Transportation Research Part C 58 (2015) 308–324

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

A gradient boosting method to improve travel time prediction

Yanru Zhang [*], Ali Haghani

Department of Civil & Environmental Engineering, University of Maryland, College Park, MD 20740, United States

A R T I C L E   I N F O

Article history:
Received 2 May 2014
Received in revised form 17 February 2015
Accepted 19 February 2015
Available online 29 March 2015

Keywords:
Short-term forecasting
Travel time
Ensemble learning
Machine learning
Gradient boosting regression tree
Random forest

A B S T R A C T

Tree based ensemble methods have reached a celebrity status in prediction field. By combining simple regression trees with 'poor' performance, they usually produce high prediction accuracy. In contrast to other machine learning methods that have been treated as black-boxes, tree based ensemble methods provide interpretable results, while requiring little data preprocessing, are able to handle different types of predictor variables, and can fit complex nonlinear relationship. These properties make the tree based ensemble methods good candidates for solving travel time prediction problems. However, applications of tree-based ensemble algorithms in traffic prediction area are limited. In this paper, we employ a gradient boosting regression tree method (GBM) to analyze and model freeway travel time to improve the prediction accuracy and model interpretability. The gradient boosting tree method strategically combines additional trees by correcting mistakes made by its previous base models, therefore, potentially improves prediction accuracy. Different parameters' effect on model performance and correlations of input–output variables are discussed in details by using travel time data provided by INRIX along two freeway sections in Maryland. The proposed method is, then, compared with another popular ensemble method and a bench mark model. Study results indicate that the GBM model has its considerable advantages in freeway travel time prediction.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

Travel time, as an effective measure of freeway traffic conditions, can be easily understood by both travelers and transportation managers (Yeon et al., 2008). The provision of accurate travel time information through Advanced Traveler Information Systems (ATIS) enables travelers to make informed decisions for departure time, route, and mode choice, and therefore releases travelers' stress and anxiety. In addition, predicted travel time is valuable information for transportation managers in proactively developing Advanced Traffic Management System (ATMS) strategies. The Freeway Performance Measurement System (PeMS), the Split Cycle Offset Optimization Technique System, and the Sydney Coordinated Adaptive Traffic System are all successful traffic operation systems that either use travel time information as performance measures or as their module input (Choe et al., 2002; Yang et al., 2010). Apart from its important role in the successful implementation of intelligent transportation systems (ITS), travel time estimation and prediction are complex and challenging tasks. Resulting from the interactions among different vehicle-driver combinations, and exogenous factors such as weather, demand, and roadway conditions, travel time often experiences strong fluctuations across different periods and traffic conditions. These rapid fluctuations are often complex and difficult to predict. Fully understanding these fluctuations and developing accurate travel time prediction algorithms is critical.

[*] Corresponding author.

http://dx.doi.org/10.1016/j.trc.2015.02.019
0968-090X/© 2015 Elsevier Ltd. All rights reserved.

Inspired by the need of travel time predictions, a wide range of methodologies has been proposed in the literature, including historical average and smoothing (Farokhi Sadabadi et al., 2010; Smith and Demetsky, 1997; Williams et al., 1998), statistical and regression methods (Min and Wynter, 2011; Fei et al., 2011; Li et al., 2013b), various machine learning techniques (Zhang and Rice, 2003; Wang and Shi, 2012; Wei and Chen, 2012) and traffic flow theory based methods (Li et al., 2013a, 2014). For an extensive review of existing traffic prediction models refer to review papers (Vlahogianni et al., 2014; Van Lint and Van Hinsbergen, 2012). Among these methods, autoregressive integrated moving average (ARIMA) type model is widely recognized as an accepted framework to build freeway traffic prediction model, due to its well-defined theoretical foundations and effectiveness in prediction (Karlaftis and Vlahogianni, 2009). The ARIMA type model is gradually becoming a benchmark to compare with newly developed prediction models (Zhang et al., 2013b). There is a vast amount of freeway traffic prediction literature on such models (Kamarianakis and Prastacos, 2005; Min and Wynter, 2011; Zhang et al., 2013a). The ARIMA type or, more generally, statistical modeling approaches, assume certain model structures for the data. These models provide interpretable parameters with straightforward model structures. They can make very good predictions when traffic shows regular variations. Another type of promising prediction method, the machine learning (ML) algorithm, also generates great interest in traffic prediction filed. Some successful applications include: neural networks (Van Lint et al., 2005; Wei and Chen, 2012; Van Hinsbergen et al., 2009), support vector machines (Wang and Shi, 2012; Hong, 2011), and hybrid or ensemble techniques (Antoniou et al., 2013; Chen and Wu, 2012; Wang and Shi, 2012). In contrast to the statistical modeling approach, machine learning algorithm does not assume any specific model structure for the data but treat it as unknown. Therefore, it is more efficient in handling large data with any degree of complexity. The ML method can also capture the underlying structure of data even when they are not apparent (Vlahogianni et al., 2014). One of the disadvantages of most machine learning algorithms is the lack of interpretability, which limits their applications in traffic prediction.

In recent years, ensemble based algorithms reached a celebrity status in solving prediction and classification problems. They have been applied to different fields and have achieved great success (Zhou, 2012). The $1 Million Netflix Prize competition is a famous example: the winning team ensembles different algorithms to predict user rating for films, which produces the best accuracy among all participates (Koren, 2009). Within all different ensemble methods, the tree-based ensemble method is a popular one. Instead of fitting a single "best" model, the tree-based ensemble method strategically combines multiple simple tree models to optimize predictive performance. Drawing on insights and techniques from both statistical and machine learning methods (Elith et al., 2008), they not only achieve strong predictive performance, but also identify and interpret relevant variables and interactions. Interpretability of the tree-based ensemble model enables transportation decision makers to better understand the output of the model and is critical in analyzing relationship between traffic and their influential factors. In addition, the tree-based ensemble method can handle different types of predictor variables, requires little data preprocessing, and can fit complex nonlinear relationship (Elith et al., 2008). These properties make the tree-based ensemble methods good candidates in solving transportation problems, such as traffic prediction and incident classification.

However, there are limited studies on the application of tree-based ensemble methods in transportation field. Leshem and Ritov (2007) combined the random forests algorithm into Adaboost algorithm as a weak base model to predict traffic flow. The proposed algorithm is proved to be able to deal with missing data and is effective in predicting multiclass classification problems. Hamner (2010) applied random forest in travel time prediction and is one of the winners for the IEEE ICDM Contest: TomTom Traffic Prediction for Intelligent GPS Navigation. Their proposed method outperforms other models in terms of prediction accuracy. Wang (2011) applied an ensemble bagging decision tree (ensemble BDT) to predict weather impact on airport capacity and demonstrated the superior performance of ensemble BDT compared with single support vector machine classifier. Ahmed and Abdel-Aty (2013) utilized a stochastic gradient boosting method in identifying hazardous conditions based on traffic data collected from different sensors. Their study results suggested that the proposed stochastic gradient boosting method has considerable advantages over classical statistical approaches. Similarly, Chung (2013) applied boosted regression trees to study crash occurrence. Both studies utilized the boosting method to classification problems. To the best of our knowledge, research on gradient boosting tree in travel time prediction has not been fully documented to date.

This study proposes a tree-based ensemble method to predict travel time on a freeway stretch by considering all relevant variables derived from historical travel time data. Belonging to the machine learning category, the tree-based ensemble methods often have superior prediction performance over classical statistical ones. Driven by the successful application of random forest in traffic parameter prediction, a gradient boosting tree-based travel time prediction method is proposed to uncover hidden patterns in travel time data to enhance the accuracy and interpretability of the model. Different from the random forest algorithm that averages a large collection of trees from random sampling (Breiman, 2001), the gradient boosting method sequentially generates base models from a weighted version of the training data to strategically find the optimal combination of trees. Each step of adding another base model is aimed at correcting the mistakes made by its previous base models. Therefore, the gradient boosting method has the potential to provide more accurate predictions.

The rest of this paper begins with a detailed description of tree-based ensemble methods from the statistical perspective, including explanations of the single regression tree, the random forest and the gradient boosting tree or the gradient boosting method (GBM). The next section discusses the application of the GBM method in freeway short-term travel time prediction, which includes details of data preparation, model optimization, interpretation, and comparison. Discussion and conclusion are outlined at the end.

2. Methodology

The ensemble based algorithms consist of multiple base models (such as decision trees, neural networks), and each base model provides an alternative solution to the problem, whose predictions are combined in some way (typically by weighted or unweighted voting or averaging) to produce the final model output. Combining predictions of a group of individual base models often generates more stable and accurate prediction than the one provided by any of the individual base models included in the ensemble. The essential idea behind the ensemble methods is often used in our daily lives. We usually seek others' opinions when making decisions. Through weighted combination of these ideas, we can make more informed decisions. The success of ensemble methods largely depends on diversity of base models that compose the ensemble. Combining results from several base models is useful only if individual models provide different output, or in other words, they disagree with each other on some inputs. Ensemble methods reduce total error through correcting mistakes made by individual models. There is no advantage of combining models that make similar mistakes. Strategically combining individual base models that make different errors (or mistakes) can reduce the total error of the model. Diversity of individual models can be achieved through using different training datasets or using different training parameters for individual models. Bagging and boosting are two popular ensemble techniques that utilize different re-sampling methods to create diverse training data for obtaining different base models. In order to produce diverse base models despite similar training datasets, the base models are often forced to be weak. Therefore, perturbing the training data can generate different model outputs.

