Knowledge and Information Systems (2019) 61:905–929
https://doi.org/10.1007/s10115-018-1302-y

REGULAR PAPER

Joint prediction of time series data in inventory management

Qifeng Zhou[1] · Ruyuan Han[1] · Tao Li[2] · Bin Xia[3]

Received: 24 April 2017 / Revised: 19 October 2018 / Accepted: 24 November 2018 /
Published online: 1 January 2019

Abstract
The problem of time series prediction has been well explored in the community of data mining. However, little research attention has been paid to the case of predicting the movement of a collection of related time series data. In this work, we study the problem of simultaneously predicting multiple time series data using joint predictive models. We observe that in real-world applications, strong relationships between different time-sensitive variables are often held, either explicitly predefined or implicitly covered in nature of the application. Such relationships indicate that the prediction on the trajectory of one given time series could be improved by incorporating the properties of other related time series data into predictive models. The key challenge is to capture the temporal dynamics of these relationships to jointly predict multiple time series. In this research, we propose a predictive model for multiple time series forecasting and apply it to the domain of inventory management. The relationships among multiple time series are modeled as a class of constraints, and in turn, refine the predictions on the corresponding time series. Experimental results on real-world data reveal that the proposed algorithms outperform well-established methods of time series forecasting.

Keywords Time series · Joint prediction · Inventory management

1 Introduction

Inventory management is the general process of efficiently and effectively monitoring the fluctuant flow of units into and out of an existing inventory [1]. This process usually involves two types of operations: (1) Transferring units into the inventory in order to guarantee the fluency of sales (a.k.a, stock in); and (2) Delivery of cargo from the inventory for sales (a.k.a, stock out). These two operations often create two types of time series data, each of which represents the amount of the corresponding operation evolving over time. Complete inventory management also seeks to control the costs associated with each operation and maintaining

✉ Qifeng Zhou
zhouqf@xmu.edu.cn

1 Department of Automation, Xiamen University, Xiamen 361005, China

2 School of Computing and Information Sciences, Florida International University, Miami, FL 33199, USA

3 School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

the status of the inventory. For simplicity, in this work, we mainly focus on the problem of forecasting the two types of time series data in inventory management.

In the existing inventory management system [2,3], inventory forecasting is often separate forecasting the amount of stock in and stock out. It is treated as a single time series forecasting which ignores the relationship between stock in and stock out. We all know that single time series forecasting has been extensively researched and there exist some outstanding methods from the statistics and data mining arenas [4–8].

In practice, the amounts of stock in and stock out in an inventory are related to each other. The amount of stock out (S_{out} for short) is usually restricted to the amount of stock in (S_{in} for short) at identical or close time periods, i.e., S_{out} is often less than S_{in} to prevent the situation that a unit is out of stock. Moreover, the scheduled S_{in} would primarily depend on the past S_{out} to avoid the situation that a unit is in excess of demand [9]. Then the two time series of S_{in} and S_{out} will bear some interdependencies according to the characteristics of inventory management. But the above single time series forecasting methods cannot capture and model the dynamics of relationships among multiple time series and to predict their future values simultaneously. And little research attention has been paid to the case of forecasting the movement of a collection of related time series data.

In this work, we propose a joint prediction model, as shown in Fig. 1, integrating the interdependencies of multiple time series into the process of forecasting. Different from the existing methods, our proposed model can capture the dynamics of relationships in multiple time series and predict their future values simultaneously. Specifically, in the domain of inventory management, the aggregated amount of S_{in} would be often larger than the aggregated amount of S_{out} in a specific period to avoid "out of stock". In addition, S_{in} and S_{out} should be close to each other to prevent a unit from being "in excess of demand". Based on such an intuition, we transform the requirement of inventory management into constraints and perform time series prediction under these constraints. And we apply the Joint prediction model on two real-world inventory datasets. Experimental results reveal that the proposed models outperform the traditional established time series forecasting models.

The paper is organized as follows. In Sect. 2, we summarize the related work. In Sect. 3, we briefly review the single time series prediction and the multiple time series predictions without the constraint. In Sect. 4, we propose the Joint predictive model and implement approaches in two cases: linear prediction model and nonlinear prediction model. In Sect. 5, we present experimental evaluations and analysis on real-world data. Finally, we conclude in Sect. 6.

2 Related work

There are a lot of real-world examples of multiple time series data that run in parallel. An illustrative one is that in cloud computing, the workload on virtual machines (e.g., CPU usage, memory usage, etc.) consists of multiple time-sensitive variables. In such a case, system administrators need the capabilities of characterizing and predicting the workload. To resolve this issue, [10] proposes a co-clustering technique to identify groups of virtual machines that frequently exhibit correlated workload patterns and the time periods in which these virtual machines groups are active. Other examples include the inventory management (e.g., in storage and delivery of storage), financial forecasting (e.g., investments and investment returns). However, despite the great demand on predicting multiple time series data in practice, little fundamental research attention has been paid to this problem.

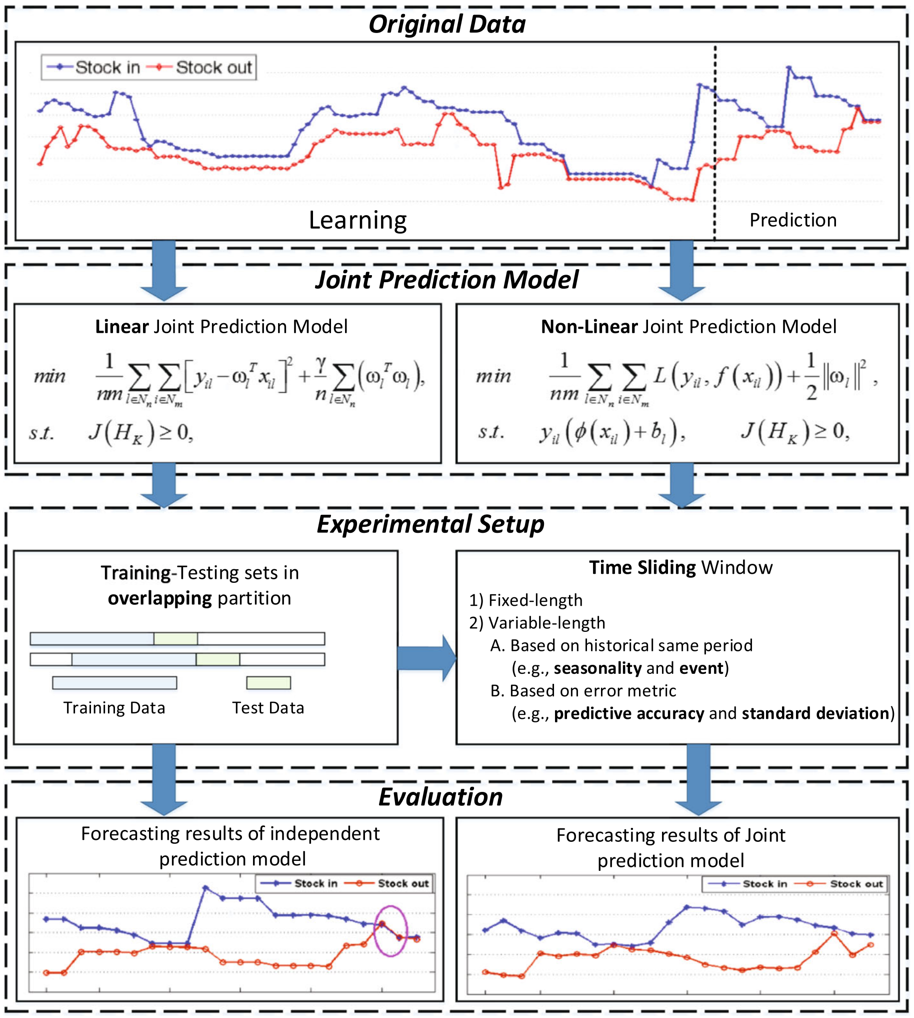

Original Data
Stock in
Stock out
Learning
Prediction
Joint Prediction Model
Linear Joint Prediction Model
$min \quad \frac{1}{nm}\sum_{l\in N_n}\sum_{i\in N_m}\left[y_{il}-\omega_l^T x_{il}\right]^2+\frac{\gamma}{n}\sum_{l\in N_n}\left(\omega_l^T\omega_l\right),$
$s.t. \quad J(H_K)\geq 0,$
Non-Linear Joint Prediction Model
$min \quad \frac{1}{nm}\sum_{l\in N_n}\sum_{i\in N_m}L\left(y_{il},f(x_{il})\right)+\frac{1}{2}\|\omega_l\|^2,$
$s.t. \quad y_{il}\left(\phi(x_{il})+b_l\right), \quad J(H_K)\geq 0,$
Experimental Setup
Training-Testing sets in overlapping partition
Training Data
Test Data
Time Sliding Window
1) Fixed-length
2) Variable-length
A. Based on historical same period (e.g., seasonality and event)
B. Based on error metric (e.g., predictive accuracy and standard deviation)
Evaluation
Forecasting results of independent prediction model
Forecasting results of Joint prediction model
Stock in
Stock out

Fig. 1 The framework of joint prediction for inventory management

Most of the time series analysis methods are drawn from statistics or machine learning fields. Classical statistical analysis models such as vector auto regression (VAR) model, vector error correction model (VECM), vector autoregressive moving average (VARMA) and multiple spectral analysis models are adopted for multiple time series analysis and forecasting. VAR model is the generalization of auto regression model, which can capture the linear interdependencies among multiple time series without requiring some strong restrictions [11,12]. VECM can be viewed as a VAR with cointegration constraints, in which an error correction term measuring the previous period's deviation from long-run equilibrium is included [13]. VECM is suitable for the time series with non-stationary cointegration constraints. VARMA models are of interest as generalizations of successful univariate ARMA models. The reduction in the number of parameters involved in VARMA models is an important consideration. For example, vector moving average (VMA) is a simplified version, and we can only use relatively simple and effective tools to determine the order of a VMA model [14]. Multiple spectral analysis in statistic and signal processing is also called frequency domain analysis, that estimates the strength of different frequency components of time series signal [15,16], a typical work is as singular-value decomposition of a specific matrix constructed upon time series [17]. These multivariate statistical analysis models are simple, easy to operate and understand and have been successfully used in economics, physiology, electric circuits, and other fields.

