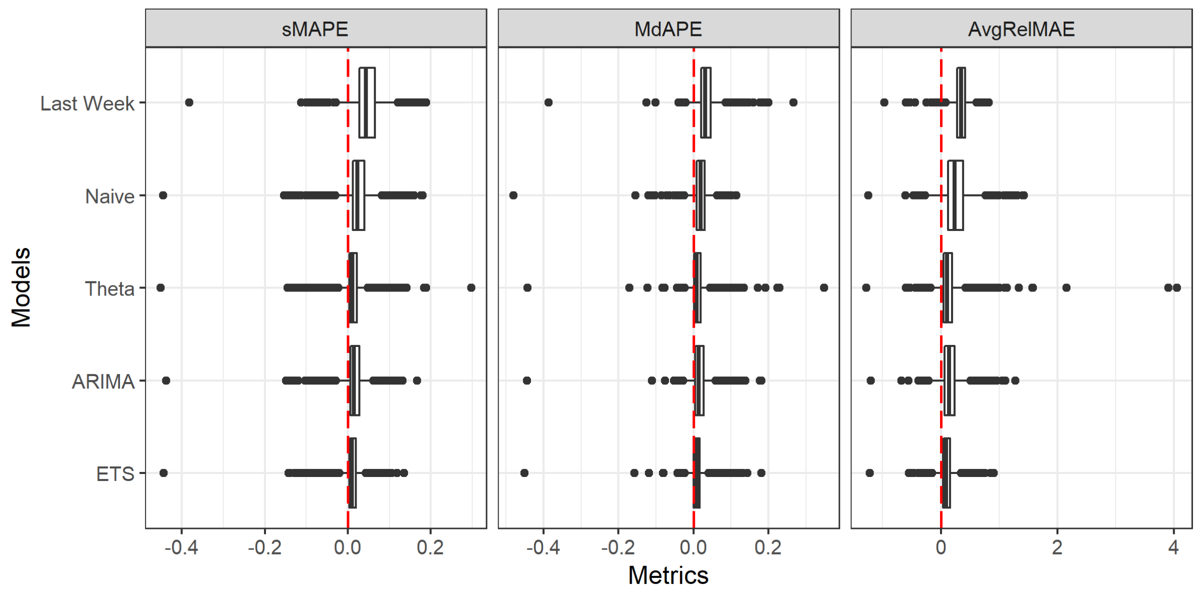


Figure 14. The store level one-step ahead forecasting accuracy differences between GBRT and five time series models over the whole test periods.

5.4. H-step ahead forecasting

We generate 1-14 days ahead customer flow forecasts for each store by means of three strategies for pooled models. Such a horizon is long enough to help the retailers in their operational planning decisions. As time series models are not designed to implement Direct or MIMO strategies, we only compare the forecasts among Pooled regression, Random Forest, and GBRT. Again, the Last Week forecasts work as the baseline for calculating AvgRelMAE. The forecasting results are reported in Table 5.

Contrasting with the one-step ahead results, for multiple-step forecasting, benchmark models have different forecasting performance on the two test sets with different strategies. GBRT together with pseudo MIMO provide the most accurate forecasts on almost all the test sets and accuracy metrics, except for sMAPE on test set 1 where GBRT-Direct and GBRT-

Recursive shows a little higher accuracy (but the Nemenyi test on store level ranks still indicate pseudo MIMO is the best performed model in test set 1). Random forest (RF) together with pseudo MIMO also provide the better forecasts than with other multi-horizon strategies. The results show the importance of using the right strategy to generate multi-step ahead forecasting when using complex models on pooled data.

