Distributed and Parallel Databases
https://doi.org/10.1007/s10619-020-07294-y

ForeXGBoost: passenger car sales prediction based on XGBoost

Zhenchang Xia[1] · Shan Xue[2] · Libing Wu[3,4] · Jiaxin Sun[1] · Yanjiao Chen[1] · Rui Zhang[1]

© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
The rapid development of machine learning has spurred wide applications to various industries, where prediction models are built to forecast sales to help enterprises and governments make better plans. Alibaba Cloud and the Yancheng Municipal Government held a competition in 2018, calling for global efforts to build machine learning models that can accurately forecast vehicle sales based on large-scale datasets. This paper presents the design, implementation and evaluation of ForeXGBoost, and our proposed model that won the first place in the competition. ForeXGBoost takes full advantage of carefully-designed data filling algorithms for missing values to improve data quality. By using the sliding window to extract historical sales and production data features, ForeXGBoost can improve prediction accuracy. An extensive study is conducted to evaluate the influence of different attributes on vehicle sales via information gain and data correlation, based on which we select the most indicative features from the feature set for prediction. Furthermore, we leverage the XGBoost prediction algorithm to achieve a high prediction accuracy with short running time for vehicle sales prediction. Extensive experiments confirm that ForeXGBoost can achieve a high prediction accuracy with a low overhead.

Keywords Vehicle sales prediction · Feature selection · XGBoost model

1 Introduction

Accurate sales forecast is critical for intelligent transportation [19, 29, 34, 38, 39], which contributes to realizing the intelligent city, and it is able to help automobile companies to plan for production and control budget. The design and production of vehicles require a huge amount of investment and time. By predicting future sales, automobile companies can adjust sales and reduce losses. In recent years, vehicle

✉ Libing Wu
wu@whu.edu.cn

Extended author information available on the last page of the article

Published online: 25 May 2020

sales prediction [3, 27, 32] has attracted great attention from the industry and the academia. Researchers have devoted to improving the accuracy of vehicle sales forecast. Fantazzini et al. [10] proposed a multivariate model that combined economic variables and Google search information to predict monthly German car sales. Hülsmann et al. [18] applied data mining techniques to study economic indicators such as GDP, API, and CPI that affect vehicle sales, and designed a general sales prediction model based on data mining and time series analysis. Kitapci et al. [22] evaluated the impact of economic guidelines on vehicle sales, and predicted vehicle sales by artificial intelligence algorithms such as multivariate regression and neural networks. Lin et al. [25] forecasted vehicle sales in China by using support vector regression (SVR). Most existing works only consider the influence of economic factors on vehicles sales, based on which prediction models are built. However, apart from economic factors, vehicle sales are also affected by many other elements, especially carrelated characteristics that have a direct impact on user experience, e.g., *brand_ID*, *vehicle_model_ID*, *displacement*, *price*, *power*, *fuel_type_ID*, and *size*. Alibaba Cloud and the Yancheng Municipal Government held a competition in 2018, calling for global efforts to build machine learning models that can accurately forecast vehicle sales based on large-scale datasets. In this paper, we present our proposed ForeXGBoost that won the champion of the competition. First, we analyze and process the data, improving the data quality by carefully-designed data filling algorithms for missing values and converting non-numeric data to numeric data. Second, we leverage historic sales and production data to improve prediction accuracy. Our analysis reveals temporal correlations between the current sales and the sales of previous months. Therefore, we propose to use a sliding window to incorporate historic information in the extracted features for prediction. Third, we conduct an extensive study to evaluate the influence of different attributes on vehicle sales via information gain and data correlation, based on which we select the most indicative features from the feature set for prediction. Finally, we choose the XGBoost tree algorithm to build the prediction model after we test various commonly-used machine learning algorithms. Experimental results confirm that ForeXGBoost yields a high prediction accuracy with a short running time since we leverage parallel computing to greatly reduce the training overhead. ForeXGBoost can be generalized to other data-driven prediction tasks, especially those involve time-series data.

Paper structure. Section 2 introduces the challenges faced by vehicle sales forecast and present corresponding solutions. Section 3 introduces the design details of ForeXGBoost, including data pre-processing, feature extraction and model training. Section 4 presents extensive experiments to evaluate the performance of ForeXGBoost against benchmark algorithms. Section 5 discusses limitations and futures works. We review the related works in Sect. 6 and conduct the conclusion in Sect. 7.

2 Challenges and our solutions

In this section, we discuss the challenges facing data-driven prediction for car sales and illustrate our solutions.

2.1 Missing and abnormal data processing

The available datasets for predicting car sales include the automobile sales dataset, the automobile production dataset and the monthly macroeconomic dataset. Nevertheless, some of the key data are missing, which greatly affects the feature extraction and the model training. Some of the data are non-numeric, which cannot be directly used for prediction. It is critically important to transform non-numeric data into numeric data in a way that can best preserve the information and facilitate the feature extraction. Moreover, some data items are abnormal due to potential recording mistakes, and should be filtered to reduce the impact on prediction accuracy. To address these problems, we find that the number of missing data is relatively large for some attributes, e.g., more than 10, 000 missing data for `fuel_type_ID`, `vehicle_model_ID` and `vehicle_department_ID`), but is relatively small for some attributes e.g., fewer than 10 missing data for `displacement`, `power`, `vehicle_length` and `vehicle_height`. According to the influence of different attributes on car sales and the quantity of missing data, we deal with the problem in different ways. For the attributes with enormous missing data and little impact on car sales, e.g., `fuel_type_ID`, we first group the data by certain attributes, such as `province`, then fill the attribute of each group by the mean value. For the attributes with a few missing data but have a significant impact on car sales, we design a precise filling method. For abnormal data and missing data that cannot be filled, we adopt zero padding to minimize their impact on prediction accuracy.

2.2 Feature extraction

As there is a huge amount of data and a large number of attributes, we have to carefully select the features to achieve a high prediction accuracy while reducing the computation cost. Some attributes use discrete values to represent different IDs, where the absolute values do not carry much meaning. Some attributes are non-numeric, such as “81/70” for `power`, which cannot be directly used as features. Some attributes are time-series data, and how to apply time-series features to regression problems is challenging. To extract features from the vast many data and attributes, we first group the attributes into basic information attributes, characteristic attributes, production attributes and sales attributes to reduce the computation overhead of the classifier. For discrete-valued attributes, we use one-hot coding to eliminate the influence of the absolute value on the selection of intermediate nodes in the tree structure. For non-numeric attributes, we convert data to numeric data by deriving the mean of two numbers. For time-series attributes, we propose a novel time-based differential sliding window method to incorporate time-series data to regression algorithms. Finally, we select the features of vehicle information through correlation analysis and information gain.

2.3 Prediction model selection

To achieve precise sales forecasts, we need to obtain as many complete data as possible, and extract useful features to build an accurate and efficient prediction model given specific requirements. More specifically, we should choose the appropriate prediction model and adjust the model parameters by training the classifier on the data with informative features to improve the predict accuracy. After carefully analyzing the business demand and data characteristics of passenger car sales, we select the following machine learning algorithms as the candidate for our passenger car sales prediction: (1) Linear Regression model (LRM) [37], (2) Light Gradient Boosting (LGB) [21], (3) Logistic Regression (LR) [16], (4) Gradient Boosting Decision Tree (GBDT) [36], (5) Decision Tree (DT) [9], (6) Support Vector Machine (SVM) [46] and (7) Extreme Gradient Boosting(XGBoost) [4]. By constantly adjusting parameters and training the model, we select the model which enjoys lower logarithmic difference square root(LDSR) value as final prediction model to predict passenger car sales, so as to improve the predict accuracy. The definition of LDSR is in Sect. 4.