In recent years, multiple time series forecasting problems also attract more attention in machine learning field. Researchers expect to adopt machine learning and data mining techniques to deal with the enormous data emerged in practice and revealing the potential relationships among multiple time series which are new challenges for statistical models. Clustering on multiple time series data might be valuable for finding patterns within time series. In [18], an evolving clustering method is employed to extract profiles of relationships among multiple climate time series, and then a non-parametric regression analysis method is used to generate the predictions based on the profiles. Finazzi et al. [19] proposes a novel clustering approach based on a modification of classic state-space modeling, which is applied to lake surface water temperature for 256 lakes globally for 5 years of data to obtain investigate information. Pravilovic et al. [20] proposes a spatio-temporal clustering-based multiple geophysical time series forecasting method. It includes two steps, partitioning time series into different clusters, and then predicting the future values of a time series by utilizing its historical values and information from other cluster-time series. Neural network models, also called artificial neural networks(ANN), have been widely used for time series forecasts [21,22]. Inspired by the study of brain architecture, ANN is constructed on a collection of connected units or nodes called artificial neurons. Each connection between artificial neurons can transmit a signal from one to another. And the artificial neuron that receives the signal can process it, thus learning from data. ANN has been applied in many areas where statistical methods are traditionally employed. ANN without the strong assumption about the learning function, but it often trapped in local minima.

The problem of joint prediction on multiple time series data has not been fundamentally studied in the research community. The key challenge of this problem is that it is difficult to characterize the constraints of different time series. In the following, we highlight the research problems closely related to our work and show their difference.

- *Multivariate time series* Multivariate or multiple time series problems mean that a problem involves more than one time series, and we need to analyze all these time series jointly. There are some models to deal with these problems, for example, a vector autoregression of order 1 (VAR(1)) on a bivariate system is:

$$\begin{aligned} y_{1,t} &= \varphi_{0,1} + \varphi_{1,1} y_{1,t-1} + \varphi_{1,2} y_{2,t-1} + \epsilon_{1,t}, \\ y_{2,t} &= \varphi_{0,2} + \varphi_{2,1} y_{1,t-1} + \varphi_{2,2} y_{2,t-1} + \epsilon_{2,t}. \end{aligned} \tag{1}$$

  which can be rewritten as follows:

$$Y_t = \Phi_0 + \Phi_1 Y_{t-1} + \epsilon_t. \tag{2}$$

- *Multi-task learning* On the other hand, joint prediction on multiple time series data can be regarded as a special case of Multi-task learning (MTL), where each time series prediction is a single prediction task. Multi-task learning is a well-explored area in machine learning community that learns a problem together with other related problem at the same time, using a shared representation over multiple problems. A myriad of research work has been conducted along this direction [23–26]. Recently, some researchers start to investigate the problem of multi-task learning with constraints. The constraints are generally stemmed from the background knowledge that defines the natural restriction among different tasks. Following this stream of research, [27] presents a general class of tractable constraints among multiple tasks, and introduces computationally efficient ways to solve the problem. Further [28], exploits a semi-supervised approach in which a potentially large set of unlabeled examples is used to enforce the constraints on a large region of the input space. Zhang [29] proposes an active learning framework exploiting the relations among tasks constrained by task outputs. Fiot and Dinuzzo [30,31] explores the application of kernel-based multi-task learning techniques to forecast the demand for electricity in multiple nodes of a distribution network. Unlike the most MTL models that use domain knowledge on the task relations to construct the learning models, Han and Zhang [32] give a formal definition of task tree structure for MTL and proposed a Task Tree model to learn the underlying tree structure among tasks and model parameters simultaneously.

– *multi-target regression* The other similar research problem is the multi-target regression, which is also known as multi-output regression. Multi-target regression aims to predict multiple continuous target variables simultaneously based on the same set of input variables. Common methods for predicting several numeric target variables at once are derived from the predictive clustering tree (e.g., multi-target regression tree) and ensemble model [33]. Moreover, Multi-target regression has been deployed with the sliding window model to handle data in a streaming fashion, such as [34] utilize sliding window model to transform event forecasting task into a predictive clustering problem.

Compared with joint prediction, multi-target regression and most of the MTL work assumes the input spaces share common representation, e.g., the feature space of these targets are unique. However, such an assumption might not hold in some practical issue.

3 Preliminaries

We formally introduce the single time series problem and multiple time series problem in this section, and then define and model the joint prediction problem in the next section. Some important notations mentioned in this paper are summarized in Table 1.

3.1 Single time series prediction

Given a single time series T, we are able to extract a sequence of samples from T. Assume there are m examples $\{(x_i, y_i) : i \in N_m\} \subset X \times Y (N_m = 1, \dots, m)$ sampled $i.i.d$ from T with an unknown probability distribution P on $X \times Y$. The input space X and the output space Y fall in a real-value space R, where X is typically a subset of R^d, i.e., the d-dimensional Euclidean space, and Y is a subset of R [35]. In the problem setup of time series forecasting, Y can be chosen as a set of real values in the time series T.

The goal of single time series prediction is to learn a function f with small expected error $E[L(y, f(x))]$, where L is a predefined loss function, and the expectation is achieved by calculating the loss with respect to the unknown probability P [36]. To obtain such a function f, a common approach that considers the expected error and regularization [37] is to learn f to minimize the function

$$\frac{1}{m} \sum_{i \in N_m} L(y_i, f(x_i)) + \gamma \|f\|_K^F, \tag{3}$$

Table 1 Important notations

Notation	Description
x_i	The i-th input sample
y_i	The i-th output sample
x_{il}	The i-th input sample in l-th time series
y_{il}	The i-th output sample in l-th time series
L	The loss function
$f(x)$	The forecasting function
m	The number of samples
n	The number of time series
N_m	The sample set
N_n	The set of time series
$\mathrm{J}(H_k)$	The constraints among multiple time series
τ	The size of unit time window
S_{in}	The amount of stock in
S_{out}	The amount of stock out

where $\|f\|_K^F$ is the F-norm of f in the function space H_K, and γ is called the regularization parameter that controls the tradeoff between the expected error made on m examples (i.e., the training error) and the complexity of the model (i.e., the smoothness). In the case that H_K consists of linear function $f(x) = w^T x$, and the regularizer is the l-2 norm, the function becomes

$$\frac{1}{m} \sum_{i \in N_m} L(y_i, w^T x_i) + \gamma w^T w, \tag{4}$$

where w is a $d \times 1$ matrix.

The form described in Eq. (3) has been adopted by a myriad of learning models. Based on the choice of the loss function L. For example, square loss $L = (y - f(x))^2$ is one such function that is well-suited for regression problems and has been greatly utilized in a lot of regression application. However, it suffers the problem that outliers in the data are punished very heavily by the squaring of the error. An alternative of such function is absolute loss $L = |y - f(x)|$, which is applicable to regression problems just like the square loss, and it avoids the problem of weighting outliers too strongly by scaling the loss only linearly instead of quadratically by the error amount. Hinge loss $\max(0, 1 - y \cdot f(x))$ [38], as a third example, works well for its purposes in SVM as a classifier, since the more a data point violates the margin, the higher the penalty is. However, hinge loss is not well-suited for regression-based problems as a result of its one-sided error.

3.2 Multiple time series prediction

In multiple time series prediction, we have n different time series data, and for the l-th time series, we have m available examples $\{(x_{il}, y_{il}) : i \in N_m\}$ sampled from the unknown distribution P_l on the space of $X_l \times Y_l$. Therefore, we have $\{(x_{il}, y_{il}) : i \in N_m, l \in N_n\}$ available data in total. The goal is to learn a function $f_l : X_l \to Y_l$ based on the available data for the l-th time series, and in total we have n such functions. Note that the input space of all n time series might share common representation, or their input space could be totally different.

Recall that in the problem of single time series prediction, our goal is to minimize the function of Eq. (3). However, in the context of multiple time series prediction, our objective needs to be changed accordingly, that is, to learn all the functions f to minimize the function

$$\frac{1}{nm}\sum_{l\in N_n}\sum_{i\in N_m} L(y_{il}, f_l(x_{il})) + \frac{\gamma}{n}\|f_l\|_{K_l}^{F}. \tag{5}$$

If we assume that the input and output spaces of all n time series are totally different, and they are independent of each other, we can have n tasks, each of which is to minimize the corresponding functions of Eq. (3). However, in practice, the output space of different time series might correlate with each other, i.e., they may have to satisfy some specific constraints naturally embedded in the applications.