Table 5. 1-14 days ahead forecasts

	Test set 1				Test set 2				Test set 1&2			
	sMAPE	MdAPE	AvgRel MAE	MPE	sMAPE	MdAPE	AvgRel MAE	MPE	sMAPE	MdAPE	AvgRel MAE	MPE
Last Week	0.135	0.086	1.000	1.245	0.150	0.102	1.000	-0.380	0.141	0.092	1.000	-2.179
Naive	0.154	0.102	1.138	-2.182	0.159	0.111	1.033	-5.901	0.155	0.105	1.082	-6.203
ETS	0.128	0.082	0.939	0.426	0.124	0.086	0.816	-2.713	0.125	0.083	0.874	-3.912
Theta	0.132	0.085	0.992	4.612	0.126	0.087	0.840	0.200	0.128	0.085	0.913	-0.316
ARIMA	0.136	0.091	1.009	1.031	0.135	0.097	0.891	-2.648	0.134	0.094	0.948	-3.652
Lasso-Recursive	0.140	0.096	1.052	6.609	0.129	0.096	0.850	1.480	0.133	0.095	0.940	0.659
RF-Recursive	0.145	0.100	1.093	8.639	0.139	0.101	0.924	3.428	0.140	0.100	0.994	1.919
GBRT-Recursive	0.113	0.073	0.827	4.690	0.114	0.079	0.758	-1.105	0.112	0.076	0.794	-2.027
Lasso-Direct	0.131	0.085	0.980	9.229	0.126	0.091	0.835	2.764	0.127	0.088	0.898	1.731
RF-Direct	0.128	0.086	0.944	-3.236	0.126	0.092	0.835	-8.080	0.126	0.089	0.881	-8.663
GBRT-Direct	0.113	0.072	0.814	-0.273	0.105	0.073	0.699	-3.659	0.107	0.072	0.753	-4.425
Lasso-pMIMO	0.138	0.095	1.013	3.077	0.123	0.088	0.810	0.782	0.128	0.091	0.901	0.803
RF-pMIMO	0.124	0.078	0.882	-4.343	0.113	0.082	0.749	-8.174	0.116	0.081	0.809	-6.539
GBRT-pMIMO	0.116	0.068	0.793	-3.770	0.101	0.069	0.663	-5.496	0.106	0.069	0.723	-4.666

N.B. The best performance is shaded; pMIMO is an abbreviation of pseudo MIMO.

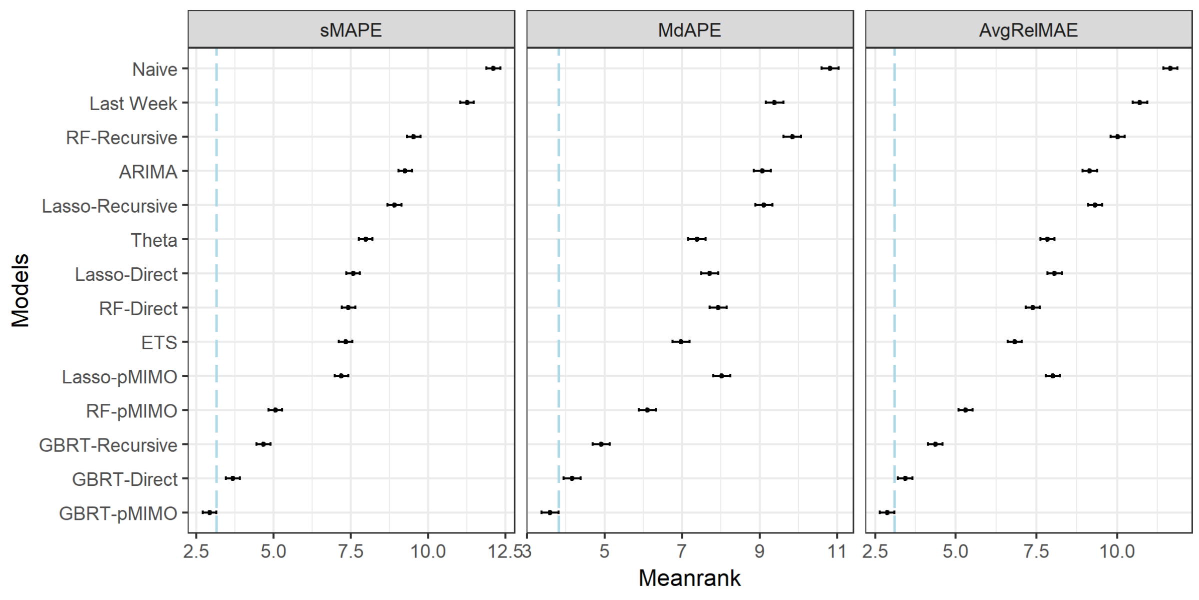


Figure 15. Nemenyi test results at 5% significance level on store level 1-14 days ahead

forecasts over the whole test periods. The critical distance for the Nemenyi test is 0.443.

Fig. 15 shows the store level Nemenyi test intervals calculated based on the whole test periods for all 14 models on three accuracy measures. For all the measures, GBRT together with pMIMO shows significantly better performance than all the others, and the GBRT together with the Direct strategy is average ranked second, which is also significantly out-performing the third ranking method, i.e., Random forest with pMIMO.