Trees are one type of the base model that are commonly used for ensembles. They can be very sensitive to small perturbations in the training data and a light change of the training data can lead to very different regression trees. This unique property makes them good candidates for ensembles. In addition, trees are fast and easy algorithms, which reduces computation time and complexity. Tree-based ensemble methods build a large number of different trees and then combine the results from each individual tree. The benefit of using an ensemble tree is that through averaging, the variance can be reduced. Details will be illustrated in the later section. In general, there are two successful tree-based ensemble algorithms: the random forest and the gradient boosting regression tree. Both methods use a single regression tree as their base model. The random forest method is derived based on the idea of bagging, while the theoretical foundation of the gradient boosting regression is the boosting technique. The following paragraphs will briefly explain a single regression tree and then illustrate how to construct different ensemble trees.

2.1. Single regression tree

A single tree model partitions the feature space into a set of regions and fits a simple model (a constant) for each region. For simplicity, consider a regression problem with continuous response variable Y and two independent variables X_1 and X_2. We first split the space into two regions and model the response Y (mean of Y) individually in each region. Then, we continue to split each individual region into two more regions and continue the process until some stopping rule is met. The upper panel of Fig. 1 partitions the feature space into five regions $\{R_1, R_2, R_3, R_4, R_5\}$ according to two variables X_1 and X_2 using four split-points b_1, b_2, b_3, b_4. The size of the tree is the total number of end nodes of the tree. In Fig. 1, the size of the tree is 5 as the tree is partitioned into 5 regions (end nodes). During each partition process, the best fit is achieved through the selection of variables and split-point. The lower panel of figure is a binary tree representation of the same model.

We now consider a generalized version of the above example: a regression problem consisting of p inputs with one response variable. For example, we have n observations, each observation consists of $(y_i, x_{i1}, x_{i2} \ldots x_{ij}, \ldots, x_{ip})$ for $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, p$. In terms of travel time prediction, y_i can be travel time at time step i, $(x_{i1}, x_{i2} \ldots x_{ij}, \ldots, x_{ip})$ are variables that are relevant to predicted travel time, such as historical travel time, volume, density information or other external factors. The feature space is partitioned into M regions $R_1, R_2, \ldots, R_m$. These different regions can be regarded as different traffic situations. In other words, the regression tree divides traffic conditions into categories based on the value of input parameters and model the response (dependent variable) individually for each category. Often the response for each region is treated as a constant C_m. If the optimization criterion is to minimize the sum of squares, the best C_m is just the average of y_i in region R_m (Hastie et al., 2009). In Fig. 1, we estimate a different value of C_m for each end leave (region) R_m. To decide the best splitting variable and the split-point, a greedy algorithm is implemented. For each splitting variable, the best split-point can be determined by scanning all possible values, which can be done very quickly. By scanning through all input variables, finding the best pair of splitting variable and split-points is feasible. A single regression tree is the basic model for random forest and gradient boosting regression tree methods.

2.2. Random forest

Random forest was developed by Breiman (2001) in 2001. It combines two powerful machine-learning techniques: Breiman's "bagging" idea (Breiman, 1996) and the random features selection introduced by Ho (1998, 1995) and Amit and Geman (1997). To understand how random forest works, the following paragraphs explains the concept of bagging and random feature selection.

In bagging or bootstrap aggregation, each individual based model is trained on the bootstrap sample from the training data. The bootstrap techniques were originally developed to estimate sampling distribution of an estimator from limited data by sampling with replacement from the original data. In recent development of ensemble techniques, bootstrap has

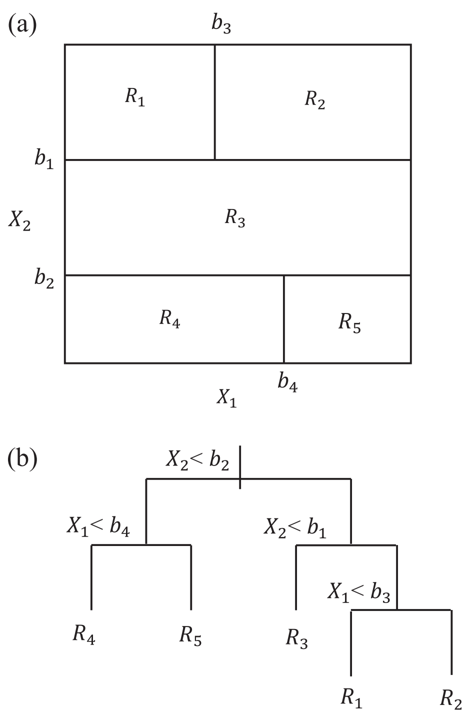


Fig. 1. Single regression tree.

been used to generate a diverse subset of data for training base models. For a given training data set with sample size n, bagging generates k new training set, each with sample size n, by sampling from the original training data set uniformly and with replacement. Through sampling with replacement, some observations appear more than once in the bootstrap sample, while other observations will be 'left out' of the sample. Then, k base models are trained using the newly generated k training set and combined through averaging (regression problem) or majority voting (classification problem).

To guarantee the success of the bagging method, the individual base model should have the property of instability. Bagging improves prediction accuracy through diversity of each base models generated from a perturbed training set. To obtain diverse base models from similar training data sets, the base model should be weak. Here, 'weak' means a model that performs slightly better than random guessing. Using a tree as the base model, we can obtain perhaps the simplest and easiest ensemble tree, the bagged tree. Each tree in the ensemble is grown on data samples that were randomly drawn with replacement from the original data. However, with lots of data, we usually learn the same regression tree. Averaging output of these trees does not improve prediction accuracy.

The second technique that random forest utilized is random feature selection. Random forest is a further development of the bagged regression tree. It is still based on the bootstrapped sampling to grow individual trees. Instead of using all features, it only allows a random subset of features at each splitting node of the tree. Therefore, it enforces diversity between base models. Fig. 2 illustrates the basic steps for random forest.

Random forest improves forecasting accuracy through variance reduction by averaging many noisy but approximately unbiased trees. According to (Hastie et al., 2009), the variance of a random forest with total number (K) of trees is:

$$\rho\sigma^2 + \frac{1-\rho}{K}\sigma^2 \tag{1}$$

where σ^2 indicates the variance of individual tree, ρ denotes correlation between the trees, and M is the total number of trees in the ensemble. It is obvious that by increasing the total number of trees M, the second term tends to be zero. Therefore, the variance of a random forest depends on three things:

(1) The correlation ρ between any pair of trees: Decreasing the correlation decreases the total variance. This can be achieved by: randomly selecting v out of the p variables to split at each splitting node when growing a tree on a bootstrapped dataset. Reducing v, reduces both the correlation between trees and the strength of individual tree, and vice versa. Therefore, there is a need to find the optimal value of v for certain dataset.
(2) The variance σ^2 of each individual tree, or in other words, the strength of each individual tree: Strengthening the performance of each individual tree can decrease the total variance of the model.
(3) The total number of trees M: The second term of the equation can be reduced by increasing M. Therefore, we should train adequate number of trees to make sure the second term of the equation goes to zero.

```
    Given a data set with total number of data sample $n$ and $p$ input variables. Initially
determine the total number ($M$) of trees to be generated and the number $v < p$ of variables used
for each individual tree:
      For $m = 1\ to\ M$ do
            Draw a random sample $S^*$ of size $n$ with replacement from the original training
            data (This is also referred to as the bootstrap sample. This sample will be the
            training data to grow the tree.);
            Grow a tree $T_m$ using the training sample $S^*$ through the following loop:
            Do until (the minimum node size $nod_{min}$ is reached)
                  For the terminal node of the tree;
                  Randomly select $v$ variables out of the $p$ variables;
                  Select the best pair of split variable/point among the $v$ variables;
                  Split the node into two daughter nodes;
            End;
            Output the constructed tree $T_v(x)$;
    End;
    Output the prediction of the ensemble trees for a given new input $x$: $\frac{1}{m}\sum_{m=1}^{M} T_m(x)$;
```

[note] the minimum node size $\boldsymbol{nod_{min}}$ *is the minimum number of observations in each end node.*

Fig. 2. Pseudo-code for random forest.

In general, random forest is based on the idea of bagging but enforces diversity of each individual tree through random feature selection. The theoretical background of random forest supports parallel computing therefore its training speed can be accelerated through parallel computing. The prediction performance of the random forest is influenced mainly by three factors: correlation between individual trees, the performance of each tree and the total number of trees.

2.3. Gradient boosting regression tree

Different from bagging, the boosting method generates base models sequentially. Prediction accuracy is improved through developing multiple models in sequence by putting emphasis on these training cases that are difficult to estimate. In the boosting process, examples that are difficult to estimate using the previous base models appear more often in the training data than the ones that are correctly estimated. Each additional base model is aimed to correct the mistakes made by its previous base models. The birth of the boosting method is from the answer (Schapire, 1990) to Kearns' question (Kearns, 1988): Is a set of weak learners equivalent to a single strong learner? A weak leaner is an algorithm that performs only slightly better than random guessing; a strong base model is a more accurate prediction or classification algorithm that is arbitrarily well correlated with the problem. The answer to this question is very important. It is often easier to estimate a weak model compared with a strong model. Schapire (Kearns, 1988) proves that the answer is positive by applying boosting algorithms (Fig. 3) to combine many weak models into a single and high accurate model.

The major difference between bagging and boosting methods is that the boosting method strategically resamples the training data to provide the most useful information for each consecutive model. The adjusted distribution during each step of training is based on the error produced by the previous models. Unlike the bagging method where each sample is uniformly selected to produce a training dataset, the probability of selecting an individual sample is not equal for the boosting algorithm. Samples that are misclassified or incorrectly estimated have more chances to be selected with higher weight. Therefore, each newly created model places emphasis on the samples that have been misclassified by previous models.

