Center for Advanced Multimodal Mobility Solutions and Education

Project ID: 2019 Project 01

PREDICTING TRAVEL TIME ON FREEWAY CORRIDORS: MACHINE LEARNING APPROACH

Final Report

by

Wei Fan (ORCID ID: https://orcid.org/0000-0001-9815-710X)
Zhen Chen (ORCID ID: https://orcid.org/0000-0001-6785-2652)

Wei Fan, Ph.D., P.E.
Director, USDOT CAMMSE University Transportation Center
Professor, Department of Civil and Environmental Engineering
The University of North Carolina at Charlotte
EPIC Building, Room 3261, 9201 University City Blvd, Charlotte, NC 28223
Phone: 1-704-687-1222; Email: wfan7@uncc.edu

for

Center for Advanced Multimodal Mobility Solutions and Education
(CAMMSE @ UNC Charlotte)
The University of North Carolina at Charlotte
9201 University City Blvd
Charlotte, NC 28223

September 2020

ACKNOWLEDGEMENTS

This project was funded by the Center for Advanced Multimodal Mobility Solutions and Education (CAMMSE @ UNC Charlotte), one of the Tier I University Transportation Centers that were selected in this nationwide competition, by the Office of the Assistant Secretary for Research and Technology (OST-R), U.S. Department of Transportation (US DOT), under the FAST Act. The authors are also very grateful for all of the time and effort spent by DOT and industry professionals to provide project information that was critical for the successful completion of this study.

DISCLAIMER

Table of Contents

EXECUTIVE SUMMARY ..... **ix**

Chapter 1. Introduction **1**

1.1 Problem Statement 1

1.2 Motivation of Study 1

1.3 Objectives of Study 1

1.4 Report Overview 1

Chapter 2. Literature Review **5**

2.1 Introduction 5

2.2 Travel Time Prediction Using Machine Learning Approaches 5

2.2.1 Support Vector Machine (SVM) Approach 5

2.2.2 Neural Network Approach 5

2.2.3 Nearest Neighbors Approach 7

2.2.4 Ensemble Learning Approach 8

2.3 Summary 10

Chapter 3. Data Collection and Processing **11**

3.1 Introduction 11

3.2 Travel Time Data Collection 11

3.3 Weather Data Collection 12

3.4 Data Processing 13

3.5 Summary 14

Chapter 4. Travel Time Prediction Methodology **15**

4.1 Introduction 15

4.2 Basic Information on the Ensemble Learning Methodology 15

4.2.1 Bagging algorithm 16

4.2.2 Boosting Algorithm 17

4.3 XGBoost Algorithm 18

4.4 Summary 21

Chapter 5. Travel Time Prediction Model Validation **23**

5.1 Introduction 23

5.2 Feature Selection and Pre-processing 23

5.3 Parameter Tuning Process 26

5.4 Summary 34

Chapter 6. Prediction Results Analysis **35**

6.1 Introduction 35
6.2 Modelling Results Analysis 35
6.3 Model Comparison 37
6.4 Summary 38
Chapter 7. Summary and Conclusions 41
7.1 Summary 41
7.2 Summary and Conclusions of Travel Time Prediction Results 41
7.3 Future Work Directions 42
References 43

List of Figures

Figure 1.1: Research Structure 3
Figure 3.1: Selected I-77 Southbound Section 11
Figure 4.1: Pseudo-code for Decision Tree 16
Figure 4.2: Pseudo-code for Bagging 17
Figure 5.1: XGBoost Travel Time Prediction Model Outputs with the Max_depth =5 28
Figure 5.2: XGBoost Travel Time Prediction Model Outputs with the Max_depth=6 29
Figure 5.3: XGBoost Travel Time Prediction Model Outputs with the Max_depth=7 29
Figure 5.4: XGBoost Travel Time Prediction Model Outputs with the Max_depth=8 30
Figure 5.5: XGBoost Travel Time Prediction Model Outputs with the Max_depth=9 30
Figure 5.6: XGBoost Travel Time Prediction Model Outputs with the Max_depth=10 31

List of Tables

Table 2.1: Summary of Travel Time Prediction Using Machine Learning Approaches 9
Table 3.1: Sample Raw Travel Time Data 12
Table 3.2: Sample Raw Weather Data 13
Table 3.3: Classification of the Weather Conditions 13
Table 5.1: Definitions and Attributes on Selected Features 24
Table 5.2: Example of the Raw Inputs of the Model 25
Table 5.3: MAEs of Different Learning Rates, Number of Trees and Max_depth 32
Table 5.4: Optimized Prediction Results and Computation Times 33
Table 6.1: Relative Importance of Each Variable and Their Ranks in the Model 36
Table 6.2: Performance Comparison between XGBoost Model and Gradient Boosting Model 38

EXECUTIVE SUMMARY

Nowadays anonymous vehicle probe data have been greatly improved in both data coverage and data fidelity. Thus, vehicle probe data have become a reliable source for freeway travel time analysis. Travel time prediction plays a significant role in traffic data analysis and applications as it can assist in route planning and reducing traffic congestion. With the development of artificial intelligence technologies, various novel prediction methods have been developed accordingly in recent years. Machine learning is an example of a data driven method which aims to increase efficiency and accuracy of predictions. Recently, different machine learning-based approaches, such as neural network, ensemble learning, and support vector machines, have been employed by the researchers and the results indicate that such approaches for prediction are adaptable and can give better performances than traditional models.

In this study, an advanced machine learning-based approach (i.e. XGBoost model) is employed to predict the freeway travel time. Detailed information about the input variables and data pre-processing is presented. Parameters of the XGBoost model are introduced and the parameter tuning process is also discussed. The relative importance of each variable in the model is presented and interpreted. Optimized modeling results of the proposed XGBoost travel time prediction model are evaluated and compared with those of the gradient boosting model. The results also demonstrate that the developed XGBoost travel time prediction model significantly improves the computation accuracy and efficiency. Summary and conclusions of the whole study are made and further research directions are given at the end of study.

Chapter 1. Introduction

1.1 Problem Statement

Nowadays anonymous vehicle probe data have been greatly improved in both data coverage and data fidelity, and thus have become a reliable source for freeway travel time analysis. Travel time prediction plays a significant role in traffic data analysis and applications as it can greatly help in route planning and reducing traffic congestion. Traditionally, methods such as linear regression and time series models have been widely applied to predict travel times using historical data. However, with the consideration of effectiveness, accuracy, and feasibility, these models may become outdated and replaceable. With the development of artificial intelligence technologies, various novel prediction methods have been developed accordingly in recent years. Machine learning is an example of a data driven method which aims to increase efficiency and accuracy of the prediction. Recently, different machine learning-based approaches, such as neural network, ensemble learning, and support vector machines (SVM), have been employed by the researchers and the results indicate that such approaches for prediction are adaptable and can give better performances than traditional models. Therefore, the machine learning-based approach is selected for the travel time prediction in this study.