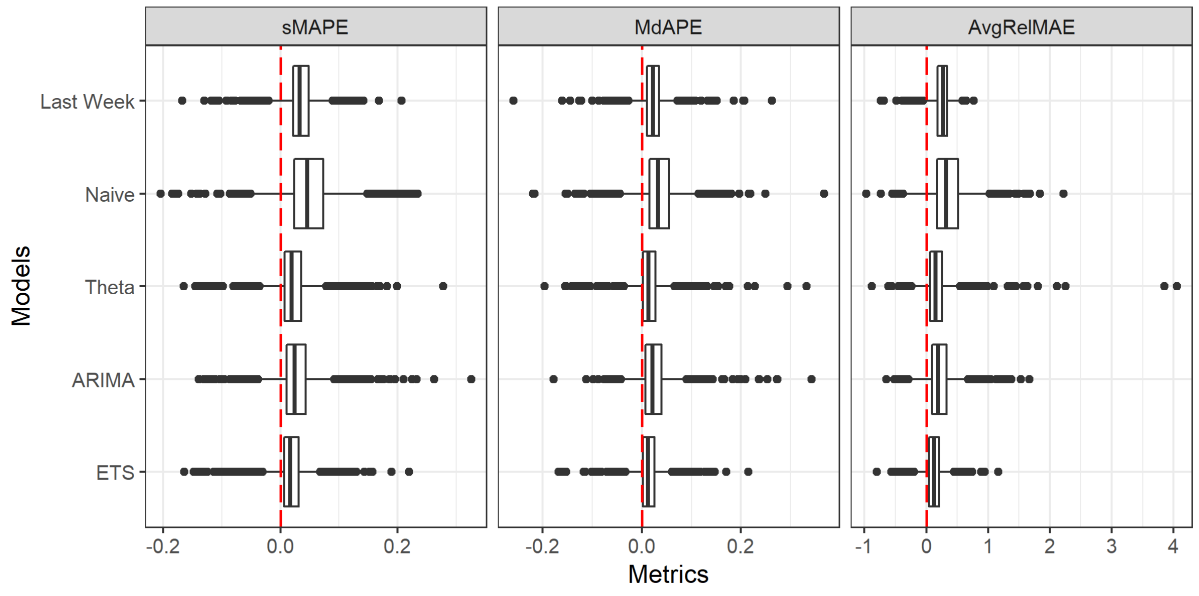


Figure 16. The boxplots on the 1-14 days ahead store level forecasting accuracy difference between GBRT-pMIMO and five time series models over the whole test periods.

Fig. 16 shows the boxplots of the store level accuracy difference between GBRT and time series models over the whole test periods for multiple steps ahead forecasting. We find similar results to those for one step ahead where, for more than three quarters of stores, we have obtained more accurate forecasts across all measures using the GBRT-pMIMO compared to the time series models.