3 ForeXGBoost: design details

The system architecture of ForeXGBoost is shown in Fig. 1. ForeXGBoost consists of three stages: data processing, feature extraction and model training. In the data pre-processing stage, we conduct an in-depth analysis of the original dataset, based on which we preprocess the data to fill in missing data and remove invalid data. In the feature extraction stage, we adopt feature engineering techniques including vehicle-based feature grouping, one-hot coding, and time-difference-based sliding window to form the feature set for model training and prediction. In the model training and prediction stage, the extracted features are fed into the tree-structure model XGBoost. We carefully adjust the parameters of the models to raise prediction

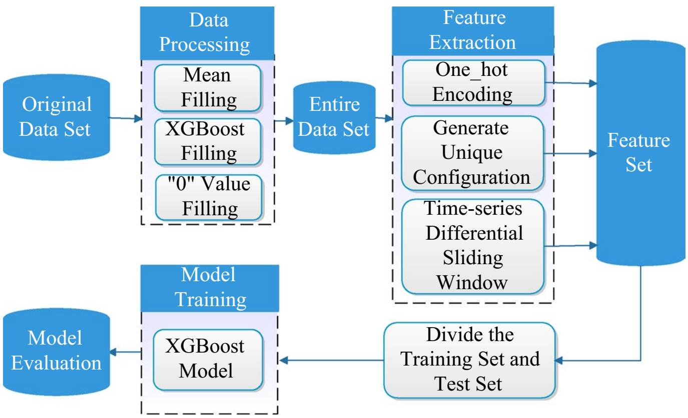


Fig. 1 The system architecture of ForeXGBoost

accuracy. The model with the highest prediction accuracy is chosen to complete the ForeXGBoost forecast system.

3.1 Dataset

To reduce computation costs, we first examine the performance of these models on a small-scale dataset. The small-scale dataset provided by the Alibaba Cloud including the national car sales dataset. The dataset contains more than 5 million data samples and 33 attributes for each data sample, and is from January 2011 to December 2017 in Table 1.

3.2 Missing data processing

For different attributes, we utilize different padding techniques to fill in lost data according to the number of missing data, data attributes, information gain, the influence of the attributes on vehicle sales, etc.

Mean padding. For `fuel_type_ID`, `department_ID` and `level_ID` attributes, the number of missing data is over 10,000. In order to pad the missing data and preserve the integrity of the dataset, we adopt a mean batch filling method. We first group the data as $\{R_j\}_{j=1}^{m}$, where group R_j contains the data with the same `province_ID`, `city_ID`, `brand_ID`, and `model_ID`. For same group of data, we fill in the lost data with the average value of the group. Take the missing data of `fuel_type_ID` in Table 2 as an example. For group R_j, we replace the missing data as $\sum_{k\in R_j} fuel\_type\_ID(j)/|R_j|$, where $\sum_{k\in R_j} fuel\_type\_ID(j)$ is the sum of all non-missing `fuel_type_ID` values in the group, and $|R_j|$ is the number of members in the group.

Prediction padding. For important attributes that have great impact on the prediction accuracy, we propose to fill in the missing data using prediction padding. For instance, `production_quantity` is an important attribute to predict car sales, however, there are some missing data in certain months due to two main reasons. The first reason is that certain months are the downtime of the car manufacturer, and the second reason is that the production may be continued but the data was not recorded. After grouping the entries by the same `province_ID`, `city_ID`, `brand_ID`, and `model_ID`, we first attempt to fill in the missing value of an entry by the entry in the same group with a nearby `sale_date`. If the nearby entries also have empty `production_quantity` values, we use the known basic data by the similarity entry to predict the output in the production cycle, and then fill in the missing data with the predicted value. For example, considering the missing `production_quantity` value of date 201607 in Table 2, we use the similarity of `province_ID`, `city_ID`, `brand_ID`, `produce_date` to predict the `production_quantity`, and we fill the missing data with the predicted value.

Zero padding. For attributes that have unique data in the database, we fill the missing values with zero. For the *provin_ID* is 24, *city_ID* is 232 and *class_ID* is 11145 data in the database, there is no other relevant information about the vehicle,

Table 1 Attributes of the car sales dataset

Attribute	Description	Example
`sale_date`	The sales date of vehicle	201201
`province_ID`	The sales province of vehicle	345 represents the Sichuan Province
`city_ID`	The sales city of vehicle	3456 represents the Chengdu
`class_ID`	The class of vehicle	23456 represents the SUV
`sale_quantity`	The number of vehicles sold on the `sale_date`	15000
`brand_ID`	The brand of vehicle sales	234 represents the Mercedes
`compartment`	The number of compartments reflect the size of the vehicle	3
`type_ID`	The type of vehicle sales	1 represents off-road vehicle
`level_ID`	The level_ID indicates the vehicle is A-Class Car, B-Class Car or C-Class Car	1 represents the vehicle is A-Class Car
`department_ID`	The department_ID represents different models derived from the same platform	1
`TR`	The number of gears	6 shift gears
`gearbox_type`	The transmission type reflects the performance of the engine	Automatic Transmission (AT)
`displacement`	The energy released by the engine per unit time, which reflects the vehicle's engine power	205 L
`if_charging`	The engine is turbocharged or naturally aspirated	L is turbocharged
`price`	The price range of vehicle sales	350,000-500,000
`driven_type_ID`	The driven type indicates the vehicle is front-wheel-driven or rear-wheel-driven	1 represents the front-wheel-driven
`fuel_type_ID`	The fuel type of vehicle	1 represents the fuel is gasoline
`newenergy_type_ID`	It reflects whether the vehicle is a new energy vehicle	1 represents the vehicle is electric car
`emission_standards_ID`	Emission standards indicate control requirements for the amount of common hydrocarbons and other emissions from automobiles	1 represents the sixth stage motor vehicle pollutant emission standard formulated by the state
`if_MPV_ID`	The vehicle is MPV or SUV	1 represents the vehicle is SUV
`if_luxurious_ID`	Whether the vehicle is Â limousine	1 means the vehicle is a luxury car
`power`	The power of the vehicle	160 ps

Table 1 (continued)

Attribute	Description	Example
cylinder_number	The number of cylinders	6
engine_torque	Engine torque	250
car_length	The average length of the vehicle	4531 mm
car_width	The average width of the vehicle	1817 mm
car_height	The average height of the vehicle	1421 mm
total_quality	The weight of the vehicle	1980 kg
equipment_quality	The curbÂ weightÂ	1565 kg
rated_passenger	The number of passengers that can be carried	5
wheelbase	The load capacity within a certain range	2760 mm
front_track	The center line distance of vehicle front wheel	1500 mm
rear_track	The center line distance of vehicle rear wheel	1529 mm

Table 2 Example sales data

sale _date	produce _date	province _ID	city	brand _ID	model _ID	power	fuel_ type_ID	production _quantity
201511	201508	1	30	692	19399	100	1	6832
201512	201508	5	69	638	10705	81	2	7241
201601	201511	15	110	638	10705	68	2	336
201602	201512	15	110	638	10705	68	2	∅
201603	201511	8	98	761	10047	98	∅	5254
201604	201602	16	142	761	10047	98	1	5604
201605	201602	17	225	∅	10085	68	∅	6805
201606	201512	17	225	290	10089	81	4	∅
201607	201601	24	230	750	19389	68	4	∅
201608	201605	24	232	75	10630	94	3	2462
201609	201604	24	232	75	10630	94	3	3653
201610	201608	1	33	12	10067	130	4	5862
201611	201609	19	221	108	10342	180	4	6238

and there is only one data of *class_ID* is 11145 in the data set, so the data can be filled with zero.