1.2 Motivation of Study

The purpose of this project is to develop a systematic approach to predicting freeway travel time. An advanced machine learning-based approach (i.e. XGBoost model) is employed to predict the freeway travel time. The prediction methodology can assist the decision makers in planning, designing, operating, and managing a more efficient highway system.

1.3 Objectives of Study

Specific objectives are to: 1) Develop the travel time prediction model using an advanced, efficient and accurate machine learning-based approach, 2) Select a real-world freeway corridor to examine the developed prediction model, and 3) Evaluate the prediction results of the developed model.

1.4 Report Overview

The report will be structured as shown in Figure 1.1. In this chapter, the significance and motivation of the study on travel time prediction have been discussed, followed by the description of study objectives.

Chapter 2 presents a comprehensive review of the current state-of-the-art and state-of-the-practice travel time prediction methodologies. In detail, several machine learning-based methods used by the reviewed studies including the neural network approach, ensemble learning approach, K-nearest neighbor (K-NN) approach, and support vector machine approach, will be presented.

Chapter 3 describes the basic information needed to predict travel time, including the travel time data and historical weather data utilized in this study. Detailed information about the

raw travel time data source is described first, followed by the discussions about weather data collection.

Chapter 4 presents the travel time prediction methodology which is utilized in this study. The idea of ensemble learning is introduced first. Detailed information on the decision tree algorithm, bagging algorithm, and boosting algorithm is presented. The basic information about the Random Forest and gradient boosting models is described including advantages and disadvantages. An introduction of the XGBoost model is also presented in this chapter. Advantages of the XGBoost model are listed. The detailed process of the XGBoost model is described including the objective function, regularization terms, and model score.

Chapter 5 discusses the validation steps of the proposed XGBoost-based travel time prediction model based on the data described in Chapter 3. Selected features include, but are not limited to, the following: time of day (TOD), day of week (DOW), month of year, year, weather conditions, segment characteristics, etc. Detailed information about the input variables and data pre-processing is presented. The parameters of the XGBoost model are introduced and the parameter tuning process is also discussed. The experiment results could give a clear picture of how the analyzed parameters impact the prediction performance.

Chapter 6 presents the interpretation and evaluation of the numerical results of the developed XGBoost model. The relative importance of each variable in the model is presented and interpreted. In order to examine the accuracy and effectiveness of the proposed model, this chapter also evaluates the optimized modeling results of the proposed XGBoost travel time prediction model and compares them with those of the gradient boosting model. The results also demonstrate that the developed XGBoost travel time prediction model significantly improves the computational accuracy and efficiency.

Chapter 7 concludes the study with a summary of the discussions about the developed travel time prediction model and the modeling results. Suggestions for future research are also provided.

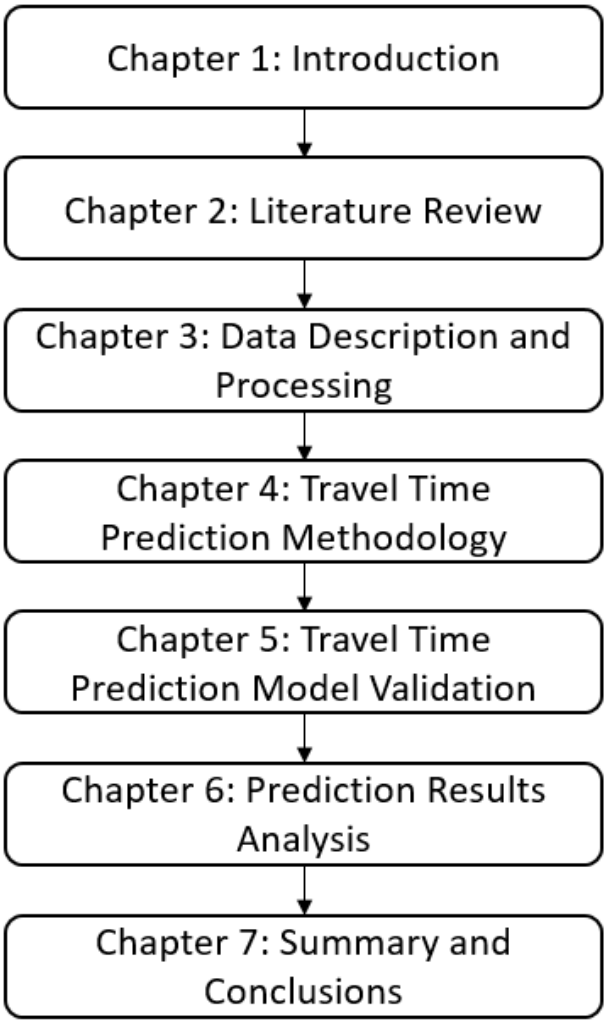

Figure 1.1: Research Structure

Chapter 2. Literature Review

2.1 Introduction

This chapter provides a comprehensive review of machine learning-based travel time prediction studies. This should give a clear picture of existing efforts toward the travel time prediction.

The following sections are organized as follows. Section 2.2 gives a comprehensive review of existing machine learning-based travel time prediction studies. Section 2.3 concludes this chapter with a summary.

2.2 Travel Time Prediction Using Machine Learning Approaches

Traditionally, methods such as linear regression and time series models have been widely applied to predict travel times using historical data. However, with the consideration of effectiveness, accuracy and feasibility, these models may become outdated and replaceable. With the development of artificial intelligence technologies, various novel prediction methods have been developed accordingly in recent years. With the help of intelligent transportation systems (ITS) and the traffic data, different machine learning approaches have been deployed in the travel time prediction area. The methodology can include, but are not limited to: Support vector machine regression, Neural network approaches (e.g., State-and-space neural network, long short-term memory neural network), nearest neighbor (e.g., k-nearest neighbor), and ensemble learning (e.g., Random Forest and gradient boosting), etc. The review of different approaches will be helpful to find the most appropriate, advanced and accurate model in this study. Research studies that used machine learning/deep learning methods to predict travel time are reviewed and summarized in this section. Table 2.1 provides a summary of the studies reviewed in this section in chronological order.

2.2.1 Support Vector Machine (SVM) Approach

2.3.1.1 Wu et al.'s research work

Wu et al. (2004) applied SVM for travel-time prediction and compared its results to other baseline travel time prediction methods using real world highway traffic data. Since support vector machines have greater generalization ability and can guarantee global minima for given training data, it was believed that SVM would perform well for time series analysis. The results showed that the SVM predictor can significantly reduce both relative mean errors and root-mean-squared errors of predicted travel times. This study demonstrated the feasibility of applying SVM in travel time prediction and proved that SVM is applicable for traffic data analysis.

2.2.2 Neural Network Approach

2.3.2.1 Park and Rilett's research work

Park and Rilett (1999) proposed a BP neural network model to predict freeway link travel time. The freeway link travel time data collected on the freeways of Houston, Texas, by the automatic vehicle identification (AVI) system were used as the validation database.

The proposed model can provide acceptable prediction results with the mean absolute percentage error (MAPE) being ranged from 7.4% to 18%.