Boosting fits additional models that minimize a certain loss function averaged over the training data, such as a squared-error or an absolute error. The loss function measures the amount the predicted value deviates from the true value. One of the approximate solutions to this problem is by using a forward stage-wise modeling approach. The forward stage-wise approach sequentially adds new base models without changing parameters and coefficients of models that have already been added. In terms of regression problem, the boosting method is a form of "functional gradient decent". It is an optimization technique that minimizes a certain loss function by adding a base model at each step that best reduces the loss function. The pseudo code for the generic gradient boosting method is as follows (Friedman, 2001; Hastie et al., 2009):

Friedman proposed a modification to the gradient boosting method by using a regression tree of fixed size as the base model. The modified version improves the quality of the model (Friedman, 2002). In this study, this modified version of the GBM model is used for travel time prediction. Assuming that the number of leaves for each tree is J. Each tree partitions the input space into J disjoint regions $R_{1m}, R_{2m}, \ldots, R_{jm}$ and predicts a constant value b_{jm} for region R_{jm}. The regression tree can be formally expressed as:

$$g_m(x) = \sum_{j=1}^{J} b_{jm} I(x \in R_{jm}) \tag{2}$$

```
Given a data sample distribution $D$. Determine the total number of base models as $M$:
Define the initial training sample distribution as $D_1 = D$
For $m = 1$ to $M$ do
    Train a base model $B_m(x)$ from the training sample distribution $D_m$.
    Compute the error of the model.
    Adjust the distribution $D_m$. to $D_{m+1}$. to make the mistake of the model more
    evident.
    Output the constructed base model $B_m(x)$.
End;
Output the prediction of the ensemble trees for a given new input $x$: $\frac{1}{M}\sum_{m=1}^{M} B_m(x)$;
```

Fig. 3. The boosting algorithm.

where $I(x \in R_{jm}) = \begin{cases} 1, & \text{if } x \in R_{jm} \\ 0, & \text{otherwise} \end{cases}$

Using a regression tree to replace $g_m(x_i)$ in the generic gradient boosting method, the model updating equation and gradient descent step size (Fig. 4):

$$f_m(x) = f_{m-1}(x) + \rho_m g_m(x) \tag{3}$$

$$\rho_m = argmin_\rho \sum_{i=1}^{n} L(y_i, f_{m-1}(x_i) + \rho g_m(x_i)) \tag{4}$$

becomes

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} \rho_m b_{jm} I(x \in R_{jm}) \tag{5}$$

$$\rho_m = argmin_\rho \sum_{i=1}^{n} L\left(y_i, f_{m-1}(x_i) + \sum_{j=1}^{J} \rho b_{jm} I(x \in R_{jm}) \right) \tag{6}$$

Using a separate optimal ρ_{jm} for each region R_{jm}, b_{jm} could be discarded. The model updating rule (Eqs. (4) and (5)) becomes:

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} \rho_{jm} I(x \in R_{jm}) \tag{7}$$

$$\rho_{jm} = argmin_\rho \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \sum_{j=1}^{J} \rho I(x \in R_{jm}) \right) \tag{8}$$

The gradient boosting regression tree builds the model in a stage-wise fashion and updates the model by minimizing the expected value of certain loss function. With many trees added to the model, the fitted model may achieve an arbitrarily small training error. However, fitting the model too closely to the training data can lead to poor generalization ability. By

```
Initialize $f_0(x)$ to be a constant, $f_0(x) = argmin_\rho \sum_{i=1}^{N} L(y_i, \rho)$.
For $m = 1$ to $M$ do
    For $i = 1$ to $n$ do
        Compute the negative gradient
            $z_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$
    End;
    Fit a regression tree $g_m(x)$ to predict the targets $z_{im}$ from covariates $x_i$ for all training
    data.
    Compute a gradient descent step size as
        $\rho_m = argmin_\rho \sum_{i=1}^{n} L(y_i, f_{m-1}(x_i) + \rho g_m(x_i))$
    Update the model as
        $f_m(x) = f_{m-1}(x) + \rho_m g_m(x)$
End;
Output the final model    $f_M(x)$
```

Fig. 4. Pseudo-code for generic gradient boosting.

increasing the number of iterations, the model becomes complex and minor fluctuations in data will be exaggerated. This leads to poor prediction performance on unseen data (testing data). It is necessary to determine the optimal number of iterations (or the number of trees) M to minimize future risks associated with prediction. Over-fitting can be prevented through controlling the number of gradient boosting iterations, or more effectively, scaling the contribution of each tree by a factor of $J \in (0, 1]$. This implies changing the model updating Eq. (6) to

$$f_m(x) = f_{m-1}(x) + J \cdot \sum_{j=1}^{J} \rho_{jm} I(x \in R_{jm}) \tag{9}$$

Parameter J, referred to as learning rate, controls the contribution of each base model by shrinking its contribution with a factor $0 < J \leqslant 1$. There is a tradeoff between the number of iterations and the learning rate. With the same number of iterations, a smaller value of learning rate tends to lead to a higher training risk. Smaller value of$J$ requires a larger number of M to obtain the same training risk. In general, a small J ($J < 0.1$) with a large$M$ is preferable.

Another parameter, tree complexity, also influences performance of the algorithm. Tree complexity refers to the number of nodes in a tree. The optimal size of each tree can be estimated separately when building the ensembles. By simply assuming each tree is the last one in the model, we usually expect very large trees, especially during the early iterations. This is a very poor assumption and potentially degrades the model performance and increases computation complexity. One simple solution is to restrict all trees to be the same size C. Therefore, for the entire process, we only need to determine one value of C to optimally estimate the data. Gains from increased C are greater with larger data sets. As large data sets provide more detailed information about the problem, increasing the value of C would capture complex variable interactions in data. The tree size C constrains the interaction level of each model. Namely, $C - 1$ is the maximum level of interaction effects for a tree with size$C$. Therefore, the size of the trees reflects the maximum depth of variable interactions.

The GBM model strategically adds each base model to minimize a certain loss function. It uses a stage-wise sampling strategy, which put more emphasis on samples that are difficult to be estimated. This distinguishes itself from random forest that trains each base model from random sampling with replacement and equal probability. Performance of the GBM model is influenced by the number of trees (M), learning rate (J) and variable interactions (C). Optimal performance of the model can be achieved through carefully selecting the best combination of these parameters. Interested readers could also refer to the tutorial (Natekin and Knoll, 2013) for detailed discussion of the GBM methods.

3. Application to travel time prediction

Travel time prediction is through modeling correlations of different parameters based on available traffic information. Accurate prediction relies on how much information we could extract from these available data. As traffic is often a complex phenomenon that involves non-linear and chaos characteristics, it is often difficult to use an exact equation to represent this phenomenon. Data driven approach becomes a promising area in modeling and predicting traffic. In recent years, tree-based ensemble methods have shown very promising results in prediction field. Developing tree-based ensemble method in travel time prediction can potentially improve prediction accuracy. This section discusses in details on how to apply GBM model in travel time prediction.

3.1. Data description and preparation

Real-word travel time data provided by a private-sector company, INRIX, are used for this study. INRIX derives travel times from its smart driver network, which aggregates traffic data from probe vehicles and traditional sensor sources. Probe vehicles utilized include: taxis, airport shuttles, service delivery vans, long-haul trucks, consumer vehicles, and GPS enabled consumer smartphones and so on. Traffic sensors range from inductive-loop detectors, radar sensors, to toll tag readers. The data fusion methods are proprietary and travel times are reported on TMC segments. The first experiment utilized travel time data from five TMC segments located along I-95 southbound in Maryland. Table 1 shows location information for the five selected TMCs, which includes corresponding TMC code, start and end location, and length of each segment. Fig. 5 is the location of the selected five segments (Different color represent different TMC section). Travel time data observed in 2012 were downloaded from the Regional Integrated Transportation Information System (RITIS) website (Laboratory), and was aggregated into every five minutes time interval. The quality of the data is excellent with less than a 1% missing rate; 561 out of 105408 observations are missing for most segments. Given the small amount of missing values, this study simply replaced the missing values with the mean of its closest surrounding values.

Table 2 summarizes the basic statistics of collected travel time data in 2012, including: mean value, standard deviation (SD), the 25th, 50th, 75th and 95th percentiles of travel time, minimum (min), and maximum (max) observations. Travel time data is recorded in the unit of minutes. To prepare the input data for the model, we considered all possible variables that are relevant to future travel time. This led to ten input variables for the prediction model as indicated in Table 3.

Table 3 is an example of the data set (training and testing set) used for our study. The first ten columns are the input variables and the last column is the corresponding output of the GMB/RF models. The output of the model is travel time at time lag t denoted as TT_t. The ten variables that are used as input to predict travel time at time step t are as follows: TT_{t-1}, TT_{t-2}, TT_{t-3} are three most recent travel time observations at time steps $t-1$,$t-2$, and$t-3$, $\Delta\mathrm{TT}_{t-1} = \mathrm{TT}_{t-1} - \mathrm{TT}_{t-2}$ is the growth

Table 1
Selected freeway segments for the study.