3.3 Feature extraction

After filling in the missing data, we extract features for building the prediction model. The original attributes cannot be directly adopted as features. For instance, some attributes are non-numeric, e.g., the data type is string or character. Therefore, we transform the attributes using the following techniques to extract features that are most useful for car sales prediction.

Data integration. We aim at predicting sales of a certain car model of a certain brand. However, the data set subdivides each car model by different powers. For example, as shown in Table 2, there are three different entries for `brand_ID` 638 and `model_ID` 10705 with different powers. Therefore, we need to integrate entries of the same car model of the same `brand_ID`. Let $\mathcal{G}$ denote the set of entries that belong to the same car model of the same brand ID. For a certain attribute, let A_j denote the value of integrated entry. The value of the integrated entry can be computed as $A_j = \sum_{i \in \mathcal{G}} w_i A_{ji}$, where A_{ji} is the attribute value of the entry $i \in \mathcal{G}$, and weight w_i can be calculated as the ratio of vehicle sales of i to the total sales of set $\mathcal{G}$, i.e., $w_i = \mathtt{sales}_i / \sum_{j \in \mathcal{G}} \mathtt{sales}_j$, where $\mathtt{sales}_i$ is the sales of the entry $i \in \mathcal{G}$.

One-hot encoding. Some attributes have discrete values that represent different categories, while the absolute values themselves carry no meanings, e.g., `province_ID`. We need to convert these attributes to extract features that are suitable for machine learning algorithms. More specifically, we use one-hot encoding to treat each discrete value as a state. Suppose that an attribute has N different categories. We encode the attribute using an N-dimensional vector with N different states to ensure that only one state is *active* for each category. Taking `province_ID` as an example, 345 means

Sichuan Province, 456 means Hubei Province and 457 means Henan Province in the database. As the `province_ID` is a digital value, which affects the prediction accuracy of model, we encode `province_ID` by one-hot, such as 00001 means Sichuan Province, 00010 means Hubei Province, 00100 means Henan Province. In this way, we transform discrete values deal for tree-structure learning models.

Sequential sliding window. As shown in Fig. 2, vehicle sales have clear temporal characteristics, namely, periodicity and proximity. The production and sales of vehicles enjoy the characteristics of timing and periodicity. The vehicle sales in one month is similar to those in the same month in previous years, and are closely related to those in the previous month. To extract the features of sales changing trend, we can match the changes in a period this year with the changes in past years by using the sliding window. Another essential reason for feature extraction with a sliding window is that vehicle sales in a certain period of a year may be the same as that in previous years, which can be well captured by using a sliding window.

To leverage the temporal characteristics in sales to improve prediction accuracy, we propose a time-series differential sliding window technique to generate features to account for historical sales. Let k denote the size of the sliding window. In order to extract the feature of the N months sales, we input the sales data of the previous N months into the window whose size is N, and then extracte sales feature of N months. We extract new sales features by sliding the window backwards. As shown in in Fig. 3, we extract features from the sales data of the previous k months. By extensive empirical study, we find that the optimal sliding window size is $k = 6$. With the time-series differential sliding window technique, we can incorporate historic information to predict future sales.

Based on the feature extracted from the sequential sliding window, we can estimate the future value as

$$\hat{p} = f(\mathbf{x}). \tag{1}$$

where $\hat{p}$ is the predicted value based on the time-series $\mathbf{x} = (x_1, x_2, \dots, x_T)$. We may adopt linear or non-linear regression functions as

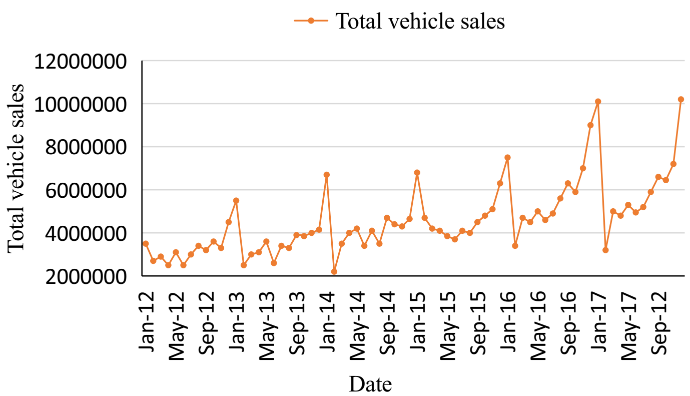


Fig. 2 Temporal characteristics of vehicle sales

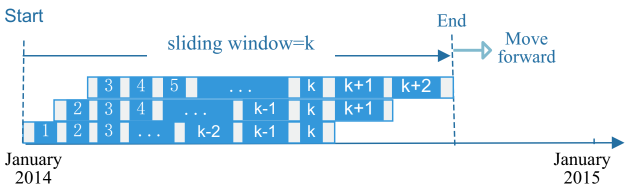


Fig. 3 Time-series differential sliding window

$$f(\mathbf{x}) = \mathbf{a} \cdot \mathbf{x} + b,$$
$$f(\mathbf{x}) = \mathbf{a} \cdot \psi(\mathbf{x}) + b. \tag{2}$$

in which **a** and b are the weights and threshold parameters, $\cdot$ is scalar product. For non-linear regression, we map the data to the high-dimensional space by a certain Kernel function $\psi(\cdot)$ to perform the LR algorithm in the higher-dimensional space [35].

We define the empirical risk function $R(f)$ as

$$R(f) = \frac{1}{T}\sum_{t=1}^{T} L(p, f(\mathbf{x})), \tag{3}$$

where p is the true value, and $L(\cdot)$ is the loss function. The goal is to obtain the ideal weights a and optimal threshold b to minimize the empirical risk function $R(f)$.

$$\hat{f} = \arg\min_{\mathbf{a},b} R(f). \tag{4}$$

Through data integration, one-hot encoding, and sequential sliding window, we obtain the set of features.

3.4 Feature selection

To improve prediction accuracy, we select features that have the most significant influence on vehicle sales and eliminate those that do not have much impact on vehicle sales.

Vehicle sales can be affected by many factors, such as price, vehicle attributes related to user experience and seasonal factors. As shown in Fig. 4, the sales and the price have a non-monotonic relationship, and the sales do not necessarily decrease as the price increases. In fact, when the price is lower than 100,000 RMB, the sales go up as the price gets higher. When the price further increases, the sales will go down. The largest sales are seen in the price range of 100,000 RMB to 150,000 RMB. Similarly, we also observe a non-monotonic relationship between the sales and the vehicle power. As shown in Fig. 5, vehicle sales first increase then decrease as the vehicle power increases, and the vehicles with power 90–120 sell the best.