2.3.2.2 Van Lint et al.'s research work

Van lint et al. (2002) presented an approach to predicting freeway travel time based on the state-space neural network. The data from freeway operations simulation (FOSIM) 4.1 were used to train and test the travel time prediction model. The authors also eliminated the insignificant parameters in the model and made it more effective without the loss of predictive performance.

2.3.2.3 Wisitpongphan et al.'s research work

Wisitpongphan et al. (2012) proposed a back propagation (BP) neural network model to predict freeway link travel time. The one-month vehicle trajectory data of 297 probe vehicles via GPS database in Thailand were used as the validation database. The prediction results of the proposed model can accurately approximate the travel time with the mean squared error (MSE) being less than 3%.

2.3.2.4 Zheng and Van Zuylen's research work

Zheng and Van Zuylen (2013) conducted a study using the probe vehicle data to estimate complete link travel times. Based on the information collected by probe vehicles, a three-layer neural network model was developed by the authors to estimate complete link travel time for individual probe vehicle traversing the link. The estimation result of this model was then compared with that of an analytical estimation model. The performance of these two models were evaluated using the data derived from VISSIM simulation model. The final results suggested that the Artificial Neural Network model performs better.

2.3.2.5 Duan et al.'s research work

Duan et al. (2016) employed a long short-term memory (LSTM) neural network model to predict freeway travel time. The authors constructed 66 series LSTM neural networks by using travel time data collected along 66 links of the highways in England. The authors discussed the predictions of multi-step ahead travel time and found 1-step ahead travel time prediction can provide best results.

2.3.2.6 Liu et al.'s research work

Liu et al. (2017) proposed a LSTM deep neural network model using 16 settings of hyper-parameters to predict the travel time on the interstate highways in California, U.S. The results of proposed model were compared with the results of other regression models and Autoregressive integrated moving average (ARIMA) model and showed that the performance of the LSTM neural network model was the best.

2.3.2.7 Wang et al.'s research work

Wang et al. (2018) presented a novel machine learning method to predict the vehicle travel time based on floating-car data. The authors adapted different machine learning models to solve the regression problem. Furthermore, the authors evaluated the solution offline with millions of historical vehicle travel data and the results showed that their

proposed deep learning algorithm significantly outperforms the other state-of-the-art algorithms.

2.3.2.8 Wang et al.'s research work

Wang et al. (2018) proposed a LSTM neural network-based travel time prediction model using the historical vehicle trajectory data. Both road segment-based travel time estimation and path-based travel time estimation were discussed in this study. The results showed that the proposed model can effectively capture the spatial and temporal dependencies and accurately predict travel time.

2.3.2.9 Wei et al.'s research work

Wei et al. (2018) combined the convolutional neural network and LSTM neural network together to predict the short-term travel time. The vehicle trajectory data on the urban roads were used in this study. The author pointed out that the prediction of the proposed model was more effective than that of other existing approaches.

2.2.3 Nearest Neighbors Approach

2.3.3.1 Yu et al.'s research work

Yu et al. (2017) combined the Random Forest model and K-NN model in their study to predict bus travel time. The proposed combined-model was compared with linear regression, K-NN, SVM and Random Forest. The results showed the proposed model achieved highest accuracy level and can be applied to real-time prediction.

2.3.3.2 Myung et al.'s research work

Myung et al. (2011) proposed a model to predict travel times on the basis of the k nearest neighbor (KNN) method using data provided by the vehicle detector system and the automatic toll collection system. By combining these two sets of data, the model minimized the limitations of each dataset and enhanced the prediction's accuracy. The authors compared the prediction results of the proposed model with the predictions of other models by using actual data. The comparison results showed that the proposed model predicts travel times much more accurately.

2.3.3.3 Moonam et al.'s research work

Moonam et al. (2019) conducted a study to predict the expected travel time based on the experienced travel time using the data mining techniques such as k-nearest neighbor (k-NN), least squares regression boosting and Kalman filter (KF) methods. The authors compared the performances of each methods from both link and corridor perspectives and concluded that the KF method offers superior prediction accuracy in a link-based model.

2.2.4 Ensemble Learning Approach

2.3.4.1 Hamner et al.'s research work

Hamner et al. (2011) applied a context-dependent Random Forest method to predict travel-time based on GPS data of the cars on the road in a simulation framework. The root mean squared error (RMSE) of the model was less than 7.5%.

2.3.4.2 Zhang and Haghani's research work

Zhang and Haghani (2015) employed a gradient boosting regression tree method to analyze and predict freeway travel time to improve the prediction accuracy. The authors used travel time data along freeway sections in Maryland and discussed the effects of different parameters on the proposed model and the correlations of input and output variables. The prediction results showed the proposed model can provide considerable advantages in freeway travel time prediction.

2.3.4.3 Li and Bai's research work

Li and Bai (2016) employed a gradient boosting regression tree method to analyze and predict travel time of freight vehicles. The authors used travel time data and vehicle trajectory data in Ningbo, China. The prediction results showed the proposed model can be feasible in the real-world.

2.3.4.4 Fan et al.'s research work

Fan et al. (2017) conducted a study using the Random Forest method to predict highway travel time based on data collected from highway electronic toll collection in Taiwan. The results can help highway drivers to select optimal departure times to avoid traffic congestion and thus minimize travel time.

2.3.4.5 Gupta et al.'s research work

Gupta et al. (2018) employed Random Forest and gradient boosting models to predict taxi travel time in Porto, Portugal. The vehicle trajectory data were used as the database and it was found that the gradient boosting model provided better prediction results than the Random Forest model.

Table 2.1: Summary of Travel Time Prediction Using Machine Learning Approaches

Year	Author	Location	Roadway Category	Data Source	Data Type	Prediction method
1999	Park and Rilett	Houston, US	Highway	AVI system	Travel time	BP Neural Network
2002	Van Lint et al.	N/A	Freeway	FOSIM (freeway operations simulation)	Travel time, travel speed	State-Space Neural Network
2005	Wu et al.	Taiwan	Highway	Loop detector	Travel speed	SVM
2010	Hamner et al.	N/A	N/A	Global Positioning System (GPS)	Travel speed	Random Forest
2011	Myung et al.	Korea	N/A	Automatic traffic count system	Travel time	K-NN
2012	Wisitpongphan	Bangkok, Thailand	Highway	GPS	Travel time, GPS	BP Neural Network
2013	Zheng and Van Zuylen	Delft, Netherlands	Urban road	GPS data	Vehicle position, travel speed	State-Space Neural Network
2015	Zhang and Haghani	Maryland, US	Interstate highway	INRIX Company	Travel time	Gradient boosting
2016	Duan et al.	England	Highway	Cameras, GPS and loop detectors	Travel time	LSTM Neural Network
2016	Li and Bai	Ningbo, China	N/A	N/A	Truck trajectory, travel time, travel speed	Gradient boosting
2017	Liu et al.	California, US	Interstate highway	Freeway performance measurement system (PeMS)	Travel time	LSTM Neural Network
2017	Fan et al.	Taiwan	Highway	Electric toll	Travel time, vehicle information	Random Forest
2017	Yu et al.	Shenyang, China	Bus route	Automatic Vehicle Location system	Bus travel time	Random Forest and K-NN
2018	Wang et al.	Beijing, China	Urban road	Floating car data	Taxi ravel time, vehicle trajectory data	LSTM Neural Network
2018	Wei et al.	China	Urban road	Vehicle passage records	Travel time	LSTM Neural Network
2018	Wang et al.	Beijing and Chengdu, China	Urban road	GPS	Vehicle trajectory data	LSTM Neural Network
2018	Gupta et al.	Porto, Portugal	Urban road	GPS	Taxi travel speed	Random forest and gradient boosting
2019	Moonam et al.	Madison, Wisconsin, US	Freeway	Bluetooth detector	Travel speed	K-NN, KF

2.3 Summary

A comprehensive review and synthesis of the current and historical researches related to travel time prediction have been discussed and presented in the preceding sections. This is intended to provide a solid reference and assistance in developing travel time prediction models.