Section	TMC	Start		End		Miles
		Latitude	Longitude	Latitude	Longitude	
I	110-04421	39.218482	−76.726905	39.200843	−76.760999	2.2
II	110N04421	39.200843	−76.760999	39.192756	−76.771665	0.8
III	110-04420	39.192756	−76.771665	39.182237	−76.78368	0.97
IV	110N04420	39.182237	−76.78368	39.175368	−76.794578	0.75
V	110N04419	39.160086	−76.823761	39.156238	−76.834836	0.66

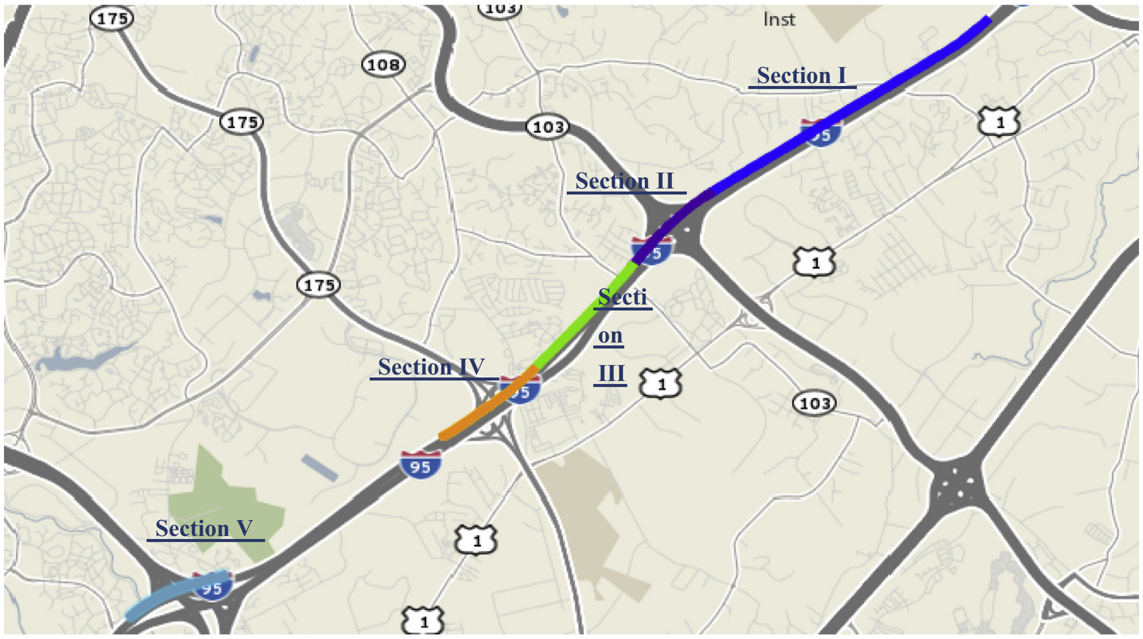


Fig. 5. Locations of TMC segments for the study (I-95).

Table 2
Basic statistics of travel time data (min).

Section	Mean	SD	25th	50th	75th	95th	Min	Max
I	2.01	0.52	1.92	1.96	2.01	2.17	1.77	26.41
II	0.73	0.19	0.69	0.71	0.73	0.78	0.65	9.66
III	0.89	0.21	0.85	0.87	0.89	1.00	0.78	22.42
IV	0.70	0.24	0.66	0.67	0.69	0.74	0.61	9.06
V	0.60	0.15	0.58	0.59	0.61	0.63	0.53	7.90

rate over two consecutive time steps $t-1$ and$t-2$, time of day is represented by every five minute time step indexed from 1 to 288, week is indexed from 0 to 6 to represent fromSunday to Saturday, day is the day when the observation is detected (from 1 to 31), and month is the month information for the observation (from 1 to 12).

3.2. Model optimization

To optimize the model, it is critical to know the effect of different combinations of parameters on the model's performance. Based on this information, we can then select the optimal parameters to achieve a lower prediction error. This section demonstrates how performance varies with different choices of parameters (number of trees M, learning rate J and interaction C) by using two months' traffic data as training data and the following seven days' data as testing data. Using traffic data from freeway segment one, we fitted GBM models with various numbers of trees (1–8000), learning rates (0.5–0.0005) and variable interactions (1–4). Figs. 6–8 show the influence of different parameters (M, J, and C) on the prediction errors. In these plots, we use mean absolute percentage error (MAPE) to represent prediction error. The definition of the MAPE is:

Table 3
Example of the training/testing data file (inputs and output of the GBM/RF models).

TT_{t-1}	TT_{t-2}	TT_{t-3}	ΔTT_{t-1}	ΔTT_{t-2}	ΔTT_{t-3}	Time of day	Day	Week	Month	TT_t
1.88	1.86	1.94	0.02	−0.08	−0.07	283	1	0	1	1.88
1.88	1.88	1.86	0	0.02	−0.08	284	1	0	1	1.87
1.87	1.88	1.88	−0.01	0	0.02	285	1	0	1	1.89
1.89	1.87	1.88	0.02	−0.01	0	286	1	0	1	1.9
1.9	1.89	1.87	0.01	0.02	−0.01	287	1	0	1	1.93
1.93	1.9	1.89	0.03	0.01	0.02	288	1	0	1	2.01
2.01	1.93	1.9	0.08	0.03	0.01	1	2	1	1	2.01
2.01	2.01	1.93	0	0.08	0.03	2	2	1	1	2.01
2.01	2.01	2.01	0	0	0.08	3	2	1	1	1.81
1.81	2.01	2.01	−0.2	0	0	4	2	1	1	1.77
1.77	1.81	2.01	−0.04	−0.2	0	5	2	1	1	1.99
...	...	...	...	...	...	...	...	...	...	...

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{t(i) - a(i)}{t(i)}\right| \tag{10}$$

where $t(i)$ is the actual value, $a(i)$ is the forecast value, and n is the total number of data intervals.

To study the effect of parameter M on prediction accuracy, Fig. 6 plots the relationship MAPE and M (with different value of J and C). The parameter M indicates how many base models are included in the ensemble. In terms of estimation, arbitrary accuracy can be achieved through increasing M. But with too many trees, over-fitting may occur, which affects prediction

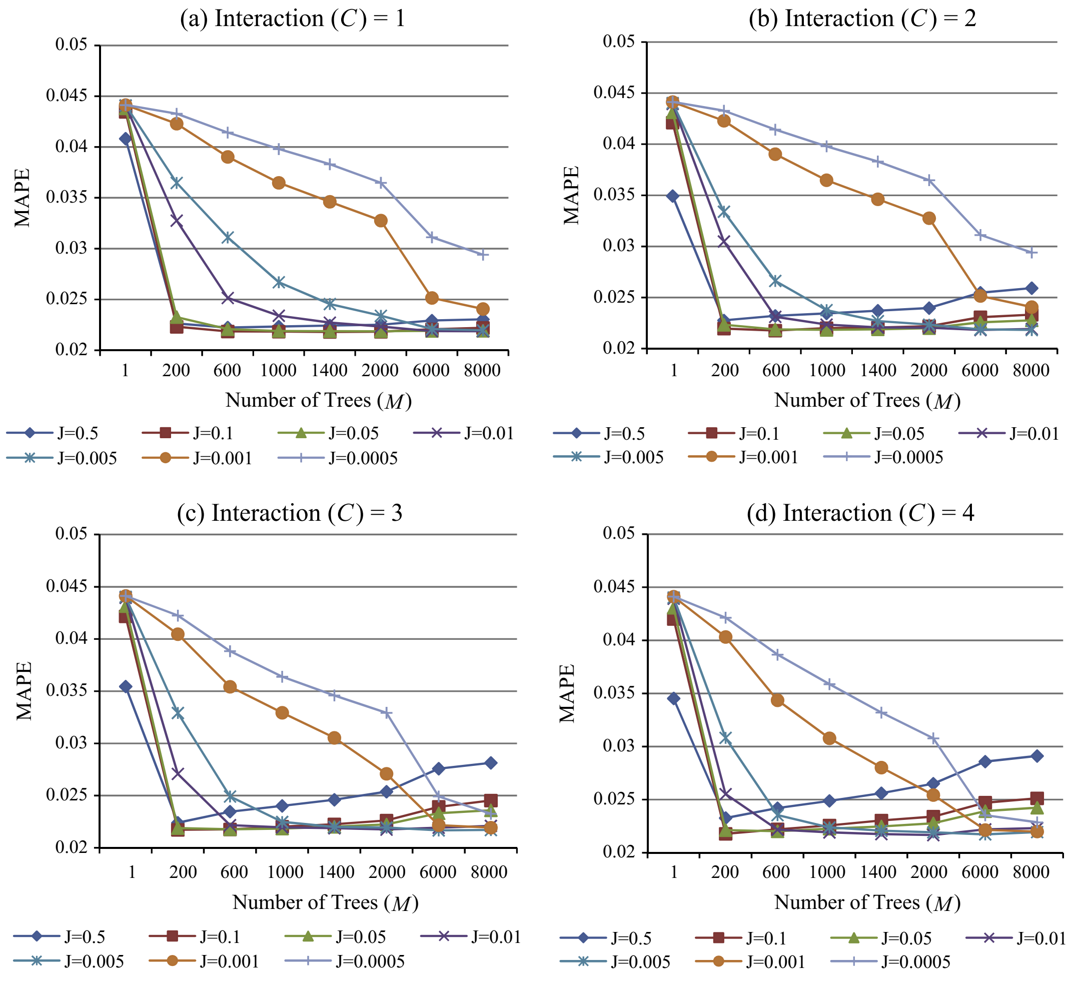


Fig. 6. The relationship between MAPE and number of trees (M) for models fitted with seven learning rates (J) and four levels of interactions (C).

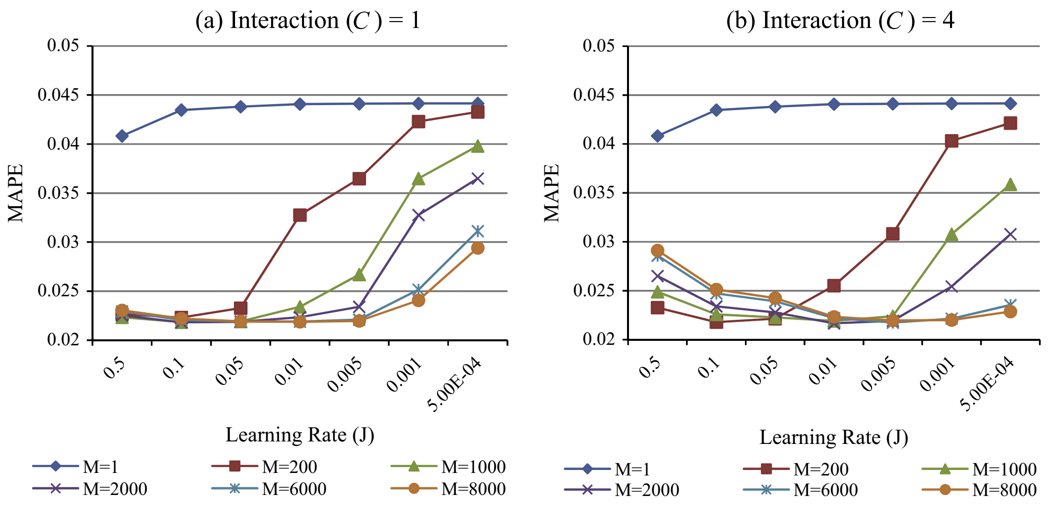


Fig. 7. MAPE against learning rate (J) for models fitted with various numbers of trees (M) and different levels of interactions (C).