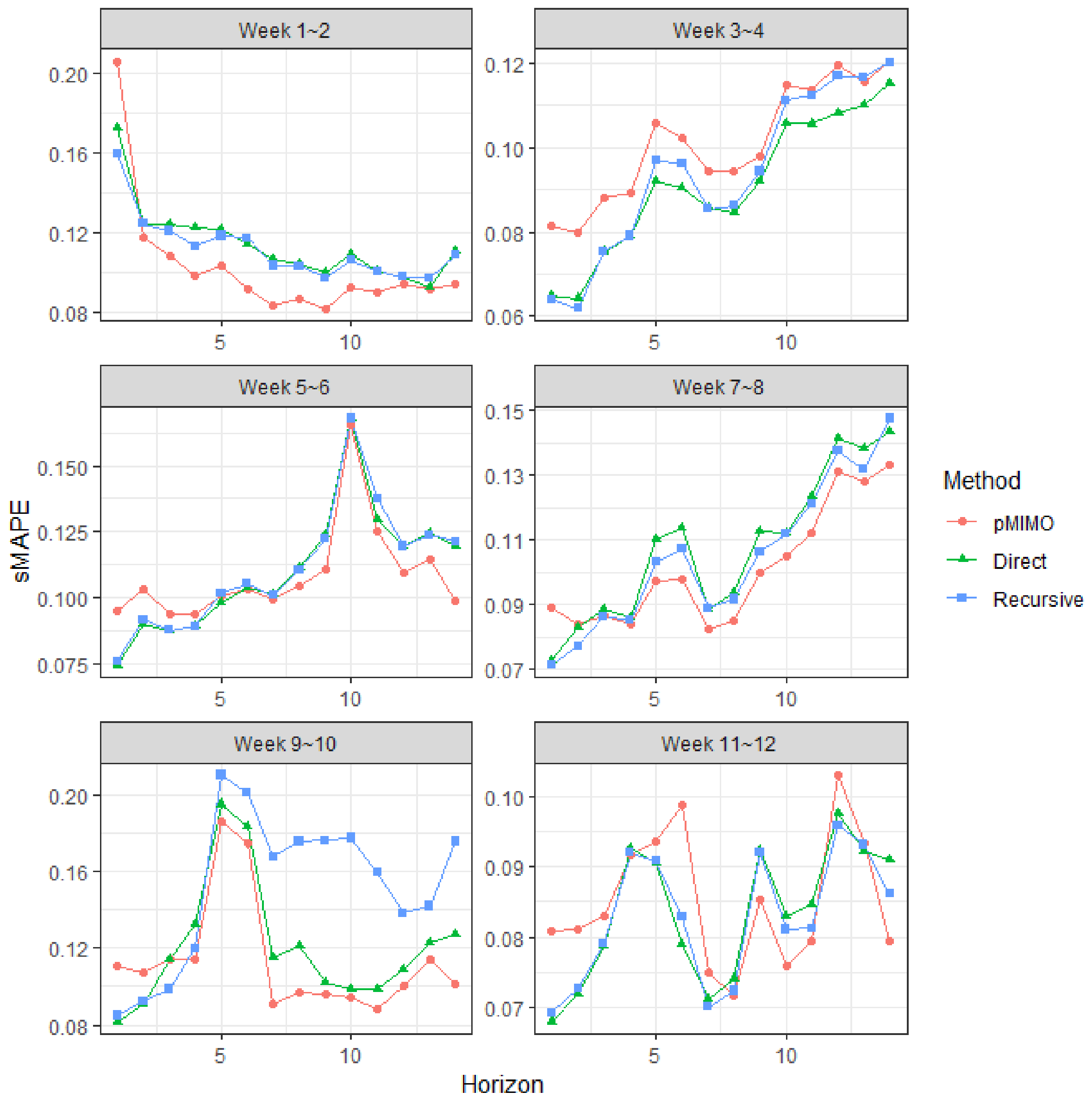


Figure 17. The forecasting accuracy comparisons across 1-14 horizons on three strategies with GBRT over six rolling test periods.

To obtain a deeper understanding on the performance of the three forecasting strategies, in Figure 17, we depict the forecasting accuracy comparisons measured by sMAPE across 1-14 horizons on the three strategies with GBRT over six rolling test periods. It is clear that pMIMO performs well in most time periods, exhibiting robust forecasting results. Recursive has similar performance to Direct in general, but a large error on Weeks 9~10 makes the Recursive strategy accumulate the error subsequently. This is mainly due to a switch between holidays and working days during the period. This indicates that Recursive strategy is sensitive to outliers. Another finding is that on all six rolling test periods, pseudo MIMO performs not as well as Recursive and Direct for shorter horizons (usually horizon 1 and 2), but perform well for longer time horizons. This means that it is possible to combine the forecasts on different horizons from different strategies to improve the forecasting accuracy further.

6. Implication for practice: prediction of total customer flows

An important implication for practice from the daily mobile payment customer flow forecasting method presented here is it can be utilized to further derive total customer flows. Total customer flows can be decomposed into customer flows using mobile payment and customer flows using other payment methods. The latter part is unobservable and therefore unpredictable by the third party mobile payment platform. But the shares of the two parts in total customer flows can be estimated by retailers, either judgmentally or through their own private transaction data. Given the estimated share of mobile payment customers, the total customer flows can be derived simply by multiplying the forecasts of mobile payment customer flows with the reciprocal of the estimated share.

Compared to the problem of accurate forecasting of total customer flows directly, for small retailers, the short term estimation of the share of mobile payment customers is a much easier task. This is due to the fact that the share is usually stable in the short term and less likely to be affected by a number of the exogenous variables, e.g., weather, day of week effects, or holidays, etc. Though the share of mobile payment customers is changing over time from a longer perspective, increasing in fact, it is not a strict assumption that the share of the mobile payments is stable in very short term forecasting, and retailers could also adjust their estimation of the share over time. The key point is that an individual retailer could make use of the mobile payment customer flow forecasts generated from this large data base for their customer flow predictions without experiencing extra costs.

7. Conclusions

Mobile payment data collected by third-party payment platforms is one of newly emerging source of big data and this has the potential to help millions of small retailers to improve their operational decision. We propose a GBRT based machine learning solution for forecasting stores' daily mobile payment customer flows, which is a novel pooled approach to capitalize on the commonalities in the data across the participating retailers.