Chapter 3. Data Collection and Processing

3.1 Introduction

This chapter provides the basic information needed to conduct travel time prediction, including the travel time data and historical weather data utilized in this study. The following sections are organized as follows. Section 3.2 presents detailed information about the raw travel time data source, followed by the discussions about weather data collection in section 3.3. Section 3.4 describes details of data processing. Finally, section 3.5 concludes this chapter with a summary.

3.2 Travel Time Data Collection

This study focuses on the travel time data gathered from the Regional Integrated Transportation Information System (RITIS) website and uses the collected data to conduct the TTR analysis and travel time prediction. A series of major freeway segments are selected for the case study: Interstate 77 (I-77) Southbound (Figure 3.1) is one of the most heavily traveled Interstate highways in Charlotte, North Carolina and runs from north to south. All the selected segments have uninterrupted coverage of RITIS data 24 hours per day and 365 days a year.

The selected section of I-77 Southbound starts from the intersection with US-21 (Exit 16) and ends at the interchange with Nations Ford Road (Exit 4) at the south part of the city. 26 roadway segments are selected in this study, and the total length of the selected section is 15 miles.

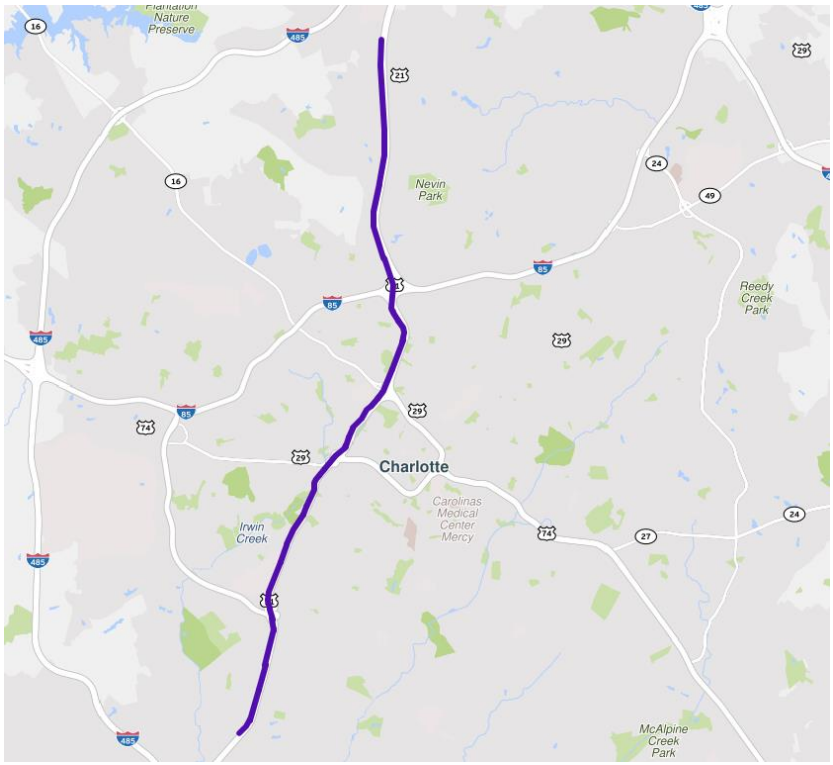

Figure 3.1: Selected I-77 Southbound Section

The raw probe data can be downloaded with the desired section and format on the RITIS website probe data analytic suite. The roadway section can be selected based on the road location, traffic message channel (TMC), directions, zip codes, etc. The date range can be selected from January 1st, 2008 to today. Seven days of week and times of day from 12:00 AM to 11:59 PM can also be selected. The averaging period can be selected as five minutes, ten minutes, fifteen minutes and one hour. A sample of raw travel time data utilized in this study is shown in Table 3.1 below:

Table 3.1: Sample Raw Travel Time Data

Timestamp	TMC code	Travel time (s)
1/1/2015 0:00	125-04785	5.16
1/1/2015 0:00	125-04782	35.68
1/1/2015 0:00	125-04783	33.22
1/1/2015 0:00	125N04789	54.5
1/1/2015 0:00	125N04787	53.76
1/1/2015 0:00	125N04788	29.6
1/1/2015 0:00	125N04781	12.42
1/1/2015 0:00	125N04782	21.73
1/1/2015 0:00	125N04780	14.59
1/1/2015 0:00	125N04785	11.85
1/1/2015 0:00	125N04786	47.56
1/1/2015 0:00	125N04783	12.82
1/1/2015 0:00	125N04784	53.58

Table 3.1 contains the following information:

TMC Code: The RITIS Probe Data Analytics Suite uses the TMC standard to uniquely identify each road segment. This column indicates the segment ID.

Timestamp: This column indicates the timestamp of the record.

Travel time: This column indicates the time it will take to drive along the roadway segment.

3.3 Weather Data Collection

The historical weather data near the Charlotte Douglas International airport can be found at the www.wunderground.com website. The raw weather data from this website were recorded per hour. Due to the discrepancy in the time interval, one-to-one mapping or correlation study cannot be done using the original data. Hence, the methodology to combine the traffic data with the weather data will be discussed in the next section. The sample of weather data achieved is shown in Table 3.2 below.

Table 3.2: Sample Raw Weather Data

Date	Time (EDT)	Conditions
Saturday, March 14, 2015	6:55 AM	Rain
Saturday, March 14, 2015	7:55 AM	Rain
Saturday, March 14, 2015	8:55 AM	Light Rain
Saturday, March 14, 2015	9:55 AM	Light Rain
Saturday, March 14, 2015	10:55 AM	Light Rain
Saturday, March 14, 2015	11:55 AM	Light Rain
Saturday, March 14, 2015	12:55 PM	Light Rain
Saturday, March 14, 2015	1:55 PM	Light Rain
Saturday, March 14, 2015	2:55 PM	Light Rain
Saturday, March 14, 2015	3:55 PM	Light Rain
Saturday, March 14, 2015	4:55 PM	Rain

3.4 Data Processing

Due to the weather characteristics in the Charlotte area and the distribution of each weather category, detailed weather conditions are categorized into three groups including normal, rain, and snow/fog/ice. Table 3.3 presents the detailed classification of the weather conditions. Conditions such as “overcast” or “mostly cloudy” are assumed to be no different from “clear” conditions due to no obvious impact on traffic conditions. These conditions are categorized into ‘normal’. All the conditions such as ‘rain’ or ‘thunderstorm’ are categorized as ‘rain’. In order to ensure the acceptable sample size, “snow”, “fog”, “ice pellet”, and other similar conditions are combined together due to their rate of occurrence. The RITIS datasets are aggregated into 15- minutes intervals, while the weather dataset is aggregated into one-hour intervals. Therefore, the weather conditions are distributed evenly with RITIS dataset based on the timestamp.