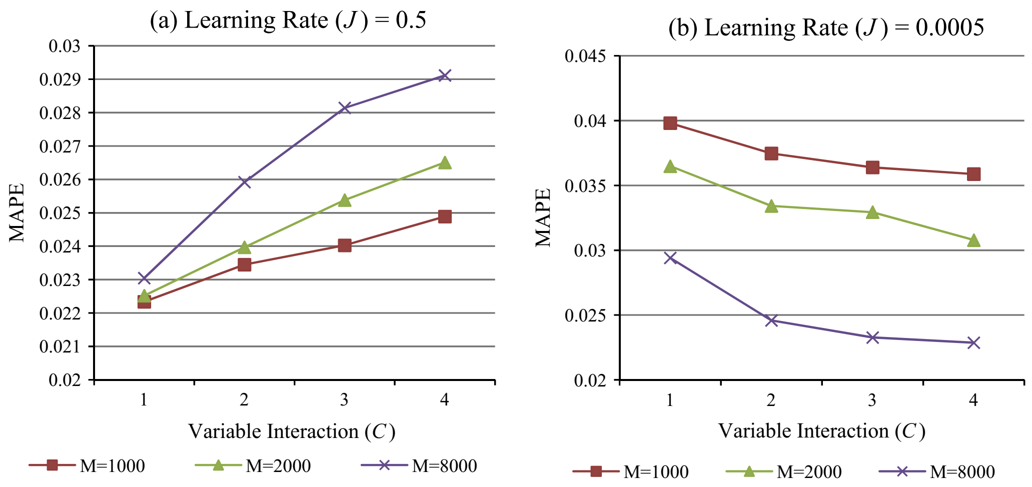


Fig. 8. MAPE against tree complexity for models fitted with various numbers of trees (M) and different learning rate (J).

performance on 'unseen data' (samples not included in the training data set). In general, Fig. 6a indicates that MAPE value decreases until to a certain value as M increases with $C = 1$. The slopes of the lines are different for different value of$J$. The line with $J = 0.0005$ has very smooth slope as the contribution of each additional tree becomes very limited with a small learning rate. On the other hand, a higher learning rate, such as$J = 0.5$, can be too fast so that it reaches its minimum error with$M = 600$. To continue increase$M$ will increase prediction error, where over-fitting occurs. This effect becomes even more obvious if we allow more variable interactions, which makes the individual tree more complex. As shown in Fig. 6b–d, higher learning rates (such as $J = 0.5$, $J = 0.01$, $J = 0.05$) reach their best prediction performances with relatively fewer numbers of trees ($M = 200$) and can easily over-fit the model if more trees are included. In general, we should guarantee enough trees to model the complexity of the data and, at the same time, prevent over-fitting with too many trees.

Fig. 7 plots the effect of learning rate on MAPE value. Learning rate adjusts the contribution of each additional tree with a factor J, $0 < J < 1$. A smaller value of$J$ limits the contribution of each tree in the model and often requires more trees to be added. Depending on the complexity and the number of trees in the ensemble, the optimal value of J can be different. Taking Fig. 7a for example, with $M = 1000$, prediction error (MAPE) increases if we continue to decrease$J$ after a certain level ($J = 0.05$). This is because the contributions of individual trees become very limited if$J$ is below a certain level and the current value of M becomes insufficient. In order to obtain better prediction results, M should be increased with a decreased value of J. On the other hand, using higher value of J leads to fewer trees being required to achieve better performance. But a higher J usually cannot achieve the minimum error. In Fig. 7b, $J = 0.5$ is fitted with relatively fewer trees, but did not achieve an error as small as the one with$J = 0.1$. With a higher value of$J$, increasing M leads to poorer prediction performance (Fitting $M = 8000$

with$J = 0.5$), because over-fitting occurs. Therefore, a smaller learning rate with a larger number of trees is preferable, but computational time wouldalso be increased with many trees being fitted. There needs to be a balance between computational time and prediction accuracy.

Tree complexity, or variable interaction (C), also influences model performance as shown in Fig. 8. With $J = 0.5$ (Fig. 8a) and many trees being fitted, the MAPE value increases as C increases. This increasing rate of MAPE value (the slope of the lines) becomes more obvious with higher M value. This is because a higher J value makes each individual base model share a higher contribution to the model output, with many trees fitted and more complex individual trees, over-fitting easily occurs. By reducing $J = 0.0005$ (Fig. 8b), the MAPE value decreases as C increases. This is partly due to the fact that the detailed information from the data can be modeled with a higher level of variable interaction. So, a smaller learning rate restricts each additional tree's contribution to the model and prevents over-fitting.

In general, according to the experiment result, we can concludes that: (a) A slower learning rate with a larger number of trees in the model is preferable to a faster learning rate with a smaller number of trees. A slower learning rate shrinks the contribution of each tree more and therefore allows smoother approach to the optimal performance and provides more reliable prediction results. (b) There is also a need to consider the tradeoff between prediction accuracy and computational time. Since a large number of trees are being fitted, model complexity also increases and requires more computational time. (c) In addition, the level of variable interaction also affects the optimal selection of learning rate and number of trees. A higher level of variable interaction leads to a more complex model and requires fewer trees to be fitted with a given learning rate.

3.3. Model interpretation

Inputs of the model, or predictor variables, usually have very different influences on the output (response variable). To explore the individual input variable's influence on the response variable, we can gain better insight into the data. Breiman (1984) proposed a method to measure the relative influence of each predictor variable on the model output for a single decision tree. This relative influence of a predictor variable is measured based on the number of times this variable is selected to split a currently terminal region (or node) into two sub-regions, weighted by the least-squares improvement for the model as a result of this split. Friedman (2001) generalized this criterion to additional tree expansions by simply averaging this criterion over all trees. The relative influence of each individual variable is scaled so that the sum of them for all the input variables equals to 100. A higher value indicates stronger influence of the input variable to the model.

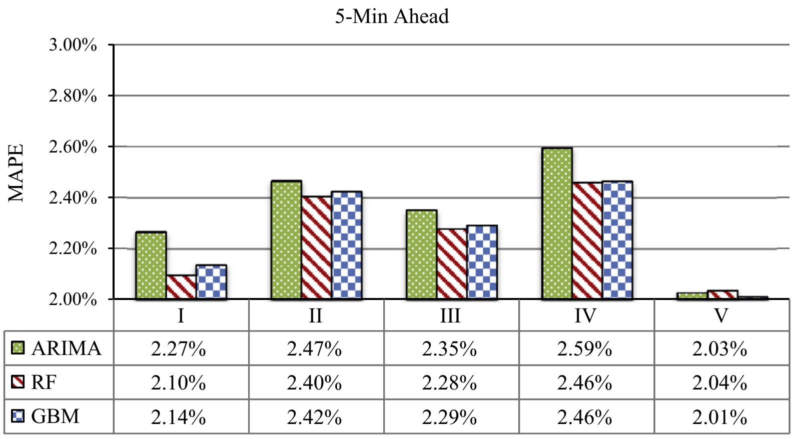
Table 4 gives the relative influence of each input variable to the model output with different prediction horizons. It is obvious that each input variable contributes differently to the response (output of the model). For all three cases, the immediate previous travel time TT_{t-1} contributes the most to the predicted travel time. This is expected, as the immediate previous traffic condition will influence traffic in the near future. Therefore, it is closely related with future travel time. The changing rate of travel time $\Delta\mathrm{TT}_{t-1}$ also has higher influence on the model output. $\Delta\mathrm{TT}_{t-1}$ is the difference of travel time between every two consecutive time steps. It indicates the changing behavior of traffic. For example, a positive value of $\Delta\mathrm{TT}_{t-1}$ indicates the increasing trend of travel time and higher positive values are highly correlated with congestion. Another interesting result indicated in Table 4 is that the influence of time of day becomes more significant when we increase the prediction horizon. For a prediction 30 min ahead, the influence of the time of the day variable contributes 17.43% to the model output. The time of day variable is associated with the periodic feature of travel time: travel time usually increases during peak hours and maintains at a certain value during non-peak hours. With prediction horizon increases, immediate past traffic information cannot be provided and the impact of the most recent available travel time become less significant. More information can be obtained through the level of travel time during certain time interval. Therefore, contribution of the time of day variable increases with increased prediction horizon.

Since TT_{t-1} and $\Delta\mathrm{TT}_{t-1}$ are two most important variables for one-step-ahead prediction, we plot their joint influence on model output (Fig. 9). The x-axis defines the lag one difference $\Delta\mathrm{TT}_{t-1}$, the y-axis is the lag one travel time TT_{t-1}, and the z-axis represents the predicted travel time $\widehat{\mathrm{TT}}_t$. As indicated in this figure, there is a positive correlation between the lag one

Table 4
Relative influence of input variables for GBM models with learning rate of 0.001 for multistep-ahead prediction.