4 Joint prediction

4.1 Joint prediction models

As described above, in some practical applications, the input spaces of multiple time series may be totally different, but the output spaces of all n time series correlated with each other. In this paper, we call this type of multiple time series prediction problems joint prediction. To capture the relations among different time series, we add a constraint term $J(H_K)$ in Eq. (5) thus the function becomes

$$\begin{aligned}&\frac{1}{nm}\sum_{l\in N_n}\sum_{i\in N_m} L(y_{il}, f_l(x_{il})) + \frac{\gamma}{n}\|f_l\|_{K_l}^{F},\\ &s.t. \quad J(H_K) \geq 0,\end{aligned} \tag{6}$$

where H_K is a function space, that is, all the n functions that need to be learned based on the available n time series data.

The forms of loss function L and constraint term $J(H_K)$ could be determined according to specific scenarios. The commonly used constraints may have the following forms:

I. Pairwise constraints: the constraint between any two time series,

$$0 \leq f_p(x_{ip}) - f_q(x_{iq}) \leq \xi_t, \qquad \xi_t \geq 0. \tag{7}$$

II. Upper or lower bound constraints: the constraints on each f_l,

$$f_l(x_i) \leq \mathcal{C}_i, \quad \mathcal{C}_i \geq 0. \tag{8}$$

where $\mathcal{C}_i$ is a constant.

For inventory problem, it has two time series S_{in} and S_{out}, and assuming the regularization term is l-2 norm, in such a case, Eq. (6) becomes

$$\begin{aligned}\min \quad &\frac{1}{2m}\sum_{l=1}^{2}\sum_{i\in N_m} L(y_{il}, f_l(x_{il})) + \frac{\gamma}{2}(\|f_1\|_K^2 + \|f_2\|_K^2),\\ s.t. \quad &J(f_1, f_2) \geq 0.\end{aligned} \tag{9}$$

Recall that in Sect. 1, we discussed the constraint of S_{in} and S_{out} in the domain of inventory management. To incorporate the constraint into the prediction model, the length of a time period should be defined in advance. For example, users can input the length of a time window, noted as τ, and partition both times series of S_{in} and S_{out} into multiple segments of length τ. Within several time segments t with length τ, we expect that the aggregated S_{in} (i.e, $\sum_{i\in t} f_1(x_{i1})$), is larger than but close to the aggregated S_{out} (i.e., $\sum_{i\in t} f_2(x_{i2})$). Thus, the first kind of constraints becomes,

I. The constraints between S_{in} and S_{out}:

$$0 \le \sum_{i\in t} f_1(x_{i1}) - \sum_{i\in t} f_2(x_{i2}) \le \xi_t, \qquad \xi_t \ge 0. \tag{10}$$

The ideal case is that $\sum_{i\in t} f_1(x_{i1})$ is always larger than or equal to $\sum_{i\in t} f_2(x_{i2})$ in any specific time window. However, in practice, S_{out} cannot be accurately controlled based on the history S_{out}, and hence $\sum_{i\in t} f_1(x_{i1})$ may be less than $\sum_{i\in t} f_2(x_{i2})$ in some time window t, but is not deviated much from $\sum_{i\in t} f_2(x_{i2})$. To handle this problem, we should modify the constraint described in Eq. (10) as

$$\eta_t \le \sum_{i\in t} f_1(x_{i1}) - \sum_{i\in t} f_2(x_{i2}) \le \xi_t, \quad \eta_t, \xi_t \ge 0. \tag{11}$$

II. The constraints on S_{in} and S_{out}, respectively:

$$\sum_{i\in t} f_1(x_{i1}) \le \mathcal{C}_1, \quad \sum_{i\in t} f_2(x_{i2}) \le \mathcal{C}_2, \quad \mathcal{C}_1, \mathcal{C}_2 \ge 0, \tag{12}$$

where C_1 and C_2 are predefined constants, such as storage capacity and production planning.
In some cases, inventory issues need to meet both constraints I and constraint II, i.e., meeting the following constraints:

III. The constraint both on case I and case II:

$$\begin{aligned} &\eta_t \le \sum_{i\in t} f_1(x_{i1}) - \sum_{i\in t} f_2(x_{i2}) \le \xi_t, \\ &\sum_{i\in t} f_1(x_{i1}) \le \mathcal{C}_1, \quad \sum_{i\in t} f_2(x_{i2}) \le \mathcal{C}_2, \\ &\eta_t, \xi_t, \mathcal{C}_1, \mathcal{C}_2 \ge 0. \end{aligned} \tag{13}$$

4.2 Learning with linear function

In statistics, linear regression is a commonly used approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted as χ. Therefore, linear regression $f_l(x_{il}) = \omega_l^T x_{il}$ can be adopted as a baseline method to solve the proposed model. Suppose $\| f_l \|_K^F$ is l-2 norm, $L = [y_{il} - f_l(x_{il})]^2$, then the objective function of joint prediction in Eq. (6) becomes the following Eq. (14).

$$\begin{aligned} \min \quad & \frac{1}{nm} \sum_{l\in N_n} \sum_{i\in N_m} [y_{il} - \omega_l^T x_{il}]^2 + \frac{\gamma}{n} \sum_{l\in N_n} (\omega_l^T \omega_l), \\ s.t. \quad & J(H_K) \ge 0, \end{aligned} \tag{14}$$

This function can be solved as a Quadratic Programming(QP) problem to find the optimal parameters of the model. The standard quadratic program function is given in Eq. (15) [39,40].

$$\begin{aligned} \min \quad & f(\boldsymbol{\omega}) = \frac{1}{2}\boldsymbol{\omega}^T H \boldsymbol{\omega} + c^T \boldsymbol{\omega}, \\ s.t. \quad & A\boldsymbol{\omega} \geq b, \end{aligned} \tag{15}$$

where $\boldsymbol{\omega}$ is a n-dimensional column vector of prediction model, H is a Hesse matrix (n order symmetric matrix), c is a n-dimensional column vector, A is a $m \times n$ matrix, and b is a m-dimensional column vector. In the first kind of constraints, i.e., if constraints are:

$$\begin{aligned} &\eta_{1,2} \leq \sum_{i \in N_m} w_1^T x_{i1} - \sum_{i \in N_m} w_2^T x_{i2} \leq \xi_{1,2} \\ &\eta_{1,3} \leq \sum_{i \in N_m} w_1^T x_{i1} - \sum_{i \in N_m} w_3^T x_{i3} \leq \xi_{1,3} \\ &\vdots \\ &\eta_{2,3} \leq \sum_{i \in N_m} w_2^T x_{i2} - \sum_{i \in N_m} w_3^T x_{i3} \leq \xi_{2,3} \\ &\vdots \\ &\eta_{n-1,n} \leq \sum_{i \in N_m} w_{n-1}^T x_{in-1} - \sum_{i \in N_m} w_n^T x_{\text{in}} \leq \xi_{n-1,n} \end{aligned} \tag{16}$$

then:

$$H = \begin{bmatrix} \frac{2}{nm}\sum_{i \in N_m} x_{i1}x_{i1}^T + \frac{2\gamma}{n} & & 0 \\ & \ddots & \\ 0 & & \frac{2}{nm}\sum_{i \in N_m} x_{iN_n}x_{iN_n}^T + \frac{2\gamma}{n} \end{bmatrix} \tag{17}$$

$$c^T = -\frac{2}{nm}\left[\sum_{i \in N_m} y_{i1}x_{i1}^T \cdots \sum_{i \in N_m} y_{iN_n}x_{iN_n}^T\right] \tag{18}$$

$$A = \begin{bmatrix} \sum_{i \in N_m} x_{i1}^T & -\sum_{i \in N_m} x_{i2}^T & 0 & \cdots & 0 \\ \sum_{i \in N_m} x_{i1}^T & 0 & -\sum_{i \in N_m} x_{i3}^T & \cdots & 0 \\ & & \vdots & & \\ 0 & \sum_{i \in N_m} x_{i2}^T & -\sum_{i \in N_m} x_{i3}^T & \cdots & 0 \\ & & \vdots & & \\ 0 & \cdots & 0 & \sum_{i \in N_m} x_{in-1}^T & -\sum_{i \in N_m} x_{\text{in}}^T \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -\sum_{i \in N_m} x_{i1}^T & \sum_{i \in N_m} x_{i2}^T & 0 & \cdots & 0 \\ -\sum_{i \in N_m} x_{i1}^T & 0 & \sum_{i \in N_m} x_{i3}^T & \cdots & 0 \\ & & \vdots & & \\ 0 & -\sum_{i \in N_m} x_{i2}^T & \sum_{i \in N_m} x_{i3}^T & \cdots & 0 \\ & & \vdots & & \\ 0 & \cdots & 0 & -\sum_{i \in N_m} x_{in-1}^T & \sum_{i \in N_m} x_{\text{in}}^T \end{bmatrix} \tag{19}$$

$$b = \left[\eta_{1,2}\ \eta_{1,3}\ \cdots\ \eta_{2,3}\ \cdots\ \eta_{n-1,n} \vdots -\xi_{1,2}\ -\xi_{1,3}\ \cdots\ -\xi_{2,3}\ \cdots\ -\xi_{n-1,n}\right]^T \tag{20}$$