Table 3.3: Classification of the Weather Conditions

New Weather Category	Original Weather Condition
Snow/fog/ice	Haze
	Fog
	Smoke
	Patches of Fog

New Weather Category	Original Weather Condition
	Mist
	Shallow Fog
	Light Freezing R
	Light Ice Pellet
	Light Freezing D
	Light Freezing F
	Ice Pellets
	Light Snow
	Snow
	Heavy Snow
Normal	Clear
	Partly Cloudy
	Mostly Cloudy
	Scattered Clouds
	Overcast
	Unknown
Rain	Light Rain
	Rain
	Heavy Rain
	Light Drizzle
	Heavy Thunderstorm
	Light Thunderstorm
	Thunderstorm
	Drizzle
	Squalls

3.5 Summary

This chapter presents the detailed information on the data source, data structure, and processing methodology to combine the travel time with raw weather data. This is intended to provide a solid reference and assistance in predicting travel time for future tasks.

Chapter 4. Travel Time Prediction Methodology

4.1 Introduction

This chapter presents the introduction to the travel time prediction methodology. The following sections are organized as follows. Section 4.2 shows the basic information about the ensemble learning methodology, which includes the ideas of bagging algorithm and boosting algorithm. Section 4.3 discusses the principles of the XGBoost algorithm. Finally, section 4.4 concludes this chapter with a summary.

4.2 Basic Information on the Ensemble Learning Methodology

The ensemble learning-based algorithms consist of multiple base models (e.g., decision tree model), and each base model provides an alternative solution to the problem. The prediction results of these base models are combined by some rules (such as weighted or unweighted voting and averaging). The final output will be achieved through the combined model.

The base model of the ensemble learning algorithm is extremely important to the final results. Since the model is expected to have enough degrees of freedom to solve the underlying complexity of the data and avoid high variance and be more robust at the same time, the two most fundamental characteristics of the base model should be a low bias and a low variance. In other words, the base model should be a 'weak learner' and needs to be converted to a 'strong learner'. In machine learning area, a 'weak learner' means a model that performs slightly better than random guessing.

Decision tree is a basic data-driven supervised learning method and has been widely used in the data mining area (Quinlan, 1986; Han and Kamber, 2011). A single decision tree is constructed by splitting the features' space into regions. The target variable can be predicted by using the values of a set of features.

In detail, the pseudo-code for decision tree is shown below in Figure 4.1, which can make it easier to understand the idea of decision tree algorithm.

DecisionTree (Dataset D, Attributes A)

Create a node N;

- If all samples are of the same class C then label N with C; terminate;
- If A is empty then label N with the most common class C in D (majority voting); terminate;
- Select attribute a belongs to A, with the highest information gain; Label N with a;
- For each value v of a:
  - Grow a branch from N with condition $a = v$;
  - Assume D_s is the subset of Dataset D with $a = v$;
  - If D_s is empty, attach a leaf labeled with the most common class in D;
  - Else attach the node generated by DecisionTree(D_s, $A - a$);

Figure 4.1: Pseudo-code for Decision Tree

Source: Quinlan (1986)

Tree model is one type of the base models that are commonly used for ensemble learning. Tree model can be very sensitive, even small perturbations in the training data can lead to very different trees. This unique property makes the tree model a good candidate for ensemble learning. In addition, the computation process of tree model is fast and easy, which can reduce model complexity and improve the efficiency.

Overfitting means that a function fits the data too well. Typically, this is because the actual equation is too complicated to consider each data point and outlier. The tree-based ensemble method can 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.

The purpose of an ensemble learning algorithm is to achieve an improved result by combining predictions of a group of individual base models. It has been shown that the combined model often generates more stable and accurate predictions in many applications (Leblanc and Tibshirani, 1996; Banfield et al., 2006).

Bagging and boosting are both ensemble techniques, where a set of base models are combined to create a model that obtains better performance than a single model. However, they utilize different re-sampling methods and therefore can have different performances and generate different outputs.

4.2.1 Bagging Algorithm

Bagging is a method for generating multiple versions of predictor and using these to get an aggregated predictor (Breiman, 1996). The bagging algorithm could help reduce the overfitting problem from a single model.

Typically, there are 3 steps to use the tree-based bagging algorithm: The first step is to create several (e.g., 100) random sub-samples of the dataset with replacement. The second step is to train a model using each sample. Finally, given a new dataset, calculate the average prediction from each model (Breiman, 1996).

In detail, the pseudo-code for bagging introduced by Breiman (1996) is shown in Figure 4.2, which can make it easier to understand the idea of bagging algorithm. Given a training set D, in each iteration (ranging from 1 to T), randomly sample with replacement N samples from the training dataset. Then train a selected base model A (e.g., decision tree model) on samples. For each test example, start with all trained base models, and then predict by combining results of all T trained models. For the regression problem, the combining rule will be averaging them; for the classification problem, the combining rule will be a majority vote.

```
Bagging (prediction algorithm A, dataset D, iterations T)

1) model generation

        for i = 1 to T:
                generate a bootstrap sample D(i) from D
                let M(i) be result of training A on D(i)

2) prediction for a given test instance x

        for i = 1 to T:
                let C(i) = output of M(i) on x

        return class that appears most often among C(1)..C(T)
```

Figure 4.2: Pseudo-code for Bagging
Source: Breiman (1996)

Random Forest is a typical bagging-based model that was introduced by Breiman (2001), and it has been widely used in the machine learning area. Random Forest is a combination of many decision trees. There are two types of randomness built into the trees. First, each tree is built on a random sample from the training dataset. Second, a subset of features are randomly allocated to each tree node to generate the best split.

The main limitation of the Random Forest is that a larger number of trees may make the model run slower. If the data include categorical variables with a different number of levels, "*Random Forests are biased in favor of those variables with more levels*" (Strickland, 2007).

4.2.2 Boosting Algorithm

The idea of boosting algorithm was first proposed by Kearns (1988). Boosting algorithm also refers to several algorithms that convert weak learners to strong learners. Several base models are combined together to form stronger model that can make generalizations (Rajsingh et al., 2018).

Different from the bagging method which has each base model run independently and then aggregates their outputs at the end without any preference, the boosting method improves the

prediction through developing multiple models in sequence by putting emphasis on these training cases that are difficult to estimate.