Fig. 4 Sales distribution for different prices

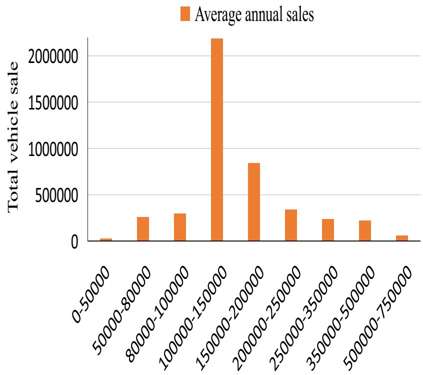


Fig. 5 Sales distribution for different powers

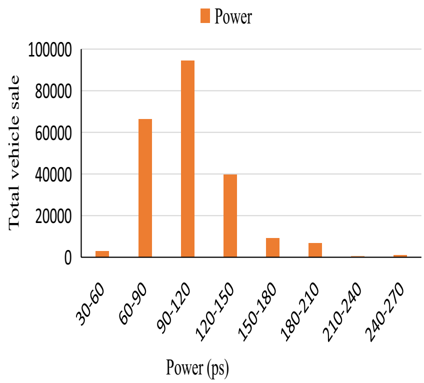
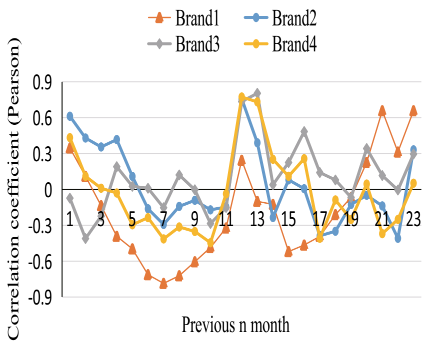
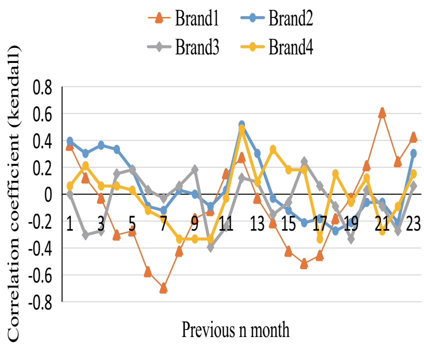


Fig. 6 Pearson correlation coefficient



To quantify the influence of different attributes on sales, we adopt the measurement of correlation and information gain. We choose Pearson correlation and the Kendall correlation to characterize linear and rank correlations between sales and different attributes. We first analyze the correlation between the current sales and the sales of the previous n months. As shown in Figs. 7 and 6, the correlation is the highest between the current sales and the sales of the previous 1 month. As

Fig. 7 Kendall correlation coefficient



the time gap increases, the correlation decreases. Nevertheless, due to periodicity, the correlation is significant between the current sales and the sales of the same month of the previous year, i.e., the previous 12 month. This confirms that we should leverage historical sales data to predict current sales.

Information gain can be used to reveal relationships which are non-monotonic and it cannot be covered by correlation analysis [7, 30]. Information gain evaluates how the knowledge of a certain attribute lowers the uncertainty of the predicted sales. Let m denote the vehicle sales, and n denotes an attribute. The information gain for m, given n, is $[I(m) - I(m|n)]/I(m)$, in which $I(\cdot)$ is the entropy of the metric. The entropy of information indicates how much information we know about random variables, and it can be computed as $I(n) = -\sum_i p(n_i) \ln p(n_i)$, in which $p(n_i)$ is the probability that the value of the random variable n is n_i. If the attribute enjoys a high information gain, it means that a certain attribute is most helpful to predict vehicle sales. As shown in Fig. 8, the attribute `brand_ID` yields the highest information gain. Furthermore, attributes `displacement`, `department_ID` and `compartment`, which are related to user experience, also have high information gain. However, attributes `cylinder_number` and `rated_passenger` have low information gain. The information gain results provide guidance on how to select most indicative features to build the prediction model.

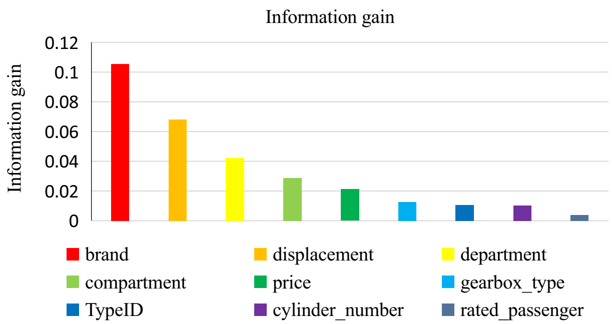


Fig. 8 Information gain of different attributes

3.5 Model selection

To reduce computation costs, we first examine the performance of these models on a small-scale dataset. We train these models on a fraction of the training set, then use the remaining training data to debug and optimize the model. We use logarithmic difference square root (LDSR) as the evaluation metric and select the model that yields the lowest error. Figure 9 shows the LDSR values of different models on the test set and the training set. We can observe that the LRM, GBDT and XGBoost perform well on both the test set and the training set, with small LDSR values and high prediction accuracy. Therefore, we narrow down to LRM, GBDT and XGBoost as the prediction model for vehicle sales. We train these three models on large-scale datasets, and adjust parameters to reach the best possible results for each model. XGBoost yields the most accurate prediction. The ForeXGBoost system based on the XGBoost model was submitted to the Alibaba cloud competition platform, which outperforms all other opponents in achieving the highest prediction accuracy.

3.6 Prediction model

We leverage XGBoost [4, 6] (extreme gradient boosting) to build the prediction model to forecast vehicle sales. XGBoost was first proposed in 2014 for implementing scalable, portable, distributed gradient boosting. As a new tree-structure learning algorithm, XGBoost consists of many decision trees, using parallel and distributed computing to speedup the learning process and model exploration. XGBoost integrates multiple weak learners together so that the accuracy is higher than any single learner. In addition to the CART (classification and regression tree) model, XGBoost can use a linear classifier as the base learner. Thanks to its superiority in efficiency and prediction accuracy, XGBoost has been widely used in many classification and regression problems such as store sales prediction [33], web text classification [40], customer behavior prediction [8], malware classification, click-through rate forecast [26] and product classification [14].

The predicted vehicle sales $\widehat{p}$ given by the XGBoost learner is

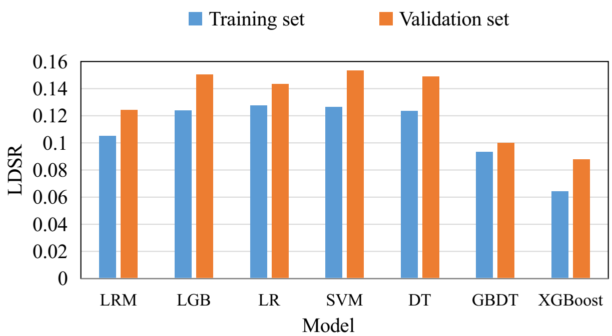


Fig. 9 LDSR of different prediction models

$$\widehat{p} = \sum_{k=1}^{K} f_k(\mathbf{x}) \tag{5}$$

where f_k is the k-th tree, $\mathbf{x}$ is the set of extracted features. The predicted value of the K-th tree is

$$\hat{p}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{p}_i^{(t-1)} + f_t(x_i), \tag{6}$$

where f_k is the k-th tree, $\hat{p}_i^{(t)}$ represents the prediction results of the combined T tree models for sample x_i, $f_t(x_i)$ represents the best available tree model. XGBoost improves prediction accuracy by continuously adding the best tree model [5].

XGBoost tries to minimize the loss function as

$$L(\{f_k\}_{k=1}^{K}) = \sum_{i=1}^{n} L(p_i, \hat{p}_i) + \sum_{k=1}^{K} \Omega(f_k), \tag{7}$$

where n counts the total samples, $L(p_i, \hat{p}_i)$ quantifies the difference between the real sales p_i and the predicted sales $\widehat{p}_i$, $\Omega(f_k)$ is the regularization item used to control the complexity of the tree structure. In XGBoost, the complexity is defined as

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^{(2)}, \tag{8}$$

where T indicates the number of model leaf nodes, γ is the drop value of the minimum loss function for node splitting, λ is the L2 regularization term for weights. $w_j^{(2)}$ represents the score obtained by the leaf node. The algorithm is more conservative when the γ is larger. The L2 modulus square of the weights is used to control the complexity of the tree and prevent overfitting.

We have to tune XGBoost parameters in order to obtain the optimal prediction model. Its parameters consist of three categories: (1) general parameters, (2) booster parameters, and (3) task parameters [31]. Table 3 lists all parameters and their descriptions.