In the second kind of constraints, i.e., if constraints are:
s.t. $\eta_t \leq \sum_{i\in N_m, l\in N_n} w_l^T x_{il} \leq \xi_t$
then

$$H = \begin{bmatrix} \frac{2}{nm}\sum_{i\in N_m} x_{i1}x_{i1}^T + \frac{2\gamma}{n} & & 0 \\ & \ddots & \\ 0 & & \frac{2}{nm}\sum_{i\in N_m} x_{iN_n}x_{iN_n}^T + \frac{2\gamma}{n} \end{bmatrix} \tag{21}$$

$$c^T = -\frac{2}{nm}\left[\sum_{i\in N_m} y_{i1}x_{i1}^T \cdots \sum_{i\in N_m} y_{iN_n}x_{iN_n}^T\right] \tag{22}$$

$$A = \begin{bmatrix} \sum_{i\in N_m} x_{i1}^T & \cdots & \sum_{i\in N_m} x_{iN_m}^T \\ -\sum_{i\in N_m} x_{i1}^T & \cdots & -\sum_{i\in N_m} x_{iN_m}^T \end{bmatrix} \tag{23}$$

$$b = \begin{bmatrix} \eta_t \\ -\xi_t \end{bmatrix} \tag{24}$$

When $n = 2$ (inventory problem) and constraint is Eq. (13) then their expressions are as follows

$$H = \begin{bmatrix} \frac{1}{m}\sum_{i\in N_m}(x_{i1}x_{i1}^T) + \gamma & \mathbf{0} \\ \mathbf{0} & \frac{1}{m}\sum_{i\in N_m}(x_{i2}x_{i2}^T) + \gamma \end{bmatrix}_{n\times n}, \tag{25}$$

$$c = \begin{bmatrix} -\frac{1}{m}\sum_{i\in N_m} y_{i1}x_{i1}^T \\ -\frac{1}{m}\sum_{i\in N_m} y_{i2}x_{i2}^T \end{bmatrix}_{n\times 1}, \tag{26}$$

$$A = \begin{bmatrix} -x_{i1}^T & x_{i2}^T \\ x_{i1}^T & -x_{i2}^T \end{bmatrix}_{m\times n}, \tag{27}$$

$$b = \begin{bmatrix} \eta_t \\ -\xi_t \end{bmatrix}_{m\times 1}. \tag{28}$$

Interior Point Methods(IPM) are a class of algorithms to solve linear and nonlinear convex optimization problems. In this work, we solved above QP problem by the Primal-dual interior point method. The theoretical time complexity of IPM is $O(n^\wedge 3)$.

4.3 Learning with nonlinear function

In this section, we will discuss adopting nonlinear function, Support Vector Regression(SVR), to implement joint prediction models. SVR is a popular regression algorithm, characterized by the usage of kernels, absence of local minima, sparseness of the solution and capacity control obtained by maximum-margin [41].

In SVR-based joint prediction model, the quality of estimation is measured by *ϵ-intensive loss function* [as shown in Eq. (29)] which can ensure the existence of the global minimum and at the same time optimization of reliable generalization bound.

$$L(y_{il}, f_l(x_{il})) = \begin{cases} 0 & \text{if } \; |y_{il} - f_l(x_{il})| \leq \epsilon \\ |y_{il} - f_l(x_{il})| - \epsilon & \text{otherwise} \end{cases} \tag{29}$$

In this case, joint prediction model becomes

$$\begin{aligned} \min \quad & \frac{1}{nm}\sum_{l\in N_n}\sum_{i\in N_m} L(y_{il}, f_l(x_{il})) + C\frac{1}{2}\|\omega_l\|^2, \\ s.t. \quad & y_{il}(\omega_l^T\phi(x_{il}) + b_l) \geq 1, \quad J(H_K) \geq 0, \end{aligned} \tag{30}$$

where C is penalty coefficient to balance the training error and the generalization ability. $\phi(\cdot)$ is a linear or nonlinear kernel function. Here, we choose nonlinear function, Radial Basis Function (RBF) [42], as kernel function. RBF kernel is defined as Eq. (31).

$$K(x, x^{'}) = \exp\left(-\frac{\|x - x^{'}\|^2}{2\sigma^2}\right), \tag{31}$$

where $\|x - x^{'}\|^2$ may be recognized as the squared Euclidean distance between the two feature vectors, σ is a parameter that sets the "spread" of the kernel.

In general, the kernel parameter σ and penalty parameter C can be determined by a grid search approach. The grid search is an exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. In this work, we select a finite set of values for each C and σ, e.g., $C \in \{10, 100, 1000\}$, $\sigma \in \{0.05, 0.1, 0.5, 1.0\}$. Then we use the grid search to train a SVR model with each pair (C, σ) and evaluate their performance on a validation set. Finally, the grid search algorithm outputs the settings that achieve the highest score in the validation procedure.

To solve SVR QP problem, we adopt commonly used Sequential Minimal Optimization(SMO) approach. SMO breaks a very large QP optimization problem into a series of smallest possible QP problems. These small QP problems are solved analytically [43]. The theoretical time complexity of SMO is $O(n^\wedge 2.1)$ less than another commonly used chunking algorithm, which is $O(n^\wedge 2.9)$.

4.4 Time series data processing

4.4.1 Partition of history data

To train a time series forecasting model, the dataset is often divided into overlapping training-validation-testing sets. This process can reduce the time-variability as well as maintain the time order. Partitioning an independent validation set is usually fit for the big size of dataset, to utilize the data more fully and further reduce the time-varying characteristic, we use k-fold cross validation technique instead of independent validation set.

Suppose the size of the whole dataset is q, and the size of each training set and testing set are $q1$ and $q2$, respectively. Then, the whole dataset will be divided into r overlapping training-testing sets,

$$r = \left\lceil \frac{q - q1}{q2} \right\rceil, \tag{32}$$

where $\lceil x \rceil$ denotes the minimal positive integer that is not less than x, and $q2$ can be viewed as the updating period for the prediction model, i.e., the prediction model established after $q2$ periods will be renewed with new data. Figure 2 shows an example of overlapping partition for two successive training-testing sets.

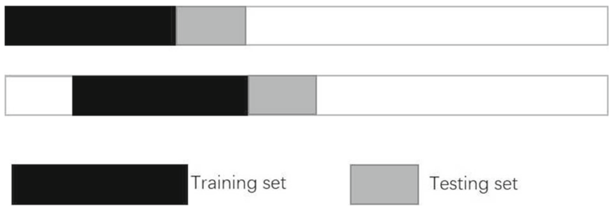


Fig. 2 An example of time series training-testing overlapping partition

4.4.2 Selecting the length of time window

Inventory management time series is always nonlinear and time-varying. Using fixed length time window to establish the prediction model cannot capture the change of the inventory over time. Therefore, how to select the length of the time window is a key step in forecasting. Set τ as a unit length of the time window, to forecast the inventory value in time $t^* \in [t, (t+e)]$, historical data set X_{t^*} can be used as modeling samples,

$$X_{t^*} = \{x_{(t-\tau)}, x_{(t-\tau+1)}, \ldots, x_{t-1}\}. \tag{33}$$

When the variable length time windows are adopted, we consider two cases: in the first case, we assume that the amount of S_{in} and S_{out} in the next period are close to the current ones. In this case, we set $\tau = \{10, 20, \ldots, 90, 99\}$ as the alternative length of the time window. Then the model is trained with different sizes of windows, and the window size which has higher predictive accuracy and lower standard deviation is selected for the current period. For the next overlapping set, we take the length of current window as the basis and select one larger and one smaller size as the alternative length. Then the model is trained on these three sizes of windows, and the window which has higher predictive accuracy and lower standard deviation is selected. This process will repeat until to the last overlapping set.

Another case is that it is reasonable to assume that S_{in} or S_{out} in the next period is close to that of the historical same period. Inventory information has typical characteristics of periodicity and seasonality, for example, S_{in} and S_{out} will increase greatly in the Spring Festival, May Day, and National Day; and in sales off-season, such as March, June, and July, S_{in} and S_{out} will decrease greatly. In this case, we select the historical same period window size as the basis and select one larger and one smaller window sizes as the alternative length. Then the model is trained on these three sizes of windows, and the one has the highest forecasting accuracy and lowest standard deviation is selected.

4.5 Model discussion

- *Temporal cross-correlation* The key challenge of joint prediction is to capture the temporal cross-correlation of multiple time series. In proposed joint prediction model, the temporal cross-correlation are embedded into the pairwise constraints on each moving window. Take inventory for example, in each predefined time window the aggregated S_{in} (i.e, $\sum_{i\in t} f_1(x_{i1})$) must be larger than but close to the aggregated S_{out} (i.e., $\sum_{i\in t} f_2(x_{i2})$). The predicted S_{in} of current period also has effect on the S_{out} of next period.
- *Online learning and incremental learning* The proposed models are suitable for online learning. Since the setting of training-testing dataset is overlapping and independent with learning methods, when the new data arriving, we only need to move the time window to

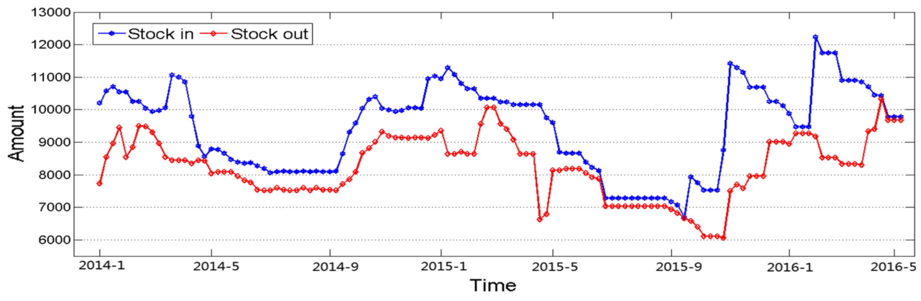


Fig. 3 Stock in and stock out time series in dataset1

the next data segment and the model is continuously updated during operation as more training data arriving. The proposed methods are also suitable for incremental learning. When the new data arriving, it only needs to update the learning models using part of old data and new data to extend the existing model's knowledge instead of retraining the models completely. For example, when we utilize SVR-based forecasting models, we can retain the support vectors (which are only a small part of samples in a total sample set) in the existing models with new data to update the learning model. If the new data are not considered as support vectors, the model will keep unchanged.