In detail, the initial model in boosting is predicted using a loss function. Each time a decision tree is generated, the model is updated based on the previous model and loss function resulting in a final model. The samples have an unequal probability of appearing in subsequent models and ones with the highest error appear most, which means that the incorrectly estimated or misclassified samples have more chances to be selected.

There are many boosting algorithms such as AdaBoost, Gradient boosting, and XGBoost. Gradient boosting is a typical boosting approach, and it has been widely used in the machine learning area. The word 'gradient' means that it uses a gradient descent algorithm to minimize the loss when adding new models (Friedman, 2001). The gradient boosting approach supports both classification and regression predictive modeling problems.

Based on previous studies, the gradient boosting model generally gives better results than Random Forest, since Random Forest has fewer parameters needing tuning and also is less sensitive to these parameters (Ogutu et al., 2011; Freeman et al., 2015). However, the gradient boosting model is harder to fit than Random Forests at the same time. The stopping criteria should also be chosen carefully to avoid overfitting on the training data.

4.3 XGBoost Algorithm

XGBoost is the short name for 'Extreme gradient boosting' that was proposed by Chen and Guestrin (2016). In recent years, it has a recognized impact in solving machine learning challenges in different application domains.

The speed of XGBoost is much faster than that of other common machine learning methods since it can process large amounts of data in a parallel way efficiently. The XGBoost model can also handle missing values in the dataset. Above all, "*XGBoost used a more regularized model formalization to control over-fitting, which gives it better performance*" (Chen and Guestrin, 2016). Therefore, the XGBoost model is selected and used to conduct travel time prediction in this study. The detailed information about the XGBoost model is described as follows:

The objective function (Obj(Θ)) of the XGBoost model is provided below (Chen and Guestrin, 2016):

$$Obj(\Theta) = L(\Theta) + \Omega(\Theta)$$

where,

$L(\Theta)$ = The training loss, which measures how well the model fit on training data

$\Omega(\Theta)$ = The regularization term, which measures the complexity of the model.

The loss on training data can be expressed as:

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i)$$

In detail, the square loss for the regression problem can be expressed as:

$$l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$$

The logistic loss for the classification problem can be expressed as:

$$l(y_i, \hat{y}_i) = y_i \ln\left(1 + e^{-\hat{y}_i}\right) + (1 - y_i) ln\left(1 + e^{\hat{y}_i}\right)$$

In this study,

$\hat{y}_i$ = the predicted travel time.

y_i = the actual travel time.

When a new tree is added to the model, the objective function can be transformed to:

$$Obj(\mathrm{t}) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i{}^{(t)}\right) + \sum_{i=1}^{t} \Omega(f_i) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i{}^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant$$

In order to get the simplest goal, the constant term should be removed from the function. The process of XGBoost uses second order Taylor expansion to extend the loss function and removes the constant term (Chen and Guestrin, 2016).

$$Obj(\mathrm{t}) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i{}^{(t-1)} + g_i f_t(x_i) + \frac{1}{2} h_i f_t{}^2(x_i)\right) + \Omega(f_t)$$

where,

$g_i = \partial_{\hat{y}_i{}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)})$, which means the first order partial derivative of the function

$h_i = \partial^2_{\hat{y}_i{}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)})$, which means the second order partial derivative of the function

After the removal of all the constants, the specific objective at step t becomes:

$$Obj(\mathrm{t}) = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t{}^2(x_i)] + \Omega(f_t)$$

In the XGBoost model, the complexity is defined as (Chen and Guestrin, 2016):

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

where,

T = the number of leaf nodes

γ = the penalty coefficient of the number of leaves

λ = the penalty coefficient of regularization

w_j= the score of leaf j

After re-formulating the tree model, the objective function with the t-th tree can be written as:

$$Obj(\mathrm{t}) = \sum_{i=1}^{n} [g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

$$Obj(\mathrm{t}) = \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T$$

where $I_j = \{i | q(x_i) = j\}$ is an instance set assigned to the j-th leaf. The objective function could be further compressed as:

$$Obj(\mathrm{t}) = \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2 \right] + \gamma T$$

where $G_j = \sum_{i \in I_j} g_i$, $H_j = \sum_{i \in I_j} h_i$

The best w_j one can get for the objective function is:

$$w_j^* = -\frac{G_j}{H_j + \lambda}$$

Therefore, the final objective function can be written as:

$$Obj(\mathrm{t}) = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T$$

The smaller the score is, the better the structure is.

XGBoost can also add branches for each leaf node. The loss reduction after the split can be expressed as:

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

where $\frac{G_L^2}{H_L+\lambda}$ is the score of the left node after the cut. $\frac{G_R^2}{H_R+\lambda}$ is the score of the right node after the cut. $\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$ is the score of combination without the cut. Finally, the best structure of the model can be obtained which can minimize the objective function by enumerating different kinds of tree structures.

4.4 Summary

This chapter presents the methodology which will be used in travel time prediction. The idea of ensemble learning is introduced first. The detailed information on the decision tree algorithm, bagging algorithm, and boosting algorithm is then presented. The basic information about the Random Forest model and the gradient boosting model is also introduced. The advantages and disadvantages of each model are discussed. The basic information about the XGBoost model is also presented in this chapter. The advantages of the XGBoost model are listed. The detailed process of the XGBoost model is described including the objective function, regularization terms, and model score.

Chapter 5. Travel Time Prediction Model Validation

5.1 Introduction

This chapter presents the validation of the proposed machine learning model based on the data described in Chapter 3. Section 5.2 shows the feature selection and pre-processing steps, and the features will include, but are not limited to: time of day, day of the week, month, weather conditions, and segment characteristics. Section 5.3 introduces the parameters in the model and discusses the parameters' tuning process. Finally, section 5.4 concludes the chapter with a summary.

5.2 Feature Selection and Pre-processing

Determining which feature to use in the model is the most important factor of a successful machine learning algorithm (Domingos, 2012). The definition of feature engineering is "*an act of extracting features from raw data and transforming them into the formats that are suitable for the machine learning model*" (Zheng and Casari, 2018). Therefore, the quality of the features will have great influence on whether the travel time prediction model is good or not.

The real-world travel time data provided by the RITIS website (which was mentioned in chapter 3) are used for this study. The quality of the data is precise enough with less than a 0.5% missing rate (4348 out of 906048). Therefore, this study simply replaces the missing values with the mean of its closest surrounding values.

Based on previous studies (Min and Wynter, 2011; Wang et al., 2018), the features that influence the accuracy of travel time prediction may not only include the basic features (such as time of day, day of the week, month, and weather), but also include the spatial and temporal characteristics of the segments. Therefore, the travel time information from several steps before and the travel time information on adjacent segments are also selected and will be used in the model.

Table 5.1 summarizes the basic information on the features used in this study and Table 5.2 is an example of the dataset. In Table 5.2, the first 19 columns are the input variables that are used to predict travel time at time step t and the last column is the travel time. In some cases, the target variable will be transformed when it is not normally distributed, however, *"since regression tree is the basic learner of XGBoost, there is no need to normalize samples, which means that features from different units would not affect the prediction result"* (Dong et al. 2018).