Model training and parameter tuning is an iterative process. We continually resize the parameters in XGBoost to obtain the ideal values that minimize the logarithmic square root of the difference between the predicted sales and the real sales. In the process of parameter tuning, we first initialize the number of decision trees and the value of *eta*, then we search for the ideal values for parameters *dmax*, *wmin*, *gamma*, and *sp*. After parameter tuning, we regularize the XGBoost model to reduce the complexity and improve the prediction accuracy. Finally, the parameters *eta* and *wmin* are readjusted to search for the optimal injection molding parameters. The training process for ForeXGBoost is summarized in Algorithm 1.

Table 3 Parameters and their descriptions

Parameter type	Parameter name	Parameter description
General parameters	*bt*	*bt* is used to select the iteration model, including tree-based models and linear models.
	st	*st* is used to determine if there is a message that needs to be output.
	nd	*nd* is used for multi-thread control.
Booster parameters	*eta*	*eta* is the learning rate, which prevents the model from over-fitting, and its scale is [0, 1].
	gamma	The medol required minimum loss reduction to make a further partition on a leaf node of the tree.
	dmax	*dmax* is the maximal depth of tree, and its scale is [0, ∞].
	wmin	*wmin* is the minimal sum of sample weights in child nodes. If the sample weight of a leaf node is less than *wmin*, the splitting process ends.
	smax	*smax* is the maximal delta step we allow each tree's weight estimation to be. If the value is set to 0, it means there is no constraint. If it is set to a positive value, it can help to make the update step more conservative. and its scale is [0, ∞]
	sp	*sp* is the proportion of sub-samples used in training model to the whole sample set, and its scale is (0, 1]
	tc	*tc* is the proportion of sampling feature when building a tree, and its scale is (0, 1]
Task parameters	*bs*	*bs* is the initial prediction score of all instances, global bias.
	me	*me* is the evaluation index which can be used to verify data.
	sd	*sd* parameter is the random seed, and it can be used to generate reproducible results and tune parameter.

Algorithm 1 ForeXGBoost Training

Require: Historical observations $\{x_1, x_1, \cdots, x_n\}$, set of features $\{e_1, e_2, \cdots, e_n\}$.
Ensure: Learned ForeXGBoost model.

Input D data samples.
$//Select\ \ the\ \ best\ \ model$
Initialize eta and $wmin$
$//Train\ \ the\ \ model$
for all parameters **do**
    Input parameter into the ForeXGBoost model
    for $k = 1$ to num_round **do**
        Feed data samples into the model
        Train the ForeXGBoost model
    end for
end for
Select the best model
Output the model

In order to determine the booster parameters, we need to continuously tune the parameters in the model to generate LDSR. In detail, the parameter selection is as follows.

1. Train the model with *dmax* chosen from $\{3, 5, 7, 9\}$ and *wmin* chose from $\{1, 3, 5\}$. $eta = 0.1$ is fixed. Figures 10 and 11 respectively show the LDSR of

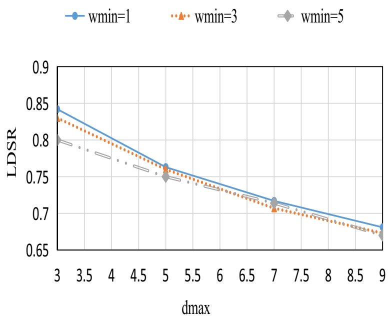


Fig. 10 Parameter tuning for *dmax* and *wmin* in the training set

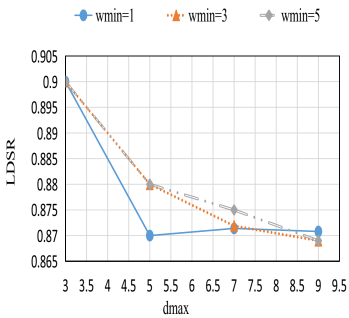


Fig. 11 Parameter tuning for *dmax* and *wmin* in the validation set

the training set and the validation set. We determine that the optimal values are $dmax = 9, wmin = 5$.

2. Train the model with $gamma$ chosen from $\{0, 1, 2, 3, 4\}$ given that $eta = 0.1, dmax = 9, wmin = 5$. Figure 12 shows the LDSR of the training set and the validation set. We determine that the optimal value is $gamma = 0$.
3. Train the model with sp and tc chosen both from $\{0.6, 0.7, 0.8, 0.9\}$ given that $eta = 0.1, dmax = 9, wmin = 5, gamma = 0$. Figures 13 and 14 respectively show the LDSR of the training set and the validation set. We determine that the optimal values are $sp = 0.7, tc = 0.7$.
4. Train the model with $lambda$ chosen from $\{1, 10, 50, 100, 500\}$ given that $eta = 0.1, dmax = 9, wmin = 5, gamma = 0, sp = 0.7, tc = 0.7$. Figure 15 shows

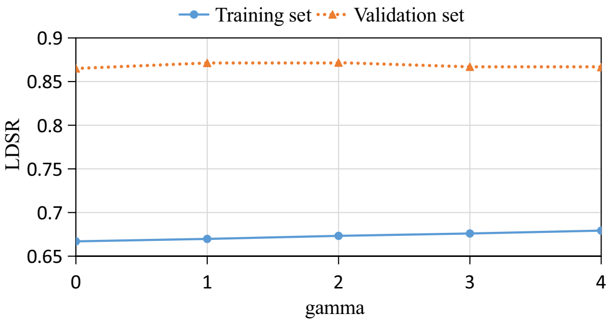


Fig. 12 Parameter tuning for *gamma*

Fig. 13 Parameter tuning for *sp* and *tc* in the training set

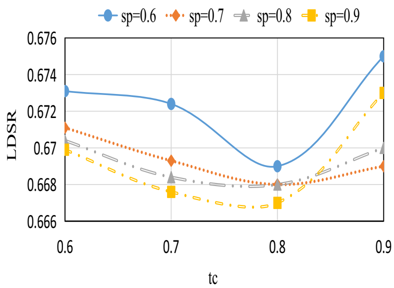
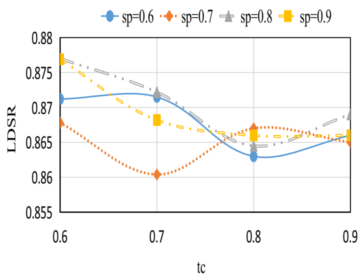


Fig. 14 Parameter tuning for *sp* and *tc* in the validation set



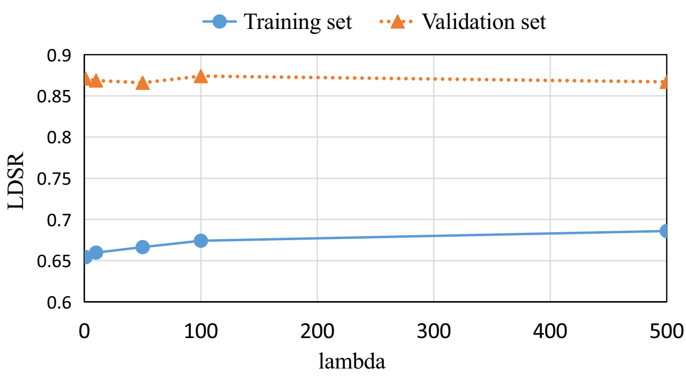


Fig. 15 Parameter tuning for *lambda*

the LDSR of the training set and the validation set. We determine that the optimal value is $lambda = 50$.