5 Experiments

To evaluate the effectiveness of proposed joint prediction framework with both linear and nonlinear models, we conduct a number of experiments on two datasets collected from real inventory transactions. We also compare the joint prediction model with some state-of-the-art time series prediction approaches, including the non-constrained independent predicting model (noted as Single), Autoregressive Integrated Moving Average (ARIMA), Vector Moving Average (VMA), Multi-target Regression (MTR), and BP-neural network method.

5.1 Dataset descriptions

Two real inventory datasets collected from different trading companies are used to conduct the experiments. The Dataset1 is provided by a real Tea E-commerce company in Jingdong Mall (One of largest self-operating E-commerce enterprise in China, JD.com,). We choose the inventory transaction data ranged from January 2014 to April 2016 including total 850 original Stock in and 850 Stock out data. Then these data are aggregated by week to generate 123 time series data S_{in} and S_{out}, respectively, as shown in Fig. 3.

The Dataset2 is collected from a large Plasma TV Sales Enterprise during September 2013 to December 2015. We also select a total of 850*2 original TV sales inventory data (850 Stock in and 850 Stock out data) and take a week as time granularity to generate 123 time series data S_{in} and S_{out}, respectively, as shown in Fig. 4.

5.2 Experimental metric

Assume $Y = \{y_1, y_2, \ldots, y_n\}$ is the observed values of a time series, and $F = \{f_1, f_2, \ldots, f_n\}$ is the estimation given by forecasting algorithm. To compare and evalu-

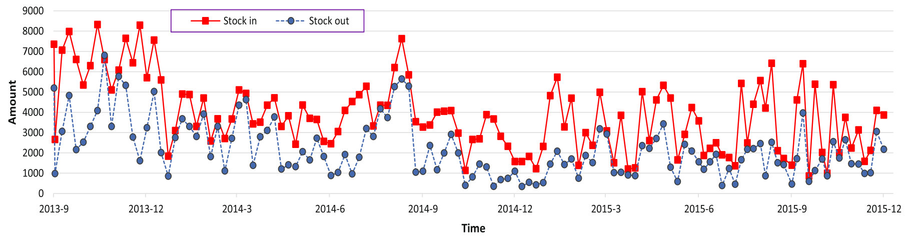


Fig. 4 Stock in and stock out time series in dataset2

ate the performance of aforementioned forecasting approaches, we adopt the following three kinds of measures [44,45].

– *Mean absolute error (MAE)* MAE is an average of the absolute errors between the prediction and the real value. It is a common measure of forecast error in time series analysis. MAE is given as follows

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|, \tag{34}$$

where f_i is the prediction and y_i is the true value.
– *Root mean square error (RMSE)* RMSE is the square root for the mean of the squares of the deviations. For an unbiased estimator, the RMSE is the square root of the variance, known as the standard error.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}|f_i - y_i|^2}{n}}. \tag{35}$$

However, MAE and MSE are scale-dependent metrics. Since the scale-dependent metrics are not conduced to the comparison between different forecasting models and multi-threshold data experiments, we usually adopt relative indicators such as SMAPE.

– *Symmetric mean absolute percentage error (SMAPE)* SMAPE is an average of ratio between the prediction and the true value.

$$\mathrm{SMAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|f_i - y_i|}{(y_i + f_i)/2}. \tag{36}$$

5.3 Experiments with linear joint prediction model

In this set of experiments, we first discuss the impact of different constraints on the predictive results, then we implement linear prediction model with two strategies: (1) learning with *fixed length time window*, and (2) learning with *variable length time window*.

To discuss the effect of different constraints on the performance of models, we adopt the fixed length time window, here we set $\tau = 30$. The experiments are carried on the linear Joint Prediction model with three kinds of constraints mentioned in Sect. 4.1, respectively, (noted as $Joint1$, $Joint2$, and $Joint3$). Table 2 shows the cumulative forecasting error under different constraints on dataset1 (note that, the minimum errors in Table 2 and the following Tables 4, 5, 6, 7, 8, 9, 10, and 11 are highlighted in bold text). Although different constraints between stock in and stock out have different effects on the performance of multiple time series

Table 2 The cumulative forecasting error of three kinds of constraints on dataset1

Evaluation index	S_{in}	S_{out}	Average
MAE			
Joint 1	12,229	8348	10,288.5
Joint 2	13,944	**6676**	10,310
Joint 3	**12,130**	8247	**10,188.5**
RMSE			
Jiont 1	16,017	9458	12,737.5
Jiont 2	17,564	9458	13,511
Jiont 3	**15,852**	9311	**12,581.5**
SMAPE			
Jiont 1	0.19	0.15	0.17
Jiont 2	0.21	**0.11**	**0.16**
Jiont 3	**0.18**	0.15	**0.16**

prediction models, their difference is not very big. Therefore, in the following experiments, we choose constraint III as the constraint relationship among multiple time series to compare the performance of different inventory forecasting models. Note that in practice, constraints should be given in advance according to specific demands.

As described in Sect. 4.4.2, to implement a linear Joint prediction model with variable length time window we proposed two strategies. Under the first strategy, we firstly set $\tau = \{10, 20, \ldots, 90, 99\}$ as the alternative length of the time window. Then, predictive accuracy and standard deviation in the current period are adopted as the criteria to determine the length of the time window in the next time period. In this process, the length of the time windows varies with time series data, and we note these model as $Joint\_dynamic1$.

In the second case, we set the lengths of time window according to the fluctuations of inventory in the historical same period, e.g., selecting a small time window from $\tau = \{10, 20 \text{ or } 30\}$ when S_{in} or S_{out} has big fluctuation on the corresponding period in last year (e.g., Spring festival, Labor day, National Day), and selecting a big time window from $\tau = \{70, 80 \text{ or } 90\}$ when S_{in} or S_{out} has small fluctuation, this model is noted as $Joint\_dynamic2$.

5.4 Experiments with nonlinear prediction model

In this set of experiments, SVR is adopted in joint prediction model to fulfill a nonlinear time series prediction, called $Joint\_SVR$. In the experiments with fixed length time window, the length of windows is selected from $\tau = \{10, 20, \ldots, 90, 99\}$. And the experiments with variable length time window are also performed under the same two strategies with the linear prediction models. These models are remarked as $Joint\_dynamic\_SVR1$ (i.e., setting the lengths of time window according to accuracy and standard error) and $Joint\_dynamic\_SVR2$ (i.e., setting the lengths of time window according to the inventory in the historical same period), respectively.

The aforementioned joint prediction models with variable length time window are compared with the fixed length time window prediction models and the single time series prediction models. In addition, in this work, some state-of-the-art time series forecasting models in statistical and machine learning fields are also compared with proposed joint prediction models, including ARIMA, VMA, MTR, and BP-neural network model. All these models are described in Table 3. Note that the original MTR model processes the training dataset such as the following form:

Table 3 Model descriptions

Model	Description
Joint_Fixed	Joint prediction model with the fixed length of time window
Joint_dynamic	Joint prediction model with the variable length of time window
Joint_dynamic1	The length of time window is determined by predictive accuracy
Joint_dynamic2	The length of time window is determined according to the fluctuations of corresponding period of last year
Joint_Fixed_SVR	Nonlinear joint model with the fixed length of time window
Joint_dynamic_SVR	Nonlinear joint model with the variable length of time window
Joint_SVR1	The length of time window is determined by predictive accuracy
Joint_SVR2	The length of time window is determined according to the fluctuations of corresponding period of last year
Single	Single prediction model without constraints between S_{in} and S_{out}
SVR_single	SVR model without constraints between S_{in} and S_{out}
ARIMA	Autoregressive integrated moving average model
VMA	Vector moving average model
MTR	Multi-target regression model
BP	BP-neural network model

$$x_1, x_2, \dots x_n \rightarrow \{y_1, y_2, \dots y_m\} \tag{37}$$

In the semantics of inventory issues, we change the Stock in and Stock out datasets into the following form to meet the MTR model:

$$x_{in,1}, x_{in,2}, \dots x_{in,n}, x_{out,1}, x_{out,2}, \dots x_{out,n} \rightarrow \{y_{\text{in}}, y_{\text{out}}\} \tag{38}$$

In these experiments, the window sizes of Joint_fixed, Single, ARIMA, VMA, MTR, and BP methods are selected from $\tau = \{10, 20, \dots, 90, 99\}$. For example, Tables 4, 5, 6, 7 present the experimental process of selecting the length of time windows in different models. We can observe that BP model needs a bigger size of window to gain good performance, this may be that BP method is more complex than linear model and ARIMA method requires more training data to construct a model with good generalization.