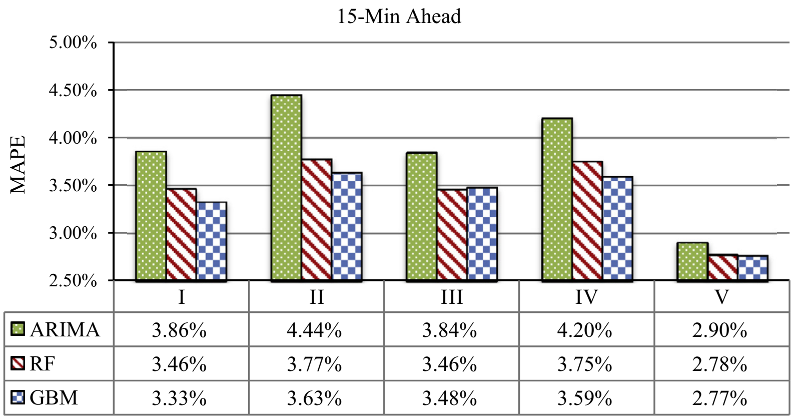
Variable	5-Min (1-step) ahead		15-Min (3-step) ahead		30-Min (6-step) ahead	
	Rank	Relative importance (%)	Rank	Relative importance (%)	Rank	Relative importance (%)
TT_{t-1}	1	96.34	1	81.47	1	66.33
TT_{t-2}	3	0.26	7	0.48	8	0.89
TT_{t-3}	7	0.04	9	0.27	7	1.63
$\Delta\mathrm{TT}_{t-1}$	2	2.95	3	6.29	3	7.61
$\Delta\mathrm{TT}_{t-2}$	9	0.02	6	0.51	9	0.1
$\Delta\mathrm{TT}_{t-3}$	6	0.09	8	0.27	10	0.08
Time of day	4	0.16	2	7.35	2	17.43
Day	10	0	10	0.26	6	1.65
Week	5	0.1	5	1.05	5	2.04
Month	8	0.04	4	2.06	4	2.24

Table 5
Comparison of 5 min ahead prediction for ARIMA, RF and GBM.



	I	II	III	IV	V
🟩 ARIMA	2.27%	2.47%	2.35%	2.59%	2.03%
▧ RF	2.10%	2.40%	2.28%	2.46%	2.04%
▦ GBM	2.14%	2.42%	2.29%	2.46%	2.01%

Table 6
Comparison of 15 min ahead prediction for ARIMA, RF and GBM.



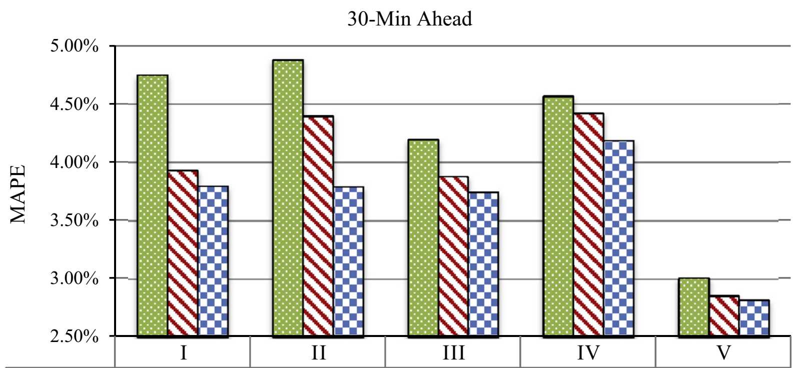
	I	II	III	IV	V
🟩 ARIMA	3.86%	4.44%	3.84%	4.20%	2.90%
▧ RF	3.46%	3.77%	3.46%	3.75%	2.78%
▦ GBM	3.33%	3.63%	3.48%	3.59%	2.77%

travel time TT_{t-1} and the predicted travel time $\widehat{TT}_t$. Traffic conditions are often consistent within a short period. If congestion occurs, it will last for several minutes or even hours. Therefore, a high value of travel time at time step $t-1$ would more likely followed by another high value of travel time. The influence of travel time difference ΔTT_{t-1} on $\widehat{TT}_t$ is more evident if ΔTT_{t-1} is a positive value. In other words, if previous travel time continues to increase, the future travel time will also tend to increase. But, if the previous travel time decreases or maintains a certain value, the difference of travel time ΔTT_{t-1} will have less impact on future travel time. In general, higher values of both TT_{t-1} and ΔTT_{t-1} often indicate that congestion occurs. Therefore, they are more likely to be followed by a higher travel time value.

3.4. Model comparison

To test the effectiveness of the GBM model, this section comprehensively evaluates prediction performance of the ARIMA, the RF and the GBM models. The ARIMA model is one of the widely recognized benchmark models for traffic parameters forecasting. Prediction is based on regression of its current and past values. A non-seasonal ARIMA model is classified as an ARIMA (p, d, q) model, in which p is the number of autoregressive terms, d is the number of non-seasonal differences, and q is the number of lagged forecast errors. Optimization of the ARIMA model involves order selection and parameter

Table 7
Comparison of 30 min ahead prediction for ARIMA, RF and GBM.



	I	II	III	IV	V
🟩 ARIMA	4.75%	4.88%	4.20%	4.57%	3.01%
▨ RF	3.93%	4.39%	3.88%	4.42%	2.85%
▦ GBM	3.80%	3.79%	3.74%	4.19%	2.82%

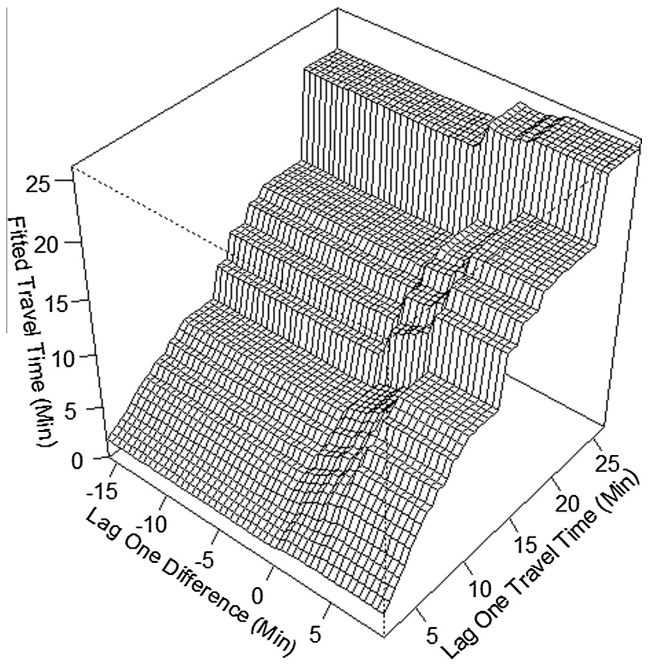


Fig. 9. Three-dimensional plots for the joint effects of lag one difference and lag one travel time on predicted travel time value.

estimation. Detailed information on theoretical background and steps in fitting an ARIMA model can be found in (Tsay, 2010).

We use two months' training data and seven days' testing data to compare these three models. Prediction accuracy of the three models are compared on the basis of 5-min (1-step-ahead), 15-min (3-step-ahead) and 30 min (6-step-ahead) ahead prediction. We test different combinations of variables for both the GBM and the RF models during the training process and select the best model according to their MAPE values. The best orders of the ARIMA model are selected based on the method proposed by Hyndman and Khandakar (2007), and the parameters of the ARIMA model are estimated based on maximum likelihood method. Table 5 through Table 7 compare these three models' prediction performance based on MAPE value. Both the RF and the GBM model outperform the ARIMA model in the one-step-ahead prediction as shown in Table 5. The GBM and the RF perform similarly, with RF has slightly lower MAPE value. As the prediction horizon increases, all three

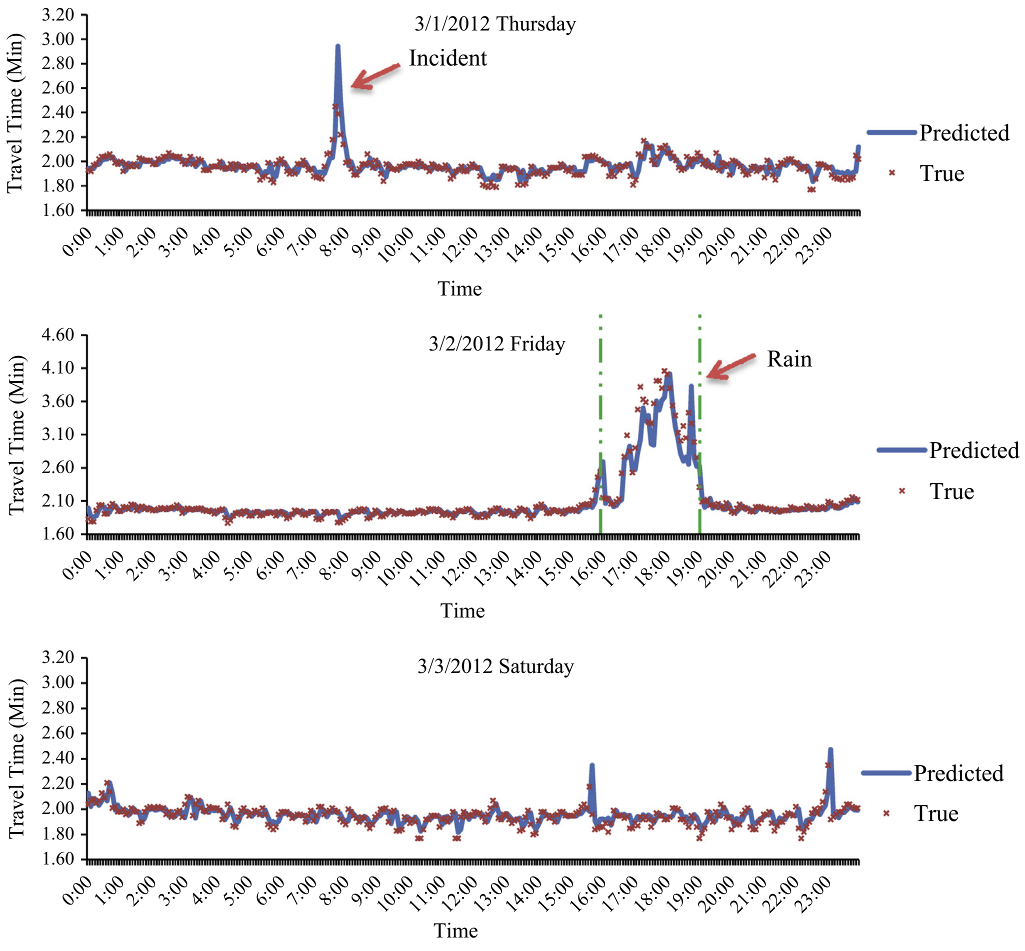


Fig. 10. Sample travel time prediction results of the GBM method (5-min ahead prediction).

models' performances drop. Comparatively, the GBM method is less sensitive to prediction horizon and maintains a good prediction performance. As indicated in Tables 6 and 7, the GBM model outperforms both ARIMA and RF models, 9 out of 10 cases. In general, the RF and the GBM methods perform better than the ARIMA model in the one-step-ahead prediction. By increasing prediction horizons, the difference among these three models becomes obvious, with the GBM model being the most accurate, compared with the RF and the ARIMA models. Therefore, we conclude that both the RF and the GBM models are promising algorithms in travel time prediction as they are more accurate compared with the ARIMA model. The advantage of the GBM model becomes even obvious in multi-step-ahead predictions.