5. Train the model with eta chosen from $\{0.05, 0.1, 0.2\}$ and $wmin$ chosen from $\{5, 50, 100, 200, 500\}$ given that $dmax = 9, gamma = 0, sp = 0.7, tc = 0.7, lambda = 50$. Figures 16 and 17 respectively show the LDSR of the training set and the validation set. We determine that the optimal values are $eta = 0.1$ and $wmin = 50$.

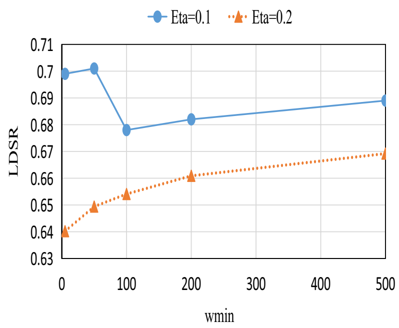


Fig. 16 Parameter training for *eta* and *wmin* in the training set

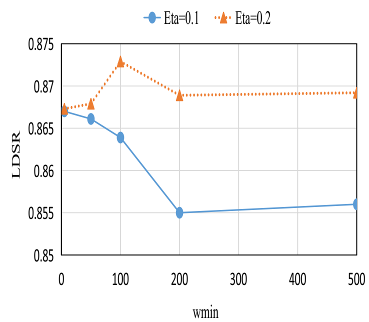


Fig. 17 Parameter training for *eta* and *wmin* in the validation set

After parameter tuning, we can find the optimal values for all parameters as $eta = 0.1$, $dmax = 9$, $wmin = 50$, $gamma = 0$, $sp = 0.7$, $tc = 0.7$, $lambda = 50$.

4 Evaluation

4.1 Setup

We utilize the large-scale dataset provided by the Alibaba Cloud including the national car sales dataset for different brands, the national car production dataset and the national monthly macroeconomic dataset [1]. The volume size of the dataset is more than 1.3 GB. The national car sales dataset includes data from January 2011 to December 2017. The dataset has 33 attributes and 11 million entries. The national car production dataset includes data from January 2012 to May 2017 which has 3 attributes and 490,000 entries. The national monthly macroeconomic dataset includes data from January 2012 to May 2017. The dataset has 4 attributes. Our aim is to forecast the passenger car sales in February 2018, thus we choose the data from January 2013 to October 2017 as the training set, and the data in December 2017 as the test set. We use large-scale dataset to test and validate the selected prediction model.

We evaluate the performance of ForeXGBoost in terms of root mean square error (RMSE), Logarithmic difference square root (LDSR), mean absolute percentage error (MAPE) and running time.

Root mean square error (RMSE). RMSE measures the prediction accuracy and is defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - \hat{p}_i)^2}, \tag{9}$$

where n indicates how many samples are in the data set, $\hat{p}_i$ indicates the i-th sample's the predicted value, and p_i is i-th sample's true value.

Logarithmic difference square root (LDSR). To lower the negative influence of imprecise sales forecasts of best-selling vehicles, we measure the LDSR, defined as

$$LDSR = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i + 1) - \log(\hat{p}_i + 1))^2}. \tag{10}$$

Mean Absolute Percentage Error (MAPE). The mean of RMSE is sensitive to outliers, thus we use the quantile of the error to replace the average as

$$MAPE = \sum_{i=1}^{n}\left(\frac{p_i - \hat{p}_i}{p_i}\right) \cdot \frac{100}{n}. \tag{11}$$

Running time. We evaluate the running time of the training set and the test set.

4.2 Prediction accuracy

We compare the prediction accuracy [45] of the proposed XGBoost-based prediction model with linear regression and GBDT. We choose 4,429,915 data samples as the training set, and 105,609 data samples as the test set (their ratio is 44 : 1). As shown in Table 4, the prediction accuracy of ForeBooster is higher than the GBDT model and the linear regression model for both evaluation metrics. The LDSR of XGBoost on the training set is reduced by 58% compared with the linear regression model, and by 9.4% compared with the GBDT model. The LDSR of XGBoost on test set is 65% lower than that of the linear regression model, and 6.2% lower than that of the GBDT model. The results confirm that XGBoost is superior over the GBDT model and the linear regression model in terms of prediction accuracy. This is because XGBoost can learn the splitting direction of the sample with missing values while the linear regression model and the GBDT model cannot. The cost function of XGBoost performs the second-order Taylor expansion, and utilizes both the first-order and the second-order derivatives. The regular function is also added to the cost function, which can help to limit the complexity of XGBoost and make the learned model simpler and prevent overfitting. Moreover, the cost function of the current iteration of XGBoost contains the predicted value of the previous iteration, which helps improve the prediction accuracy.

4.3 Running time

To evaluate the efficiency of ForeXGBoost, we measure the running time of training the model and using the model for prediction. We choose 4,429,915 data samples as the training set, and 105,609 data samples as the test set. The three models are trained for the same number of rounds. Figure 18 illustrates the running time of different models. We can observe that ForeXGBoost takes less time to complete the whole process than GBDT and the linear regression model for different numbers of training iterations. This is because that ForeXGBoost supports parallel processing. In particular, the node splitting based on the feature gain to determine the best separation point can be parallelized to improve the efficiency of

Table 4 Prediction accuracy

Model	LDSR		MAPE	
	Training set	Test set	Training set	Test set
Linear regression	1.25216	1.54206	26.21610	28.27289
GBDT	0.76383	0.88968	18.99992	19.88037
XGBoost	**0.66950**	**0.88347**	**16.23195**	**18.54067**

This bold is to emphasize this line is the predictive accuracy of our proposed prediction system

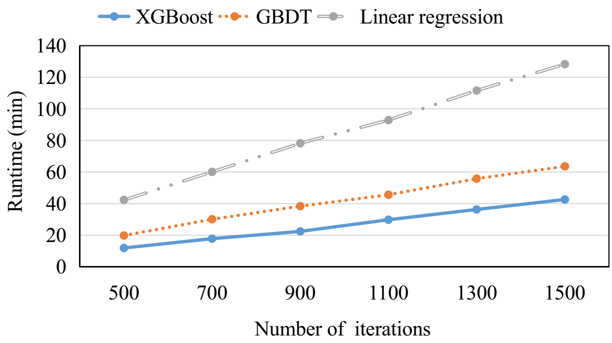


Fig. 18 Running time

the model. Furthermore, XGBoost benefits from the cache compression sensing algorithm that further improves the system efficiency.

4.4 Optimal iteration

ForeXGBoost aims at realizing the desirable prediction accuracy with the lowest possible computational cost. To achieve this goal, we have to achieve a balance between the prediction accuracy and the learning speed, tree depth, maximum number of nodes, row sampling, column sampling and feature acquisition of the XGBoost model. In particular, we have to determine the ideal number of iterations for training XGBoost model. In order to obtain the best prediction model, we evaluate the optimal number of iterations of the XGBoost model based on the selected model parameters in Sect. 3. Figure 19 demonstrates that as the number of iterations increases, the LDSR of the test set and the training set will increase, and the prediction accuracy will decrease. Therefore, we choose 1100 iterations for training the XGBoost model.

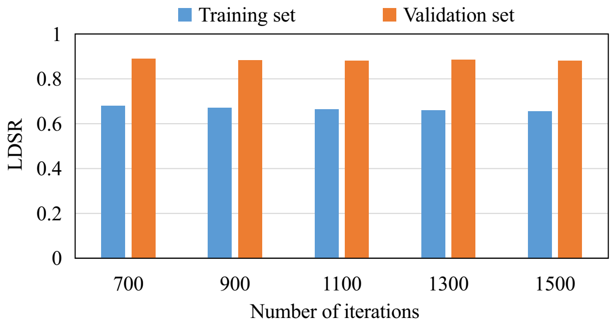


Fig. 19 LDSR of different iterations

5 Discussion

Applicability of ForeXGBoost. ForeXGBoost can achieve a very high accuracy for vehicle sales prediction based on the attributes of vehicles, and historic production and sales information. ForeXGBoost relies on the temporal correlation to incorporate time-series data for prediction. During the model training, parallel processing is conducted to split nodes based on the feature gains and determine the best separation point of features, which significantly reduces the training overhead. Given its high prediction accuracy and low training overhead, we envisage that ForeXGBoost will be of great use for vehicle sales prediction, which can be leveraged by governments and car manufacturers to better enact production plans and investment policies.