5.5 Results and analysis

5.5.1 Experimental results on Dataset1

Table 8 shows the experimental results on dataset1 with linear joint prediction models. For each model, we select its best predictive results under the alternative parameters. The data in Table 8 represent the average error of predicted S_{in} and S_{out} under different criteria. Each pair

Table 4 The forecasting error of Joint_fixed model under different size of windows

Joint_fixed	MAE_S_{in}	MAE_S_{out}	RMSE_S_{in}	RMSE_S_{out}	SMAPE_S_{in}	SMAPE_S_{out}
Win_10	383.6796	297.5744	702.3403	446.2667	0.0358	0.0328
Win_20	391.7472	282.6855	669.2211	428.9323	0.0364	0.0313
Win_30	383.1043	254.7431	666.6108	426.2464	0.0357	0.0281
Win_40	337.3474	252.6593	660.3014	430.2162	0.0315	0.0279
Win_50	**311.6005**	**246.4348**	**645.8475**	**423.5693**	**0.0291**	**0.0272**
Win_60	314.0946	250.9371	648.6223	425.5185	0.0293	0.0276
Win_70	321.3418	251.4372	647.4843	425.3410	0.0300	0.0277
Win_80	326.2138	249.5264	648.2067	424.0754	0.0305	0.0275
Win_90	323.6240	250.0751	647.9675	424.3760	0.0303	0.0276
Win_99	320.1518	253.3774	646.6877	426.0657	0.0299	0.0279

Table 5 The forecasting error of Single model under different size of windows

Single	MAE_S_{in}	MAE_S_{out}	RMSE_S_{in}	RMSE_S_{out}	SMAPE_S_{in}	SMAPE_S_{out}
Win_10	411.2073	318.3925	722.7793	461.1539	0.0386	0.0348
Win_20	385.5120	297.4001	683.4295	440.8310	0.0357	0.0327
Win_30	365.7634	268.8608	678.3583	438.1244	0.0338	0.0295
Win_40	327.4217	253.9615	673.0014	440.0261	0.0304	0.0279
Win_50	296.5950	238.8933	662.1404	430.3847	0.0275	0.0263
Win_60	**295.2932**	**235.9657**	**659.3663**	**429.6929**	**0.0275**	**0.0258**
Win_70	302.9679	236.4625	660.9606	430.2592	0.0282	0.0259
Win_80	310.3697	236.1715	661.5028	429.8569	0.0288	0.0260
Win_90	302.8185	235.4926	660.6595	430.0499	0.0281	0.0259
Win_99	296.0903	237.0044	659.5862	431.0336	0.0276	0.0261

Table 6 The forecasting error of ARIMA model under different length of windows

ARIMA	MAE_S_{in}	MAE_S_{out}	RMSE_S_{in}	RMSE_S_{out}	SMAPE_S_{in}	SMAPE_S_{out}
Win_10	779.9150	472.1413	993.7222	616.5589	0.0727	0.0554
Win_20	767.1857	465.7854	961.5539	607.1688	0.0719	0.0538
Win_30	762.6950	455.9603	953.6886	609.5810	0.0708	0.0506
Win_40	722.1252	425.9252	987.3089	581.3845	0.0679	0.0477
Win_50	657.0462	442.2947	950.4241	599.5640	0.0618	0.0493
Win_60	675.8487	430.0410	961.1186	587.5729	0.0628	0.0481
Win_70	688.3899	439.7919	965.1973	594.3256	0.0639	0.0491
Win_80	621.4150	442.7437	918.7579	597.2386	0.0581	0.0494
Win_90	**620.6401**	**417.2078**	**910.2400**	**583.3575**	**0.0579**	**0.0466**
Win_99	634.9884	439.2677	930.8533	591.2309	0.0593	0.0491

Table 7 The forecasting error of BP model under different length of windows

BP	MAE_S_{in}	MAE_S_{out}	RMSE_S_{in}	RMSE_S_{out}	SMAPE_S_{in}	SMAPE_S_{out}
Win_10	1084.6953	630.0341	1499.1284	264.2952	0.1045	0.0713
Win_20	1431.1342	1034.1287	2151.1385	1338.1774	0.1248	0.1223
Win_30	1157.0386	807.6845	1619.1473	1065.2492	0.1145	0.0935
Win_40	1239.9745	771.6616	1829.8122	1034.2987	0.1199	0.0903
Win_50	988.8868	740.4088	1390.4941	878.1146	0.0921	0.0838
Win_60	796.3875	795.6159	1246.9763	992.3734	0.0770	0.0917
Win_70	732.1713	636.3702	1134.2393	832.8367	0.0717	0.0723
Win_80	663.1521	572.2429	966.8686	727.6441	0.0625	0.0636
Win_90	**577.3470**	529.1933	956.1066	646.7348	0.0542	0.0607
Win_99	580.8523	**473.6979**	**890.7115**	**586.2252**	**0.0555**	**0.0532**

Table 8 The experimental results on dataset1 using linear joint prediction models

Evaluation index	Joint_dynamic1	Joint_dynamic2	Joint_fixed_win_50	Singl_win_60
MAE_Aver	277.0960	283.7128	279.0176	**266.5288**
RMSE_Aver	556.4216	549.3468	556.6372	**547.7782**
SMAPE_Aver	0.0280	0.0286	0.0282	**0.0268**

of predicted S_{in} and S_{out} is shown in Fig. 5. It can be observed from Table 8 that among all the models, the predictive error of Single time series model is the smallest, but Fig. 5 shows that in this model some predictive results of Stock in are bigger than those of Stock out. These results are not meet the practical demand for inventory management. The proposed $Joint\_Dynamic$ model has the slightly greater forecast error than the Single model, but all the results meet the constraints given in advance.

The experimental results on dataset1 with nonlinear joint prediction models are shown in Table 9 and Fig. 6. From Table 9, we can observe that under the three criteria, $Joint\_dynamic\_SVR1$ model gets the best prediction performance in all the nonlinear models. SVR_single has the slightly bigger error than $Joint\_dynamic\_SVR1$. Figure 6 shows the relationship of predicted Stock in and Stock out. We can observe that SVR_single model could not guarantee that the predictive results satisfy the practical constraints, i.e., some predicted S_{in} exceed predicted S_{out}. However, the experimental results of all the nonlinear Joint predictive models, $Joint\_dynamic\_SVR$ and $Joint\_fixed\_SVR$, satisfy the practical constraints.

The final performance comparison of all aforementioned predictive models on dataset1 is shown in Table 10. From Table 10, we can see that besides independent prediction model Single, two joint prediction models with the variable length time window and a joint prediction model with the fixed length time window all obtain better performance. Joint_dynamic1 and Joint_dynamic2 have the smaller RMSE, MTR also shows better performance (i.e., smaller MAE and REMS). The ARIMA and BP-neural network models have the biggest error. Joint_dynamic prediction models have more stable performance (i.e., small RMSE). Note that, MAE and RMSE are the most commonly used performance metrics for regression tasks. MAE is equivalent to L1 norm (first-order moment), and RMSE is equivalent to L2 norm (second-order moment), therefore RMSE is more sensitive to the anomaly. In our

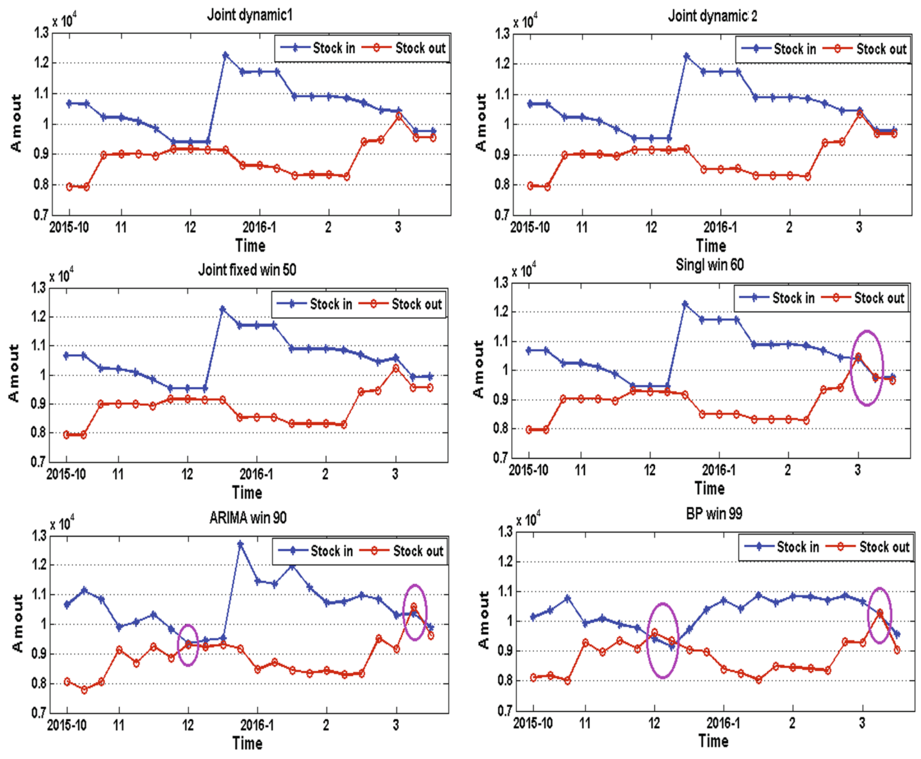


Fig. 5 Experimental results using linear joint prediction models

Table 9 The experimental results on dataset1 using nonlinear joint prediction models

Evaluation index	S_{in}	S_{out}
MAE		
Joint_dynamic_SVR1	**508.5897**	438.6564
Joint_dynamic_SVR2	521.2316	**435.6443**
Joint_fixed_SVR_win_99	583.7693	487.4188
SVR_single_win_99	521.9607	528.2754
RMSE		
Joint_dynamic_SVR1	746.6545	597.8720
Joint_dynamic_SVR2	**693.3393**	539.5030
Joint_fixed_SVR_win_99	740.8427	**524.6678**
SVR_single_win_99	728.5716	605.9881
SMAPE		
Joint_dynamic_SVR1	**0.0478**	**0.0491**
Joint_dynamic_SVR2	0.0504	0.0521
Joint_fixed_SVR_win_99	0.0520	0.0538
SVR_single_win_99	0.0495	0.0578

experiments, the best algorithm changes according to the considered metrics, which may be caused by the metrics that differ in the sensitivity of time series data.