Fig. 10 shows three days' forecasting results provided by the GBM model. The blue line stands for the predicted travel time by the GBM model, the red cross stands for the true (original) value of travel time. We could see that the overall performance of the GBM model is very good not only in normal traffic conditions (lower panel of Fig. 10), but also is very effective during traffic transitional period. On 3/1/2012 (upper panel of Fig. 10), there are three recorded incidents between 6:31 and 7:16, which are possible reasons for the traffic state change from uncongested to congested. The GBM model is able to capture this sudden change. On 3/2/2012 (middle panel of Fig. 10), rain begins falling shortly after 16:00 and ends at 19:00. Congestion occurs during this period and travel time increases. The GBM model also adequately captures this congestion. Theoretically, the GBM model is able to handle complex interactions among input variables and can fit complex nonlinear relationship. Therefore, the GBM model is able to model non-linear characteristics of dynamic traffic systems and leads to superior prediction performance.

3.5. Prediction under congested conditions

To further test the GBM model's performance, an additional experiment is designed to compare the performance of the GBM, RF and ARIMA models in both congested and uncongested situations using longer freeway segments. Three freeway segments located along southbound of MD 295 in Maryland are selected for this experiment as shown in Fig. 11. Each

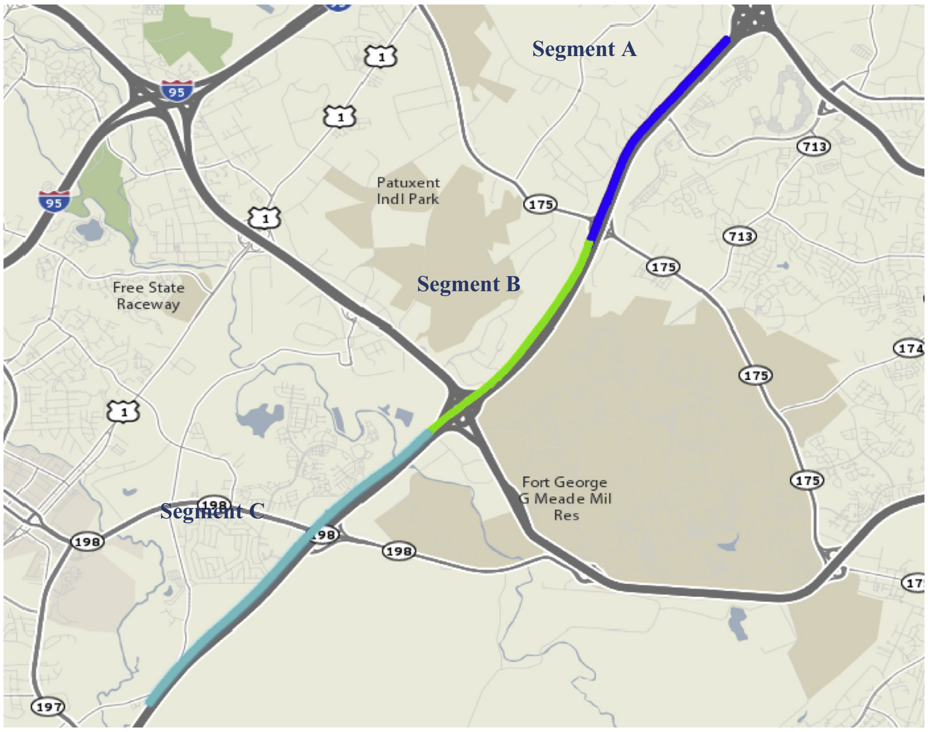


Fig. 11. Study segments (MD 270).

Table 8
Statistics of peak and non-peak hour travel time.

Segment	Peak		Non-peak	
	Mean	Standard deviation	Mean	Standard deviation
A	217.67	65.52	126.63	5.66
B	335.94	210.82	141.09	40.49
C	566.49	308.19	246.49	82.81

Table 9
Comparison of ARIMA, RF, GBM model during peak hours based on MAPE criterion.

Model	Segment	1-step (%)	2-step (%)	3-step (%)	4-step (%)	5-step (%)	6-step (%)
GBM	A	11.26	**16.31**	19.93	**22.11**	**23.56**	**26.06**
	B	**8.68**	**13.95**	**18.11**	**21.06**	22.67	23.22
	C	**9.84**	**15.26**	**20.40**	**23.38**	**26.15**	**28.42**
RF	A	12.20	16.85	**19.87**	25.77	24.72	30.01
	B	10.78	16.37	19.02	22.13	**21.97**	**22.95**
	C	12.89	18.22	22.01	24.75	27.56	30.87
ARIMA	A	**10.98**	16.60	21.57	23.52	25.85	29.27
	B	9.55	15.48	19.57	23.00	24.59	24.75
	C	10.12	17.26	23.89	27.80	29.09	29.54

The bold and underline values are the smallest MAPE values within each segment.

segment is covered by multiple TMCs: segment A (TMC110-04498, TMC 110N04498), segment B (TMC 110-04497, TMC 110N04497, TMC 110-04496, TMC 110N04496), segment C (TMC 110-04495, TMC 110N04495, TMC 110-04494). The lengths of the three segments are 2.28 mile, 2.37 mile and 3.84 mile respectively. Based on traffic information collected from these nine TMCs, path travel time for the three segments of study can be estimated. Details of calculating the path travel time for multiple TMC segments can be found in Hamedi et al. (Ali Haghani and Sadabadi, 2010). Then, the travel time information for each segment is aggregated into five minutes time interval. Table 8 summarized the basic statistics of these three segments during both peak and nonpeak hours. For this studied segment, peak hours are from 6–9 am (6:00–9:00) and 5–7 pm (17:00–19:00), the rest are considered as nonpeak hours.

Table 10
Comparison of ARIMA, RF, GBM model during non-peak hours based on MAPE criterion.

Model	Segment	1-step (%)	2-step (%)	3-step (%)	4-step (%)	5-step (%)	6-step (%)
GBM	A	**2.34**	**3.46**	**4.07**	**4.39**	**4.72**	**5.30**
	B	**2.81**	**4.81**	**6.04**	7.78	**9.62**	**11.18**
	C	**3.58**	**6.27**	**8.47**	11.12	13.45	14.81
RF	A	2.82	3.48	4.49	4.83	5.26	5.40
	B	3.26	5.09	6.13	**7.44**	9.90	11.47
	C	4.60	6.98	8.86	**10.63**	**11.02**	**12.55**
ARIMA	A	2.49	3.96	4.86	5.38	5.66	5.94
	B	3.29	6.12	8.30	10.46	13.02	15.65
	C	4.17	7.41	10.40	13.01	16.05	18.96

The bold and underline values are the smallest MAPE values within each segment.

In this experiment, the ARIMA, RF and GBM models' performance are evaluated during both peak and nonpeak periods. Please note that we also considered seasonal ARIMA (SARIMA) model in our study, but the SARIMA model is no better than ARIMA. This is partly because the over fitting of the SARIMA model. Therefore, we did not include the SARIMA model in our comparison. Two months travel time data are used for model training and testing. Tables 9 and 10 provide MAPE values of the three models from one-step up to six-step ahead prediction during peak and non-peak hours. As shown in Table 9, the GBM model has the smallest MAPE value 14 out of 18 cases. While, in Table 10, the GBM model outperforms the other two models 14 out of 18 cases. The GBM model shows its advantages in both congested (Table 9) and uncongested (Table 10) cases. All three models perform well in uncongested conditions and show a decrease in predicting accuracy when congestion occurs. This is because traffic can be easily affected when congestion occurs and the variations of travel time are often higher than that during uncongested conditions. The traffic's transitional behavior and the existence of variant traffic regimes can be difficult to be identified (Chen et al., 2012), therefore, increase the prediction error of the model. The GBM model has the advantage in modeling very complex relations of the data and outperforms the other two models especially during congested situations.

4. Discussion and conclusion

The GBM model has its unique feature that distinguishes it from other popular ensemble methods, such as bagged trees and random forests. Both bagged trees and random forests are able to reduce variance more than single trees through averaging. Random forests enhance diversity through randomly selecting a subset of variables at each splitting node. The training sample is produced from bootstrap sampling with its distribution similar to the original training set. The bias of the model cannot be reduced through averaging. On the other hand, the GBM model grows trees sequentially by adjusting the weight of the training data distribution to minimize certain loss function. It reduces both model bias through forward stage-wise modeling and reduces variance through averaging (Elith et al., 2008). The proposed GBM-based travel time prediction method has considerable advantages over classical statistical approaches and other ensemble methods. Especially, it has superior performance in terms of prediction accuracy.