Extension of ForeXGBoost to other applications. Although ForeXGBoost is designed for vehicle sales prediction, many of the proposed schemes in ForeXGBoost can be generalized to other prediction tasks based on time-series data. When performing machine learning, some public databases may also suffer from the problems of missing data, non-numeric data, and data anomalies, thus data filling techniques can be used to preprocess the datasets for feature extraction. The proposed feature extraction from time-series data, which accounts for the influence of historic data on future predictions, can be applied to various areas, e.g., commodity sales, financial market, power load, weather and environmental status. The proposed model training and parameter tuning techniques can help improve the performance of XGBoost-based prediction models.

Limitations and future works. ForeXGBoost model can predict vehicle sales with a relatively high accuracy, but we did not leverage economic indicators such as consumer price index (CPI) and gross domestic product (GDP) in our prediction model. A possible way to improve the prediction accuracy of ForeXGBoost is to conduct an in-depth analysis of the influence of economic indicators on the passenger car sales, based on which we can incorporate economic indicators in the prediction model. Moreover, we only evaluate the performance of ForeXGBoost on the dataset for China's market. In the future, we intend to carry out global vehicle sales forecast by taking into consideration national economic policies, imports and exports, exchange rates and other factors.

6 Related work

Vehicle sales prediction. Due to a lack of available public datasets, there is few research literature on vehicle sales prediction. Most existing works focus on the impact of economic factors on vehicle sales, without a comprehensive analysis on vehicle-related attributes such as *brand*, *aerodynamic volume displacement*, *power*, *fuel_type*, and *vehicle size*. Gao et al. [11] proposed a hybrid model that is based on ant colony method and particle swarm optimization to forecast automobile sales. The prediction model is designed for automobile sales forecast employing GDP, CPI, highway mileage, and automobile ownership. Wang et al. [41] proposed a stepwise regression model to select the most influential economic indicators such as current automobile sales, industrial production index, stock price index, automobile price, oil price, unemployment rate and exchange rate. However, these models only adopt economic indicators to predict nationwide sales. It is not enough to predict the car sales of different brands only based on regional economic indicators. ForeXGBoost is the first to use XGBoost for vehicle sales prediction that can accurately predict the sales of vehicles based on various available information.

Prediction model based on time-series data. Prediction models based on time-series data [13, 20, 28, 43, 47, 50] have been applied in various attributes such as commodity sales forecasting [52], financial market forecasting [17, 42], electrical utility loading prediction [15], weather and environmental parameter estimation [23, 35, 49, 53]. According to different prediction characteristics, the models can be divided into linear prediction model and nonlinear prediction model. The linear prediction models focus on predicting time series data by using linear algorithms. For example, Yuan et al. [48] forecasted primary energy consumption which enjoys the attribute of time-series by using the linear model ARIMA. The non-linear prediction models focus on using and training non-linear algorithms such as neural networks and k-means algorithm. For example, Kuremoto et al. [24] proposed the DBN model that captures the feature of the input space of time series data for time series prediction. Gao et al. [12] proposed a time-aware item prediction model, which utilized the evolution of both user's profiles and item's topics for item recommendation. Astakhova et al. [2] utilized the k-means algorithm to implement the forecasting method for time series groups, which can improve prediction accuracy. Common linear models include basic least squares models, auto-regressive models, and linear regression models. Common nonlinear models [44, 51] include support vector machines (SVMs), DT, GBDT, neural network, and XGBoost. With an intensive analysis and empirical study, we choose XGBoost to build our prediction model.

7 Conclusion

In this paper, we presented the design, implementation and evaluation of ForeXGBoost, a sales forecast system based on large-scale datasets with comprehensive information including the vehicle's *brand_ID*, *model*, *power*, and *displacement*. To build a predicted model, we first propose a carefully-designed data filling algorithms that can improve the data quality by "0" value filling, mean filling and XGBoost filling. After this, a sliding window technology based on historical sales and production data is leveraged and combines with one-hot coding technology, which can extract the feature value of vehicle information. Our extensive experiments show that XGBoost outperforms benchmark algorithms in both prediction accuracy and overhead. ForeXGBoost won the champion of Alibaba Cloud's vehicle sales forecast competition with 5% margin over the second-place competitor.

Acknowledgements Funding was provided by the National Natural Science Foundation of China (Nos. 61772377, 61572370, 91746206), the Natural Science Foundation of Hubei Province of China (No. 2017CFA007), Science and Technology planning project of ShenZhen (JCYJ20170818112550194).

References

1. Alibaba Cloud TIANCHI Prediction of Passenger Car Sales Challenge (2018). https://tianchi.aliyun.com/competition/ information.htm?raceId=231640
2. Astakhova, N.N., Demidova, L.A., Nikulchev, E.V.: Forecasting method for grouped time series with the use of k-means algorithm. Contemp. Eng. Sci. **8**(2015), 1659–1677 (2015)
3. Barfield, J.R., Welch, S., Taylor, T.S., et al.: Prediction of vehicle transactions and targeted advertising using vehicle telematics. US Patent App. 14/197,286 (2015)
4. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: ACM International Conference on Knowledge Discovery and Data Mining (2016)
5. Chen, T., He, T.: Higgs Boson discovery with boosted trees. In: NIPS 2014 Workshop on High-Energy Physics and Machine Learning (2015)
6. Chen, T., He, T., Benesty, M., et al.: Xgboost: extreme gradient boosting. R package version **04–2**, 1–4 (2015a)
7. Chen, Y., Chen, Q., Zhang, F., Zhang, Q., Wu, K., Huang, R., Zhou, L.: Understanding viewer engagement of video service in wi-fi network. Comput. Netw. **91**, 101–116 (2015b)
8. Do, D., Huynh, P., Vo, P., Vu, T.: Customer churn prediction in an internet service provider. In: IEEE International Conference on Big Data (Big Data), pp. 3928–3933 (2017)
9. Drucker, H., Cortes, C.: Boosting decision trees. In: Advances in Neural Information Processing Systems, pp. 479–485 (1996)
10. Fantazzini, D., Toktamysova, Z.: Forecasting German car sales using Google data and multivariate models. Int. J. Product. Econ. **170**, 97–135 (2015)
11. Gao, J., Xie, Y., Gu, F., Xiao, W., Hu, J., Yu, W.: A hybrid optimization approach to forecast automobile sales of China. Adv. Mech. Eng. **9**(8), 1687814017719422 (2017a)
12. Gao, L., Wu, J., Zhou, C., Hu, Y.: Collaborative dynamic sparse topic regression with user profile evolution for item recommendation. In: Thirty-First AAAI Conference on Artificial Intelligence (2017b)
13. Hassan, M., Yang, M., Rasheed, A., Jin, X., Xia, X., Xiao, Y., He, Z.: Time-series multispectral indices from unmanned aerial vehicle imagery reveal senescence rate in bread wheat. Remote Sens. **10**(6), 809 (2018)
14. Hebert, J.: Predicting rare failure events using classification trees on large scale manufacturing data with complex interactions. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 2024–2028 (2016)
15. Hong, T., Fan, S.: Probabilistic electric load forecasting: a tutorial review. Int. J. Forecast. **32**(3), 914–938 (2016)
16. Hosmer Jr., D.W., Lemeshow, S., Sturdivant, R.X.: Applied Logistic Regression. Wiley, New York (2013)
17. Hsu, M.W., Lessmann, S., Sung, M.C., Ma, T., Johnson, J.E.: Bridging the divide in financial market forecasting: machine learners vs. financial economists. Expert Syst. Appl. **61**, 215–234 (2016)
18. Hülsmann, M., Borscheid, D., Friedrich, C.M., Reith, D.: General sales forecast models for automobile markets and their analysis. Trans MLDM **5**(2), 65–86 (2012)
19. Javed, M.A., Zeadally, S., Hamida, E.B.: Data analytics for cooperative intelligent transport systems. Veh. Commun. **15**, 63–72 (2019)
20. Jiang, B., Fei, Y.: Vehicle speed prediction by two-level data driven models in vehicular networks. IEEE Trans. Intell. Transport. Syst. **18**(7), 1793–1801 (2016)
21. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, pp. 3146–3154 (2017)