We find that daily mobile payment customer flow forecasting based on a large pool of stores that includes a variety of categories, can generate more accurate forecasts than the forecasts generated by methods based on each store individually (as shown in Tables 4 and 6).

This is an interesting result because it shows that stores could obtain more accurate customer flow forecasting from participating on the shared platform than relying on the forecasts derived using only their own data. We also find that customer flows on the last observed day, median/percentile 20/percentile 80 of flows in last observed week, and the global day of week ratios are the most important features to forecast customer flows: weather appeared relatively unimportant, perhaps a feature of the stores' location.

Second, we demonstrate the benefits of complex models and data pooling in forecasting many time series. For the many time series forecasting problem, the data is often lacking at the individual time series level but is generally rich after pooling the data together. If each time series is modeled individually, simple models tend to perform better and are more robust because complex models can easily over-fit. When pooling time series together, simple methods may well fail to capture the complex patterns in the complete database leading to under-fitting, so complex methods are able to deliver improved accuracy. Our results show that the linear model with Lasso regression cannot beat the time series model under data pooling, but the complex tree methods (both GBRT & RF) perform very well. Overall, the improvements in accuracy demonstrated in Tables 3 and 5 are substantial over the naïve forecasts (probably those most used in practice) – and there are consistent gains from pooling across time series.

The third contribution we make is on the methodological side, where we develop a forecasting solution based on a newly developed GBRT algorithm named XGboost for forecasting customer flows. Customer flows aggregated from mobile payments are characterized by highly volatility and skewness (as we see in section 4), multiple seasonal cycles, and high heterogeneity typical of 'big data'; these factors make customer flow forecasting a difficult problem for forecasters. Using a sample of 2000 stores and with two extensive out-of-sample test sets with a range of methods evaluated on three error measures, we find that when using the right forecasting strategy, the proposed solution performs well for both one-step and multi-step ahead forecasting tasks, compared to traditional time series models, pooled regression, and the popular Random Forest. It is worth mentioning that our forecasting solution is not limited to the customer flow forecasting problem focused on in this paper. It could be applied to many forecasting problems where a large number of disparate time series must be forecast.

Fourth, by comparing the performance of various forecasting strategies for generating multi-step ahead forecasts, we find that the Recursive strategy performs well over a stationary period, especially when forecasting for short horizons, but it fails to provide robust forecasts when there are breaks in the forecasting period. The Direct strategy is in general a more robust method compared with Recursive and it also works well for short horizons. The proposed pseudo MIMO is also a robust strategy for multi-step ahead forecasting; it provides the most accurate forecasts on almost all the test sets and metrics beating the other two strategies. But it does not work as well as Direct or Recursive over short horizons, therefore further improvements could be achieved by combining the forecasts from different strategies over different horizons.

An overarching theme of the analysis has been to demonstrate the potential of ML methods in a realistic context. Our results offer unambiguous evidence that in particular problem contexts adopting such complex methods as GBRT can lead to valuable gains in accuracy.

Further research could apply our approach to many practical forecasting scenarios, e.g., retail demand at item level (Huang et al., 2014; Ma et al., 2016), and evaluating its forecasting performance over different data sets that share demand features in common. Also, we call for researches using more newly developed complex models, e.g., deep neural networks (Krizhevsky et al., 2012) and deep forests (Zhou and Feng, 2017), to explore the benefit of complex models in forecasting many time series. One limitation of this research is our multi-step ahead forecasting experimental results are all based on the features selected and parameters tuned for one-step ahead forecasting, otherwise too much computing time would be required for such a big data forecasting problem (especially for the Direct strategy). We expect that further research could overcome the computing capability limitations to investigate the value of feature selection and parameter tuning specially designed for different multi-step ahead strategies.

Acknowledgments The authors acknowledge the help from the anonymous referees which has led to further clarification of the results. The first author acknowledges the support of the National Natural Science Foundation of China under grant no. 71571089.