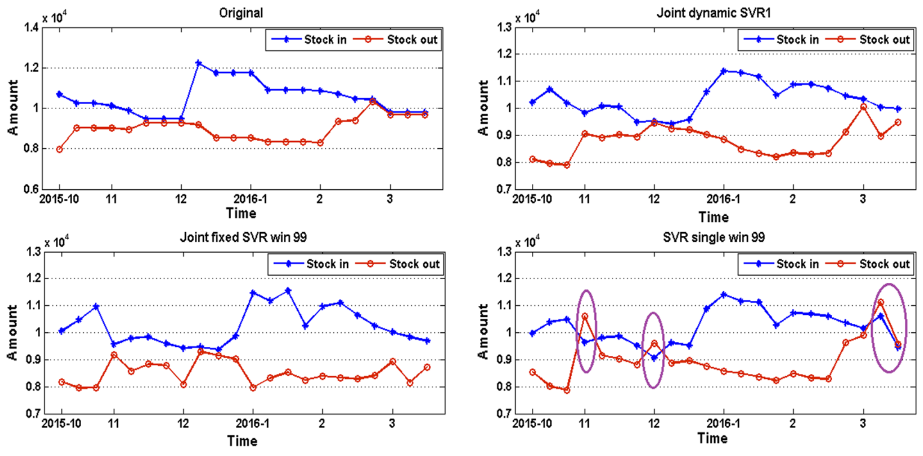


Fig. 6 Experimental results using nonlinear joint prediction models

Table 10 The average forecasting error on dataset1 using different models

Evaluation index	MAE_Aver	RMSE_Aver	SMAPE_Aver
Joint_dynamic1	277.10	**398.52**	0.0280
Joint_dynamic2	283.71	538.08	0.0286
Joint_fixed_win_50	279.02	534.71	0.0282
Joint_dynamic_SVR1	473.62	672.27	0.0485
Joint_dynamic_SVR2	478.45	616.42	0.0512
Joint_fixed_SVR_win_99	535.60	632.76	0.0529
Single_win_60	**265.63**	544.53	**0.0267**
SVR_single_win_99	525.08	667.28	0.0537
ARIMA_win_90	518.92	746.81	0.0523
VMA_win_10	526.14	686.34	0.0632
MTR_win_20	302.62	483.37	0.0309
BP_win_99	527.28	753.99	0.0543

The experimental results also show that average error rate of linear prediction models is lower than that of the nonlinear prediction models. It is possible that when the time series changing smoothly linear models have better generalization ability than nonlinear models. According to the statistical learning theory, generalization error is bounded by the training error plus VC confidence (i.e., a function of the hypothesis space size). Compared with nonlinear models, linear models have simpler structure, i.e., smaller VC confidence. These results also partly reflect the Occam's razor "the simpler one is usually better".

5.5.2 Experimental results on Dataset2

Figures 7, 8, and Table 11 show the stock forecasting results utilizing ten different methods on dataset 2. For each model, we select its best predictive results under the alternative parameters. From Figs. 7 and 8, we can see that independent prediction model (Single) and both linear and nonlinear Joint prediction models (Joint_dynamic and Joint_dynamic_SVR) have the

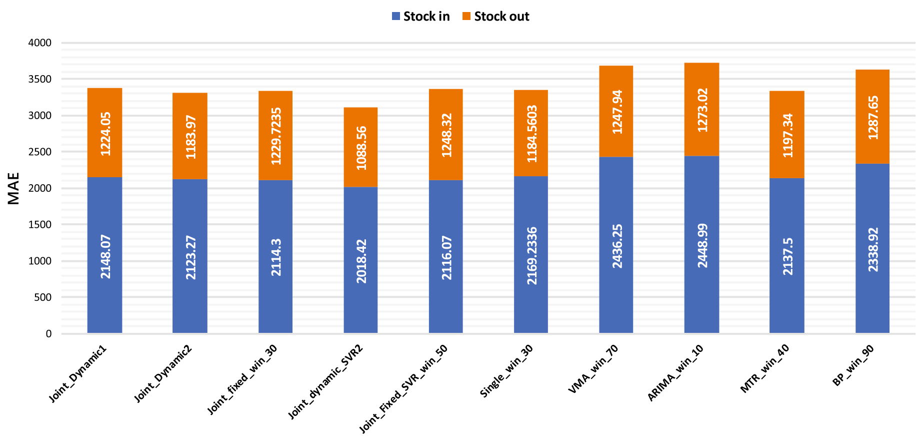


Fig. 7 Experimental results of MAE_in and MAE_out on dataset2

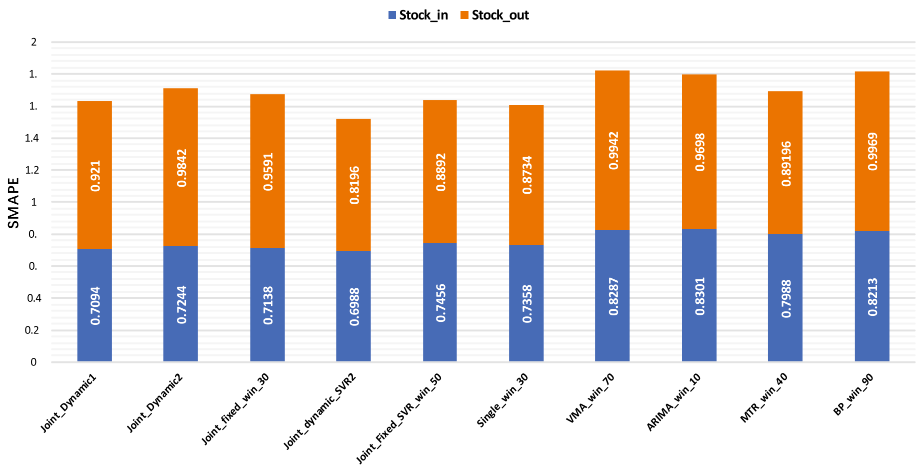


Fig. 8 Experimental results of SMAPE_in and SMAPE_out on dataset2

smaller prediction error, MTR shows better performance than ARIMA and VMA, and BP-neural network has the biggest prediction error. In addition, same as the experimental results on Dataset1, in Single time series forecasting model, ARIMA and VMA, some predictive results of Stock in are out of Stock out. These results cannot be used in the practical inventory management. Our proposed $Joint\_dynamic$ models meet the constraints given in advance. Unlike the experimental results on dataset 1, Table 11 shows that nonlinear prediction models have better performance than linear prediction models on dataset2, mainly because that the time series of dataset2 have stronger nonlinearity and time variability, and nonlinear models can capture these variations better.

In addition, the statistical analysis results of T-test between proposed joint prediction models and other compared algorithms are shown in Table 12. In Table 12, JD1 represents $Joint\_dynamic1$ model, JDS2 represents $Joint\_dynamic\_SVR2$ model, the first three columns are T-test results between $Joint\_dynamic1$ with ARIMA, VMA, and MTR on dataset1, respectively; the latter three columns are T-test results between

Table 11 The average forecasting error on dataset2 using ten different models

Evaluation index	MAE_Aver	RMSE_Aver	SMAPE_Aver
Joint_dynamic1	1686.06	2285.43	0.8152
Joint_dynamic2	1653.62	2290.11	0.8543
Joint_fixed_win_30	1672.01	2351.05	0.8365
Joint_dynamic_SVR2	**1553.49**	2270.49	0.**7592**
Joint_fixed_SVR_win_50	1682.20	**2163.62**	0.8174
Single_win_30	1676.89	2270.25	0.8046
VMA_win_70	1842.09	2281.95	0.9115
ARIMA_win_90	1861.01	2308.42	0.8999
MTR_win_40	1667.42	2594.26	0.8454
BP_win_90	1813.29	2276.66	0.9091

Table 12 The T-test results

T test	JD1_ARIMA_1	JD1_VMA_1	JD1_MTR_1	JDS1_ARIMA_2	JDS1_VMA_2	JDS1_MTR_2
S_{in}	0.047	0.008	0.087	0.026	0.040	0.109
S_{out}	0.042	0.022	0.049	0.018	0.037	0.042

$Joint\_dynamic\_SVR2$ with ARIMA, VMA, and MTR on dataset2, respectively. From Table 12, we can see that compared with ARIMA and VMA approaches, our proposed joint prediction methods have statistical significance improvement (the results of T-test are less than 0.05) on both two datasets. Compared with MTR model, stock out prediction results also have some improvement, but the T-test results of stock in on both datasets are large than 0.05.