For the Categorical Variable, the most commonly used method is One-hot encoding in the Python software. One-hot encoding is a process by which categorical variables are converted into a form that could be provided to machine learning algorithms to do a better job in prediction. For example, the category weekdays with 7 variables will be transferred as dummy variables. It should be noticed that if the range of category variable is too large (over hundreds of variables), this method is not suitable anymore.

Table 5.1: Definitions and Attributes on Selected Features

Variable	Definition	Attribute
ID	Segment ID	Categorical
L	Length of the segment	Categorical
TOD	The TOD is represented by every 15-minute timestep indexed from 1 to 96	Categorical
DOW	The DOW is indexed from 1 to 7 to represent from Monday through Sunday	Categorical
Month	The Month is indexed from 1 to 12 to represent January to December	Categorical
Weather	Weather is indexed from 1 to 3 to represent normal, rain and snow/ice/fog, respectively	Categorical
T_{t-1}	Travel time at time step t-1 (15 minutes before)	Float
T_{t-2}	Travel time at time step t-2 (30 minutes before)	Float
T_{t-3}	Travel time at time step t-3 (45 minutes before)	Float
ΔT_{t-1}	Travel time change value at time step t-1 (15 minutes before)	Float
ΔT_{t-2}	Travel time change value at time step t-2 (30 minutes before)	Float
ΔT_{t-3}	Travel time change value at time step t-3 (45 minutes before)	Float
T_{t-1}^{i-1}	Travel time of first upstream segment at time step t-1 (15 minutes before)	Float
T_{t-1}^{i-2}	Travel time of second upstream segment at time step t-1 (15 minutes before)	Float
ΔT_{t-1}^{i-1}	Travel time change value of first upstream segment at time step t-1 (15 minutes before)	Float
ΔT_{t-1}^{i-2}	Travel time change value of second upstream segment at time step t-1 (15 minutes before)	Float
T_{t-1}^{i+1}	Travel time of first downstream segment at time step t-1 (15 minutes before)	Float
T_{t-1}^{i+2}	Travel time of second downstream segment at time step t-1 (15 minutes before)	Float
ΔT_{t-1}^{i+1}	Travel time change value of first downstream segment at time step t-1 (15 minutes before)	Float
ΔT_{t-1}^{i+1}	Travel time change value of second downstream segment at time step t-1 (15 minutes before)	Float
T_t	Travel time at time step t	Float

Table 5.2: Example of the Raw Inputs of the Model

ID	Weather	TOD	DOW	Month	L	T_{t-1}	T_{t-2}	T_{t-3}	...	ΔT_{t-3}	T_{t-1}^{i-1}	T_{t-1}^{i-2}	ΔT_{t-1}^{i-1}	ΔT_{t-1}^{i-2}	T_{t-1}^{i+1}	T_{t-1}^{i+2}	ΔT_{t-1}^{i+1}	ΔT_{t-1}^{i+1}	T_t
125-04790	normal	21	Friday	1	2.245989	121.91	121.54	127.10	...	-8.19	33.87	29.47	-0.31	-0.39	30.63	91.23	-0.21	1.24	116.28
125-04790	normal	22	Friday	1	2.245989	116.28	121.91	121.54	...	5.56	34.70	30.88	-0.83	-1.41	29.41	90.87	1.22	0.36	117.98
125-04790	normal	23	Friday	1	2.245989	117.98	116.28	121.91	...	-0.37	34.35	30.12	0.35	0.76	28.75	86.16	0.66	4.71	113.31
125-04790	normal	24	Friday	1	2.245989	113.31	117.98	116.28	...	5.63	31.54	27.34	2.81	2.78	27.71	83.33	1.04	2.83	111.28
125-04790	normal	25	Friday	1	2.245989	111.28	113.31	117.98	...	-1.70	31.33	26.36	0.21	0.98	27.50	82.45	0.21	0.88	108.89
125-04790	normal	26	Friday	1	2.245989	108.89	111.28	113.31	...	4.67	30.58	26.38	0.75	-0.02	27.20	83.74	0.30	-1.29	118.92

5.3 Parameter Tuning Process

In the XGBoost model, there are many parameters that should be considered. There are three types of parameters: general parameters, booster parameters and task parameters.

General parameters are related to which booster is being used to do boosting, commonly in the tree or linear models. In detail, the general parameters include:

- **Booster:** Select the type of model to run at each iteration. It has 2 options: tree-based models and linear models. The default value of booster is '*gbtree*'.

- **Silent:** Silent controls whether to print message. If the value is set to 1, no running messages will be printed. The default value of silent is 0. It is generally good to keep it as 0 since the messages might help in understanding the model.

- **Nthread:** This parameter is used for controlling the parallel processing and the number of cores in the system that would be used. The default value is the maximum number of threads available on the computer. The algorithm will detect it automatically.

Booster parameters depend on which booster one has chosen. For the tree booster in this study, the parameters include:

- **Learning rate:** Learning rate is the rate at which the model learns patterns in data. After every round, it shrinks the feature weights to reach the best optimum. Lower learning rate leads to slower computation. The default value is 0.3.

- **Gamma:** Gamma controls regularization (or prevents overfitting). The optimal value of gamma depends on the data set and other parameter values. The larger the gamma is, the more conservative the algorithm will be. The value of Gamma usually is 0. The default value is 0.

- **Max_depth:** Maximum depth controls the depth of the tree. The larger the depth, the more complex the model, and the higher the chance of overfitting. There is no standard value for max_depth. Larger dataset requires deeper tree to learn the rules from data. The value of Max_depth usually ranges from 3 to 10. The default value is 6.

- **Min_child_weight:** Minimum child weight refers to the minimum number of instances required in a child node. It blocks the potential feature interactions to prevent overfitting. The default value is 1.

- **Subsample:** Percentage of samples used per tree. This parameter will also help to prevent overfitting. The value of subsample usually ranges from 0.5 to 1. The default value is 1.

- **Colsample_bytree:** Percentage of features used per tree. A high value can lead to overfitting. The value of colsample_bytree usually ranges from 0.5 to 1. The default value is 1.

- **Lambda:** This parameter can help to handle the regularization part of the XGBoost model. Usually, the value of Lambda is 1 and the default value is 1.

- **Alpha:** This parameter can also help to handle the regularization part of the XGBoost model. The value of Alpha usually is 0 and the default value is 0.

- **N_estimators:** This parameter refers to the number of trees one wants to build in the model. The number is up to the complexity of the model.

Task parameters depend on the learning scenario. For example, regression tasks may use different parameters with ranking tasks. The task parameters include:

- **Objective:** This parameter defines the task of learning (the loss function to be minimized). The mostly used values are 'reg:linear', 'binary:logistic', 'multi:softmax' and 'multi:softprob'. The default value is 'reg:linear'.