Appendix A. Missing values imputing algorithm for customer flow time series

Algorithm A1.

```
Input: X (a customer flow time series with missing value), K (number of nearest neighbor) and w (a penalty parameter)
Output: Y (the imputed time series)
    Define M as the set of missing values, O as the set of observations in the X;
    For each element $i$ in the M, do
        Calculate $D_i$, which is a vector of distances defined as the date difference between $M_i$ and all the elements in the O;
        Calculate $S_i$, which is a vector of boolean indicating whether the date of each element in the O is in the same day of week with that of $M_i$;
        Let $P_i=1/(D_i+(1-S_i)*w)$, and then normalize $P_i$ to be a probability vector (w is a penalty parameter to be set by user, the larger w is set the more likely the observation on the same day of week is sampled. We set w=7 in the experiments);
        Sampling K observations randomly from O with probability $P_i$, impute $M_i$ with the mean of K observations (K is set as 10 in the experiments).
    End of the loop
    Return the imputed time series.
```

Appendix **B**. Cross-validation plot for Lasso

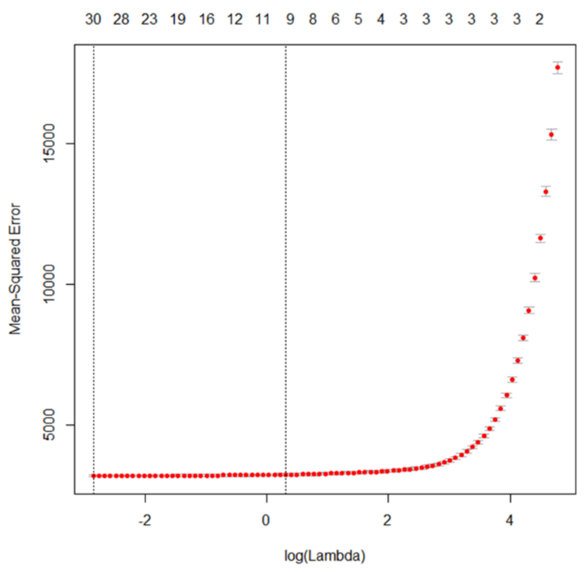


Figure B1 Cross-validation plot for Lasso. It includes the cross-validation curve (red dotted line), and upper and lower standard deviation curves along the regularization parameter Lambda sequence (error bars). Two selected Lambda's are indicated by the vertical dotted lines. One is the value that gives

minimum mean cross-validated error. The other gives the most regularized model such that error is within one standard error of the minimum. We use the first value of Lambda in the experiments.

References

Abrishami, S., Kumar, P., & Nienaber, W. (2017). Smart Stores: A Scalable Foot Traffic Collection and Prediction System. In: Perner P. (eds) *Advances in Data Mining Applications and Theoretical Aspects*. ICDM 2017. Lecture Notes in Computer Science, vol 10357. Cham: Springer, pp. 107-121.

Arunraj, N.S., & Ahrens, D. (2015). A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. *International Journal of Production Economics, 170,* 321-335.

Assimakopoulos, V., & Nikolopoulos, K. (2000). The theta model: a decomposition approach to forecasting. *International Journal of Forecasting, 16,* 521-530.

Au, K.-F., Choi, T.-M., Yu, Y. (2008). Fashion retail forecasting by evolutionary neural networks. *International Journal of Production Economics, 114*, 615-630.

Aye, G.C., Balcilar, M., Gupta, R., & Majumdar, A. (2015). Forecasting aggregate retail sales: The case of South Africa. *International Journal of Production Economics, 160,* 66-79.

Bao, Y., Xiong, T., Hu, Z. (2014). Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing, 129,* 482-493.

Bandara, K., Shi, P., Bergmeir, C., Hewamalage, H., Tran, Q., & Seaman, B. (2019). Sales demand forecast in e-commerce using a long short-term memory neural network methodology. arXiv preprint arXiv:1901.04028.

Ben Taieb, S., Sorjamaa, A., & Bontempi, G. (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing, 73,* 1950-1957.

Birattari, M., Bontempi, G., & Bersini, H. (1999). Lazy Learning Meets the Recursive Least Squares Algorithm. *Advances in Neural Information Processing Systems, 11,* 375-381.

Bontempi, G., & Taieb, S.B. (2011). Conditionally dependent strategies for multiple-step-ahead prediction in local learning. *International Journal of Forecasting, 27,* 689-699.

Breiman, L. (2001). Random Forests. *Machine Learning, 45,* 5-32.

Chapados, N., Joliveau, M., L'Ecuyer, P., & Rousseau, L.-M. (2014). Retail store scheduling for profit. *European Journal of Operational Research, 239,* 609-624.

Chen, H., & Boylan, J.E. (2008). Empirical evidence on individual, group and shrinkage seasonal indices. *International Journal of Forecasting, 24,* 525-534.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System, *Proceedings of the 22nd*

ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, California, USA:ACM, pp. 785-794.

Chuang, H.C., Oliva, R., & Perdikaki, O. (2016). Traffic ‐ Based Labor Planning in Retail Stores. *Production & Operations Management, 25,* 96-113.