In summary, above experimental results show that joint prediction model can capture the temporal dynamics of relationships among multiple time series data, meanwhile keeping a competitive prediction performance. Although multiple separate prediction models can achieve lower prediction error, these prediction results may not be used in practice, because single time series prediction ignores the relationship between stock in and stock out. Compared with the state-of-the-art time series forecasting approaches, both linear and nonlinear joint prediction models have better generalization for inventory forecasting, and MTR method also shows better performance. Besides, dynamic selecting the length of time window according to the transaction fluctuations over the period of history can improve the forecast accuracy.

6 Conclusion

In this article, a joint prediction framework for multiple time series prediction is proposed and applied to the domain of inventory management. Two Joint learning models: learning with linear prediction model and learning with nonlinear prediction model are studied. And three types of constraints between Stock in and Stock out are discussed. The proposed joint prediction models can predict multiple time series data simultaneously via considering the temporal dynamics of these relationships. Experiments on real inventory datasets are conducted to demonstrate the effectiveness of our proposed approaches.

In future work, we will evaluate the effectiveness of the proposed methodology in new scenarios, and implement the joint prediction models on more than two time series dataset. Moreover, we will further study the methods for adapting our model that could cope with the low latency series where samples arrive at high frequency, and analysis the sensibility of the different algorithms under the different window size.

Acknowledgements This work is partially supported by the National Natural Science Foundation of China under Grant No. 61503313, by the Natural Science Foundation of Fujian Province of China No.2017J01118. The authors would also like to thank the former members of Florida International University KDRG research team for their support and contributions to this work.

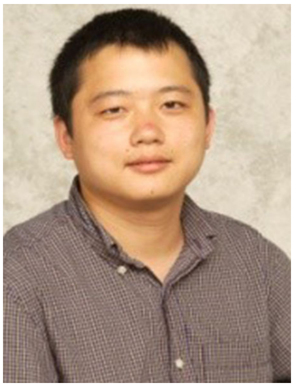
References

1. Zipkin PH (2000) Foundations of inventory management. McGraw-Hill, New York
2. Li L, Shen C, Wang L, et al (2014) iMiner: mining inventory data for intelligent management. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management. ACM, pp 2057–2059
3. Spedding TA, Chan KK (2000) Forecasting demand and inventory management using Bayesian time series. Integr Manuf Syst 11(5):331–339
4. Weigend AS (1994) Time series prediction: forescasting the future and understanding the past. Routledge, Abingdon
5. Van Gestel T, Suykens JAK, Baestaens DE et al (2001) Financial time series prediction using least squares support vector machines within the evidence framework. IEEE Trans Neural Netw 12(4):809–821
6. Taylor SJ (2007) Modelling financial time series. World Scientific, Sinagapore
7. He Z, Wang XS, Lee BS et al (2008) Mining partial periodic correlations in time series. Knowl Inf Syst 15(1):31–54
8. Aminikhanghahi S, Cook DJ (2017) A survey of methods for time series change point detection. Knowl Inf Syst 51(2):339–367
9. Scarf H (2005) The optimality of (S,s) policies in the dynamic inventory problem. Mathematical Methods in the Social Sciences, New York
10. Khan A, Yan X, Tao S, et al (2012) Workload characterization and prediction in the cloud: a multiple time series approach. In: IEEE network operations and management symposium (NOMS), pp 1287–1294
11. Banbura M, Giannone D, Reichlin L et al (2010) Large bayesian vector auto regressions. J Appl Econ 25(1):71–92
12. Sims C (1980) Macroeconomics and reality. Econometrica 48(1):1–48
13. Enders W (2010) Applied econometric time series, 3rd edn. Wiley, New York
14. Galbraith JW, Ullah A, Zinde-Walsh V (2002) Estimation of the vector moving average model by vector autoregression. Econom Rev 21(2):205–219
15. Dubman M, Goodman RN (1969) Spectral analysis of multiple time series. Wiley, New York
16. Zhang XD, Takeda H (1987) An approach to time series analysis and ARMA spectral estimation. IEEE Trans Acoust Speech Signal Process 35(9):1303–1313
17. Golyandina N, Zhigljavsky A (2013) Singular spectrum analysis for time series. Springer Science and Business Media, Berlin
18. Widiputra H, Pears R, Kasabov N (2011) Multiple time-series prediction through multiple time-series relationships profiling and clustered recurring trends. In: Advances in knowledge discovery and data mining. Springer Berlin Heidelberg, pp 161–172
19. Finazzi F, Haggarty R, Miller C et al (2015) A comparison of clustering approaches for the study of the temporal coherence of multiple time series. Stoch Environ Res Risk Assess 29(2):463–475
20. Pravilovic S, Bilancia M, Appice A et al (2017) Using multiple time series analysis for geosensor data forecasting. Inf Sci 380:31–52
21. Frank RJ, Davey N, Hunt S et al (2001) Time series prediction and neural networks. J Intell Robot Syst 31:91–103
22. Morariu N, Iancu E, Vlad S et al (2009) A neural network model for time-series forecasting. Romanian J Econ Forecast 12(4):213–223

23. Argyriou A, Evgeniou T, Pontil M (2008) Convex multi-task feature learning. Mach Learn 73(3):243–272
24. Evgeniou T, Pontil M (2004) Regularized multi–task learning. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 109–117
25. Argyriou A, Evgeniou T, Pontil M (2007) Multi-task feature learning. In: Proceedings in advances in neural information processing systems (NIPS)
26. Chandra R, Ong YS, Goh CK (2017) Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction. Neurocomputing 243:21–34
27. Lugosi G, Papaspiliopoulos O, Stoltz G (2009) Online multi-task learning with hard constraints. arXiv preprint arXiv:0902.3526
28. Maggini M, Papini T (2010) Multitask semiCsupervised learning with constraints and constraint exceptions. In: Artificial neural networks CICANN. Springer, Berlin, pp 218–227
29. Zhang Y (2010) Multi-task active learning with output constraints. AAAI, Menlo Park
30. Fiot JB, Dinuzzo F (2018) Electricity demand forecasting by multi-task learning. IEEE Trans Smart Grid 9(2):544–551
31. Fiot JB, Dinuzzo F (2015) Electricity demand forecasting by multi-task learning. Comput Sci PP(99):1–1
32. Han L, Zhang Y (2015) Learning tree structure in multi-task learning. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 397–406
33. Struyf J, Doeroski S (2005) Constraint based induction of multi-objective regression trees. In: International workshop on knowledge discovery in inductive databases. Springer, Berlin, pp 222–233
34. Pravilovic S, Appice A, Malerba D (2013) Process mining to forecast the future of running cases. International workshop on new frontiers in mining complex patterns. Springer, Cham, pp 67–81
35. Hamilton JD (1994) Time series analysis. Princeton University Press, Princeton
36. Evgeniou T, Micchelli CA, Pontil M (2005) Learning multiple tasks with kernel methods. J Mach Learn Res 6:615–637
37. Müller KR, Smola AJ, Tsch G, et al (1997) Predicting time series with support vector machines. In: Artificial neural networks ICANN'97. Springer, Berlin, pp 999–1004
38. Bartlett PL, Wegkamp MH (2008) Classification with a reject option using a hinge loss. J Mach Learn Res 9:1823–1840
39. Frank M, Wolfe P (1956) An algorithm for quadratic programming. Naval Res Logist Q 3(1–2):95–110
40. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
41. Smola AJ, Lkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199–222
42. Vert JP, Tsuda K, Lkopf B (2004) A primer on kernel methods. Kernel Methods Comput Biol 47:35–70
43. Platt J (1998) Sequential minimal optimization: a fast algorithm for training support vector machines. Microsoft Research Technical Report
44. Zhang GP, Qi M (2005) Neural network forecasting for seasonal and trend time series. Eur J Oper Res 160(2):501–514
45. Sapankevych NI, Sankar R (2009) Time series prediction using support vector machines: a survey. IEEE Computat Intell Mag 4(2):24–38

Qifeng Zhou received the Ph.D degree in control theory and control engineering from Xiamen University, Xiamen, China in 2007. She is currently an associate professor in the Institute of Pattern Recognition & Intelligent System at Xiamen University. Her research interests include data mining, information retrieval, and reinforcement learning.

Ruyuan Han received the B.S. degree in control theory and control engineering from Xiamen Univetsity, Xiamen, China in 2016. Her research interest includes time series analysis and machine learning.



Tao Li received the Ph.D. degree in computer science from the Department of Computer Science, University of Rochester, Rochester, NY, USA, in 2004. He is a professor with the School of Computing and Information Sciences, Florida International University, Miami, USA. His research interests include data mining, computing system management, and information retrieval.

Bin Xia is currently an assistant professor at School of Computer Science, Nanjing University of Posts and Telecommunications. He received Ph.D degree at School of Computer Science and Engineering, Nanjing University of Science and Technology in 2018. His research area includes deep learning, matrix factorization, and recommender system.