There are very limited studies discussed the RF model application in traffic prediction (Hamner, 2010; Leshem and Ritov, 2007). To the best knowledge of the authors, we did not find any studies on the application of the GBM model with freeway travel time data. There is no comparison and discussion on the performance of the RF and the GBM models in traffic prediction. The GBM model can handle sharp discontinuities, an important feature when modeling abnormal traffic conditions where traffic states change from uncongested to congested, and vice versa. Based on data used in this study, the GBM model is able to capture the sudden changes of traffic (for example, in incidents or raining conditions). In addition, the GBM model can automatically select relevant variables, fit accurate models, and identify and model parameter interactions. More importantly, different from other machine learning algorithms as a 'black-box', the relative importance or contribution of input variables are also discussed through the GBM model. This is critical for us to get powerful insight into the structure of the data. In addition, comparisons of the ARIMA, the RF and the GBM models indicate that both the RF and the GBM models outperform the conventional statistical model, ARIMA model; the advantage of the GBM becomes more evident for multi-step-ahead prediction.

One issue regarding the application of the GBM model in travel time prediction is related with parameter optimization. As addressed in the model optimization section, the performance of the GBM model is largely influenced by its parameters, including number of trees, learning rate and tree complexity (variable interactions). Therefore, there is a need to test the optimal combination of variables when developing the GBM model. Computational time is another issue when tree complexity or the number of trees increases. The tradeoff between computation cost and model accuracy should also be considered when building the model.

In short, the GBM model has its considerable advantages in freeway travel time prediction. In particular, the recent advanced technology development enables collecting different kinds of traffic data from road sensors, smart phones, GPS devices and so on. As more information can be accessed to study traffic phenomena, it is critical to find a model that is able to make good use of the big data and modeling the complex relations in the data. The capability of the GBM model in handling different types of input variables, in modeling complex nonlinear relationship makes it a promising algorithm for travel time prediction. This study only considers the temporal correlations of data. Further research can also incorporate spatial correlations by considering travel time information from locations other than the location of interest to gain more information and therefore to capture the traffic dynamics more accurately.

References

Ahmed, M.M., Abdel-Aty, M., 2013. Application of stochastic gradient boosting technique to enhance reliability of real-time risk assessment. Transp. Res. Rec.: J. Transp. Res. Board 2386, 26–34.

Ali Haghani, M.H., Kaveh Farokhi Sadabadi, 2010. Estimation of Travel Times for Multiple TMC Segments. Available: http://www.i95coalition.org/i95/Portals/0/Public_Files/uploaded/Vehicle-Probe/I-95%20CC-Estimation%20of%20Travel%20Times%20for%20Multiple%20TMC%20Segments%20-%20FINAL2.pdf.

Amit, Y., Geman, D., 1997. Shape quantization and recognition with randomized trees. Neural Comput. 9, 1545–1588.

Antoniou, C., Koutsopoulos, H.N., Yannis, G., 2013. Dynamic data-driven local traffic state estimation and prediction. Transp. Res. Part C: Emerg. Technol. 34, 89–107.

Breiman, L., 1984. Classification and Regression Trees. Wadsworth International Group, Belmont, Calif.

Breiman, L., 1996. Bagging predictors. Mach. Learn. 24, 123–140.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Chen, H.K., Wu, C.J., 2012. Travel time prediction using empirical mode decomposition and gray theory example of National Central University Bus in Taiwan. Transp. Res. Rec., 11–19

Chen, X., Li, L., Li, Z., 2012. Phase diagram analysis based on a temporal–spatial queueing model. Intell. Transp. Syst. IEEE Trans. 13, 1705–1716.

Choe, T., Skabardonis, A., Varaiya, P., 2002. Freeway performance measurement system: operational analysis tool. Transp. Res. Rec.: J. Transp. Res. Board 1811, 67–75.

Chung, Y.-S., 2013. Factor complexity of crash occurrence: an empirical demonstration using boosted regression trees. Accid. Anal. Prev. 61, 107–118.

Elith, J., Leathwick, J.R., Hastie, T., 2008. A working guide to boosted regression trees. J. Anim. Ecol. 77, 802–813.

Farokhi Sadabadi, K., Hamedi, M., Haghani, A., 2010. Evaluating moving average techniques in short-term travel time prediction using an AVI data set. Transportation Research Board 89th Annual Meeting.

Fei, X., Lu, C.C., Liu, K., 2011. A bayesian dynamic linear model approach for real-time short-term freeway travel time prediction. Transp. Res. Part C – Emerg. Technol. 19, 1306–1318.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Ann. Stat., 1189–1232

Friedman, J.H., 2002. Stochastic gradient boosting. Comput. Stat. Data Anal. 38, 367–378.

Hamner, B., 2010. Predicting travel times with context-dependent random forests by modeling local and aggregate traffic flow. In: Data Mining Workshops (ICDMW), 2010 IEEE International Conference on IEEE, 1357–1359.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., Tibshirani, R., 2009. The Elements of Statistical Learning. Springer.

Ho, T. K. Random decision forests. Document Analysis and Recognition, 1995, Proceedings of the Third International Conference on, 1995. IEEE, 278–282.

Ho, T.K., 1998. The random subspace method for constructing decision forests. Pattern Anal. Mach. Intelligence, IEEE Trans. 20, 832–844.

Hong, W.-C., 2011. Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. Neurocomputing 74, 2096–2107.

Hyndman, R.J., Khandakar, Y., 2007. Automatic Time Series for Forecasting: the Forecast Package for R.

Kamarianakis, Y., Prastacos, P., 2005. Space–time modeling of traffic flow. Comput. Geosci. 31, 119–133.

Karlaftis, M.G., Vlahogianni, E.I., 2009. Memory properties and fractional integration in transportation time-series. Transp. Res. Part C: Emerg. Technol. 17, 444–453.

Kearns, M., 1988. Thoughts on Hypothesis Boosting. Unpublished manuscript, December.

Koren, Y., 2009. The Bellkor Solution to the Netflix Grand Prize. Netflix Prize Documentation.

Laboratory, T.C.F.A.T.T. Available: http://www.cattlab.umd.edu/?portfolio=ritis.

Leshem, G., Ritov, Y.A., 2007. Traffic flow prediction using Adaboost algorithm with random forests as a weak learner. Int. J. Intelligent Technol. 2.

Li, L., Chen, X., Li, Z., Zhang, L., 2013a. Freeway travel-time estimation based on temporal–spatial queueing model. Intelligent Transp. Syst. IEEE Trans. 14, 1536–1541.

Li, L., Li, Y., Li, Z., 2013b. Efficient missing data imputing for traffic flow by considering temporal and spatial dependence. Transp. Res. Part C: Emerg. Technol. 34, 108–120.

Li, L., Chen, X., Zhang, L., 2014. Multimodel ensemble for freeway traffic state estimations. IEEE Trans. Intell. Transp. Syst. 15.

Min, W., Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transp. Res. Part C: Emerg. Technol. 19, 606–616.

Natekin, A., Knoll, A., 2013. Gradient boosting machines, a tutorial. Front. Neurorobotics 7.

Schapire, R.E., 1990. The strength of weak learnability. Mach. Learn. 5, 197–227.

Smith, B., Demetsky, M., 1997. Traffic flow forecasting: comparison of modeling approaches. J. Transp. Eng. 123, 261–266.

Tsay, R.S., 2010. Analysis of Financial Time Series. Wiley, Cambridge, Mass.

van Hinsbergen, C., van Lint, J., van Zuylen, H., 2009. Bayesian committee of neural networks to predict travel times with confidence intervals. Transp. Res. Part C: Emerg. Technol. 17, 498–509.

van Lint, J.W.C., van Hinsbergen, C.P.I.J., 2012. Short-term traffic and travel time prediction models. Artif. Intell. Appl. Critical Transp. Issues 22.

van Lint, J., Hoogendoorn, S., van Zuylen, H.J., 2005. Accurate freeway travel time prediction with state-space neural networks under missing data. Transp. Res. Part C: Emerg. Technol. 13, 347–369.

Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2014. Short-term traffic forecasting: where we are and where we're going. Transp. Res. Part C: Emerg. Technol.

Wang, Y., 2011. Prediction of weather impacted airport capacity using ensemble learning. In: Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th, 2011. IEEE, 2D6-1-2D6-11.

Wang, J., Shi, Q., 2012. Short-term traffic speed forecasting hybrid model based on Chaos-Wavelet Analysis-Support Vector Machine theory. Transp. Res. Part C: Emerg. Technol.

Wei, Y., Chen, M.-C., 2012. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. Transp. Res. Part C: Emerg. Technol. 21, 148–162.

Williams, B., Durvasula, P., Brown, D., 1998. Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models. Transp. Res. Rec.: J. Transp. Res. Board 1644, 132–141.

Yang, M., Liu, Y., You, Z., 2010. The reliability of travel time forecasting. Intell. Transp. Syst. IEEE Trans. 11, 162–171.

Yeon, J., Elefteriadou, L., Lawphongpanich, S., 2008. Travel time estimation on a freeway using Discrete Time Markov Chains. Transp. Res. Part B: Methodol. 42, 325–338.

Zhang, X., Rice, J.A., 2003. Short-term travel time prediction. Transp. Res. Part C: Emerg. Technol. 11, 187–210.

Zhang, Y., Sun, R., Haghani, A., Zeng, X., 2013a. Univariate volatility-based models for improving quality of travel time reliability forecasting. Transp. Res. Rec.: J. Transp. Res. Board 2365, 73–81.

Zhang, Y., Zhang, Y., Haghani, A., 2013b. A hybrid short-term traffic flow forecasting method based on spectral analysis and statistical volatility model. Transp. Res. Part C: Emerg. Technol.

Zhou, Z.-H., 2012. Ensemble Methods: Foundations and Algorithms. Taylor & Francis, Boca Raton, FL.