Cortez, P., Matos, L.M., Pereira, P.J., Santos, N., & Duque, D. (2016). Forecasting Store Foot Traffic Using Facial Recognition, Time Series and Support Vector Machines, In: Graña M., López-Guede J., Etxaniz O., Herrero Á., Quintián H., Corchado E. (eds), International Joint Conference SOCO'16-CISIS'16-ICEUTE'16. SOCO 2016, ICEUTE 2016, CISIS 2016 (vol 527). Cham:Springer, pp. 267-276.

Crone, S. F., Hibon, M., & Nikolopoulos, K. (2011). Advances in forecasting with neural networks? Empirical evidence from the nn3 competition on time series prediction. *International Journal of Forecasting, 27*, 635-660.

Davydenko, A., & Fildes, R. (2013). Measuring forecasting accuracy: The case of judgmental adjustments to SKU-level demand forecasts. *International Journal of Forecasting, 29,* 510-522.

Dekker, M., van Donselaar, K., & Ouwehand, P. (2004). How to use aggregation and combined forecasting to improve seasonal demand forecasts. *International Journal of Production Economics, 90,* 151-167.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research ,7*, 1-30.

Duncan, G.T., Gorr, W.L., & Szczypula, J. (2001). Forecasting Analogous Time Series. In J. S. Armstrong (Ed.), *Principles of Forecasting: A Handbook for Researchers and Practitioners*. Norwell, MA.: Kluwer, pp. 195-213.

Fan, C., Liu, D., Huang, R., Chen, Z., & Deng, L. (2016). PredRSA: a gradient boosted regression trees approach for predicting protein solvent accessibility. *BMC Bioinformatics, 17,* S8.

Fildes, R., & Petropoulos, F. (2015). Simple versus complex selection rules for forecasting many time series. *Journal of Business Research, 68,* 1692-1701.

Frees, E.W., & Miller, T. W. (2004). Sales forecasting using longitudinal data models. International *Journal of Forecasting, 20,* 99-114.

Friedman, J.H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics, 29,* 1189-1232.

Gür Ali, Ö., Sayin, S., van Woensel, T., & Fransoo, J. (2009). SKU demand forecasting in the presence of promotions. *Expert Systems with Applications, 36,* 12340-12348.

Gür Ali, Ö., & Yaman, K. (2013). Selecting rows and columns for training support vector regression models with large retail datasets. *European Journal of Operational Research, 226,* 471-480.

Green, K.C., & Armstrong, J.S. (2015). Simple versus complex forecasting: The evidence. *Journal of*

Business Research, 68, 1678-1685.

Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications, 39,* 3659-3667.

Hamza, Ebi, C., Akay, D., Kutay, F. (2009). Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert Systems with Applications, 36,* 3839-3844.

Hartmann, C., Hahmann, M., Lehner, W., & Rosenthal, F. (2015). Exploiting big data in time series forecasting: A cross-sectional approach, *IEEE International Conference on Data Science and Advanced Analytics* (DSAA). Paris: IEEE, pp. 1-10.

Huang, T., Fildes, R., & Soopramanien, D. (2014). The value of competitive information in forecasting FMCG retail product sales and the variable selection problem. *European Journal of Operational Research, 237,* 738-748.

Hyndman, R.J., & Khandakar, Y. (2008). Automatic Time Series Forecasting: The forecast Package for R. *Journal of Statistical Software, 27,* 1-22.

Kline, D. (2004). Methods for multi-step time series forecasting with neural networks. *Neural Networks in Business Forecasting*. Hershey: IGI Global, pp. 226-250.

Koning, A. J., Franses, P. H., Hibon, M., & Stekler, H. (2005). The M3 competition: Statistical tests of the results. *International Journal of Forecasting, 21*, 397–409.

Kourentzes, N.(2019). tsutils: Time Series Exploration, Modelling and Forecasting. R package version 0.9.1. URL https://github.com/trnnick/tsutils/

Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks, *International Conference on Neural Information Processing Systems*, Lake Tahoe, Nevada, USA , pp. 1097-1105.

Lago, J., De Ridder, F., & De Schutter, B. (2018). Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms. *Applied Energy, 221,* 386-405.

Lam, S., Vandenbosch, M., Pearce, M. (1998). Retail sales force scheduling based on store traffic forecasting. *Journal of Retailing, 74,* 61-88.

Li, X., Peng, L., Hu, Y., Shao, J., & Chi, T. (2016). Deep learning architecture for air quality predictions. *Environmental Science & Pollution Research, 23,* 1-10.