22. Kitapcı, O., Özekicioğlu, H., Kaynar, O., Taştan, S.: The effect of economic policies applied in Turkey to the sale of automobiles: multiple regression and neural network analysis. Procedia **148**, 653–661 (2014)
23. Koochakpour, K., Tarokh, M.J.: Sales budget forecasting and revision by adaptive network fuzzy base inference system and optimization methods. J. Comput. Robot. **9**(1), 25–38 (2016)
24. Kuremoto, T., Kimura, S., Kobayashi, K., Obayashi, M.: Time series forecasting using a deep belief network with restricted Boltzmann machines. Neurocomputing **137**, 47–56 (2014)
25. Lin, K., Lin, Q., Zhou, C., Yao, J.: Time series prediction based on linear regression and SVR. In: IEEE International Conference on Natural Computation (2007)
26. Ling, X., Deng, W., Gu, C., Zhou, H., Li, C., Sun, F.: Model ensemble for click prediction in bing search ads. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 689–698 (2017)
27. Litman, T.: Autonomous Vehicle Implementation Predictions. Victoria Transport Policy Institute, Victoria (2017)
28. Lu, W.X., Zhou, C., Wu, J.: Big social network influence maximization via recursively estimating influence spread. Knowl. Based Syst. **113**, 143–154 (2016)
29. Meneguette, R.I.: A vehicular cloud-based framework for the intelligent transport management of big cities. Int. J. Distrib. Sens. Netw. **12**(5), 8198597 (2016)
30. Mitchell, T.M., Learning, M.: Mcgraw-Hill science. Eng. Math. **1**, 27 (1997)
31. Nielsen, D.: Tree Boosting With XGBoost-Why Does XGBoost Win “Every” Machine Learning Competition? Master’s Thesis, NTNU (2016)
32. Pai, P.F., Liu, C.H.: Predicting vehicle sales by sentiment analysis of twitter data and stock market values. IEEE Access **6**, 57655–57662 (2018)
33. Pavlyshenko, B.M.: Linear, machine learning and probabilistic approaches for time series analysis. In: IEEE First International Conference on Data Stream Mining & Processing (DSMP), pp. 377–381 (2016)
34. Perallos, A., Hernandez-Jayo, U., Zuazola, I.J.G., Onieva, E.: Intelligent Transport Systems: Technologies and Applications. Wiley, New York (2015)
35. Sapankevych, N.I., Sankar, R.: Time series prediction using support vector machines: a survey. IEEE Comput. Intell. Mag. **4**, 2 (2009)
36. Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT, Cambridge (2001)
37. Seber, G.A., Lee, A.J.: Linear Regression Analysis. Wiley, New York (2012)
38. Sjoberg, K., Andres, P., Buburuzan, T., Brakemeier, A.: Cooperative intelligent transport systems in europe: current deployment status and outlook. IEEE Veh. Technol. Mag. **12**(2), 89–97 (2017)
39. Sładkowski, A., Pamuła, W.: Intelligent Transportation Systems-Problems and Perspectives, vol. 303. Springer, Berlin (2016)
40. Stein, R.A., Jaques, P.A., Valiati, J.F.: An analysis of hierarchical text classification using word embeddings. Inf. Sci. **471**, 216–232 (2019)
41. Wang, F.K., Chang, K.K., Tzeng, C.W.: Using adaptive network-based fuzzy inference system to forecast automobile sales. Expert Syst. Appl. **38**(8), 10587–10593 (2011)
42. Wang, J., Wang, J., Fang, W., Niu, H.: Financial time series prediction using elman recurrent random neural networks. Comput. Intell. Neurosci. (2016). https://doi.org/10.1155/2016/4742515
43. Weigend, A.S.: Time Series Prediction: Forecasting the Future and Understanding the Past. Routledge, London (2018)
44. Wu, J., Cai, Z., Zeng, S., Zhu, X.: Artificial immune system for attribute weighted Naive Bayes classification. In: IEEE the 2013 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2013)
45. Wu, J., Pan, S., Zhu, X., Zhang, C., Wu, X.: Multi-instance learning with discriminative bag mapping. IEEE Trans. Knowl. Data Eng. **30**(6), 1065–1080 (2018)
46. Ye, J., Chow, J.H., Chen, J., Zheng, Z.: Stochastic gradient boosted distributed decision trees. In: ACM Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 2061–2064 (2009)
47. Yu, H.F., Rao, N., Dhillon, I.S.: Temporal regularized matrix factorization for high-dimensional time series prediction. In: Advances in Neural Information Processing Systems, pp. 847–855 (2016)

48. Yuan, C., Liu, S., Fang, Z.: Comparison of China's primary energy consumption forecasting by using ARIMA (the autoregressive integrated moving average) model and GM (1, 1) model. Energy **100**, 384–390 (2016)
49. Zaytar, M.A., El Amrani, C.: Sequence to sequence weather forecasting with long short-term memory recurrent neural networks. Int. J. Comput. Appl. **143**(11), 7–11 (2016)
50. Zhang, Q., Wu, J., Yang, H., Tian, Y., Zhang, C.: Unsupervised feature learning from time series. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, AAAI Press, IJCAI'16, pp. 2322–2328 (2016)
51. Zhang, Y., Wu, J., Zhou, C., Cai, Z.: Instance cloned extreme learning machine. Pattern Recogn. **68**, 52–65 (2017)
52. Zhao, K., Wang, C.: Sales Forecast in E-commerce using Convolutional Neural Network. arXiv preprint arXiv:170807946 (2017)
53. Zheng, H., Yuan, J., Chen, L.: Short-term load forecasting using EMD-LSTM neural networks with a Xgboost algorithm for feature importance evaluation. Energies **10**(8), 1168 (2017)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Zhenchang Xia[1] · Shan Xue[2] · Libing Wu[3,4] · Jiaxin Sun[1] · Yanjiao Chen[1] · Rui Zhang[1]

Zhenchang Xia
xiazhenchang@whu.edu.cn

Shan Xue
emma.xue@mq.edu.au

Jiaxin Sun
1534545218@qq.com

Yanjiao Chen
chenyanjiao@whu.edu.cn

Rui Zhang
zrkl100@foxmail.com

[1] School of Computer Science, Wuhan University, Wuhan, China

[2] Department of Computing, Faculty of Science and Engineering, Macquarie University, Sydney, Australia

[3] School of Cyber Science and Engineering and School of Computer Science, Wuhan University, Wuhan, China

[4] Shenzhen Research Institute of Wuhan University, Shenzhen, China