Ma, S., Fildes, R., & Huang, T. (2016). Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra- and inter-category promotional information. *European Journal of Operational Research, 249,* 245-257.

Makridakis, S., Hibon, M.(2000). The M3-Competition: results, conclusions and implications. *International Journal of Forecasting, 16*, 451-476.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018a). Statistical and machine learning forecasting methods. *PLOS One, 13*, 1-26.

Makridakis, S., Spiliotis, E., Assimakopoulos, V. (2018b). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting, 34*, 802-808.

Makridakis, S., Spiliotis, E. Assimakopoulos, V., (2019). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, *36*, 54-75.

Mou, S., Robb, D.J., & DeHoratius, N. (2017). Retail store operations: Literature review and research directions. *European Journal of Operational Research, 265,* 399-422.

Sorjamaa, A., Hao, J., Reyhani, N., Ji, Y., & Lendasse, A. (2007). Methodology for long-term prediction of time series. *Neurocomputing, 70,* 2861-2869.

Sorjamaa, A., Lendasse, A. (2006). Time Series Prediction using DirRec Strategy. *European Symposium on Artificial Neural Networks*. Bruges, Belgium, pp. 143-148.

Taieb, S.B., Bontempi, G., Atiya, A.F., & Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications, 39,* 7067-7083.

Tibshirani, R. (2011). Regression shrinkage and selection via the LASSO: A retrospective. *Journal of the Royal Statistical Society, 73,* 273-282.

Valiant, L.G. (1984). A theory of the learnable. *Communications of the Acm, 27,* 1134-1142.

van Donselaar, K.H., Peters, J., de Jong, A., & Broekmeulen, R.A.C.M. (2016). Analysis and forecasting of demand during promotions for perishable items. *International Journal of Production Economics, 172,* 65-75.

Veeling, B. (2014). Improving Visitor Traffic Forecasting in Brick-and-Mortar Retail Stores with Neural Networks, *21th Twente Student Conference on IT*, Enschede, The Netherlands.

Walters, R.G., Mackenzie, S.B. (1988). A Structural Equations Analysis of the Impact of Price Promotions on Store Performance. *Journal of Marketing Research, 25,* 51-63.

Walters, R.G., Rinne, H.J. (1986). An empirical investigation into the impact of price promotions on retail store performance. *Journal of Retailing, 62,* 237-266.

Weigend, A.S., Hubermann, B.A., Rumelhart, D.E. (1992). Predicting sunspots and exchange rates with connectionist networks, in: Casdagli, M., Eubank, S. (Eds.), *Nonlinear Modeling & Forecasting*. Addison-Wesley, Santa Fe, NM, USA, pp. 395-432.

Williams, B.D., Waller, M.A., Ahire, S., & Ferrier, G.D. (2014). Predicting retailer orders with POS and order data: The inventory balance effect. *European Journal of Operational Research, 232,* 593-600.

Wong, W.K., Guo, Z.X. (2010). A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm. *International Journal of Production Economics, 128*, 614-624.

Wright, M.N., Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. *Journal of Statistical Software, 77,*1-71.

Xia, Y., Liu, C., Li, Y., & Liu, N. (2017). A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications, 78,* 225-241.

Xiong, T., Bao, Y., &Hu, Z. (2014). Interval forecasting of electricity demand: A novel bivariate EMD-based support vector regression modeling framework. *International Journal of Electrical Power & Energy Systems, 63,* 353-362.

Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., Gong, P., Ye, J., & Li, Z. (2018). Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction. *arXiv:1802.08714*.

Zhang, Y., & Haghani, A. (2015). A gradient boosting method to improve travel time prediction. *Transportation Research Part C Emerging Technologies,58,* 308-324.

Zhou, Z.H., Feng, J. (2017). Deep Forest: Towards An Alternative to Deep Neural Networks. *arXiv:1702.08835v2*.

Zięba, M., Tomczak, S.K., & Tomczak, J.M. (2016). Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction. *Expert Systems with Applications, 58,* 93-101.

Ziegler, A., & König, I.R. (2014). Mining data with random forests: current options for real-world applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 4,* 55-63.

Zotteri, G., Kalchschmidt, M. (2007). A model for selecting the appropriate level of aggregation in forecasting processes. *International Journal of Production Economics, 108*, 74-83.

Zotteri, G., Kalchschmidt, M., Caniato, F. (2005). The impact of aggregation level on forecasting performance. *International Journal of Production Economics, 93-94*, 479-491.