In order to optimize the modelling result, it is necessary to explore the effect of different combinations of parameters on the model performance. Based on previous studies (Zhang and Haghani, 2015; Dong et al. 2018), the parameters that could be optimized include, but are not limited to: N_estimators (number of trees), learning rate, and Max_depth (maximum depth of the tree). Therefore, these parameters are considered to be optimized in this study.

There are several optimization methods considered in previous studies and the grid search method is the most widely used one. Therefore, the grid search method is selected as the optimization method with the consideration of time-efficiency. In this study, 80% of the traffic data is used as training data and 20% of the data is used as the testing data. The XGBoost model is fitted with a different number of trees (N_estimators ranges from 1 to 500), maximum depth (Max_depth ranges from 5 to 10) and learning rates (Learning_rate ranges from 0.1 to 0.5). The number of stopping rounds is set as 50, which means stopping iteration after 50 rounds when there is no performance improvement.

Figure 5.1 to Figure 5.6 below show the effects of different selected variables on the prediction results. Table 5.3 below presents the detailed prediction results including the prediction results at each step, computation time, and optimized results. The mean absolute error (MAE) is used to evaluate the performance of the model.

The equation of the MAE is provided below:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|$$

where,

m = The total number of the data.

y_i = The actual travel time value in the test dataset of record i.

$\hat{y}_i$= The predicted travel time value in the test dataset of record i.

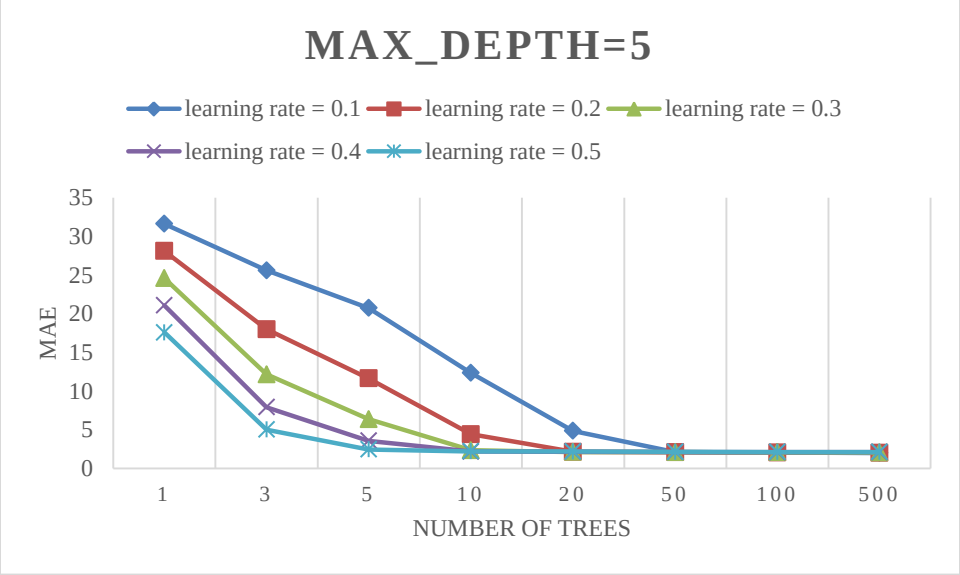

Figure 5.1: XGBoost Travel Time Prediction Model Outputs with the Max_depth =5

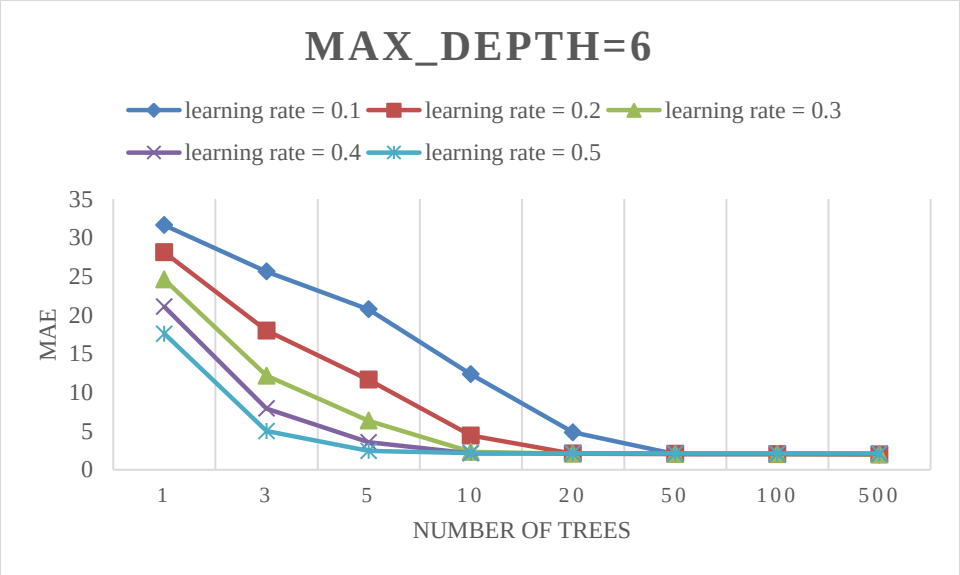

Figure 5.2: XGBoost Travel Time Prediction Model Outputs with the Max_depth=6

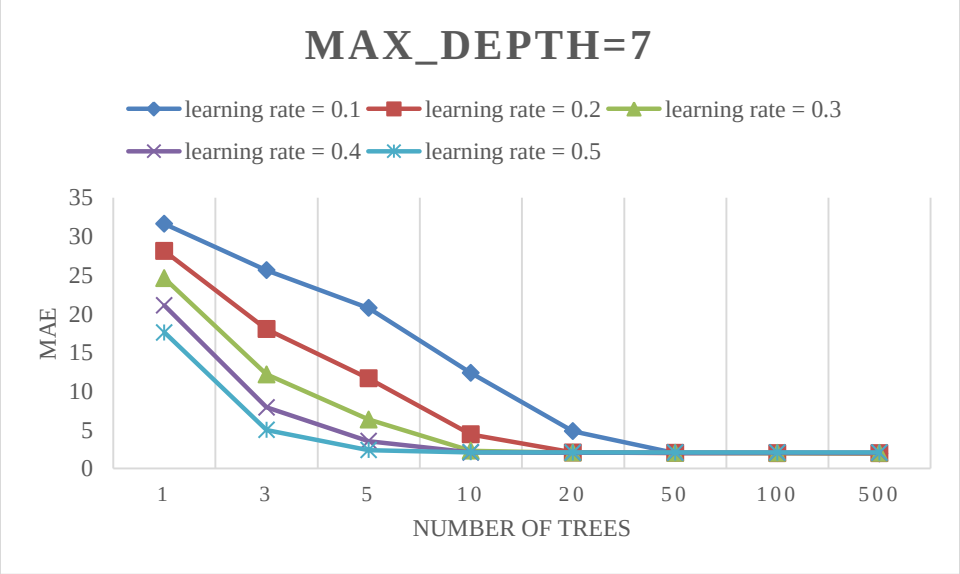

Figure 5.3: XGBoost Travel Time Prediction Model Outputs with the Max_depth=7

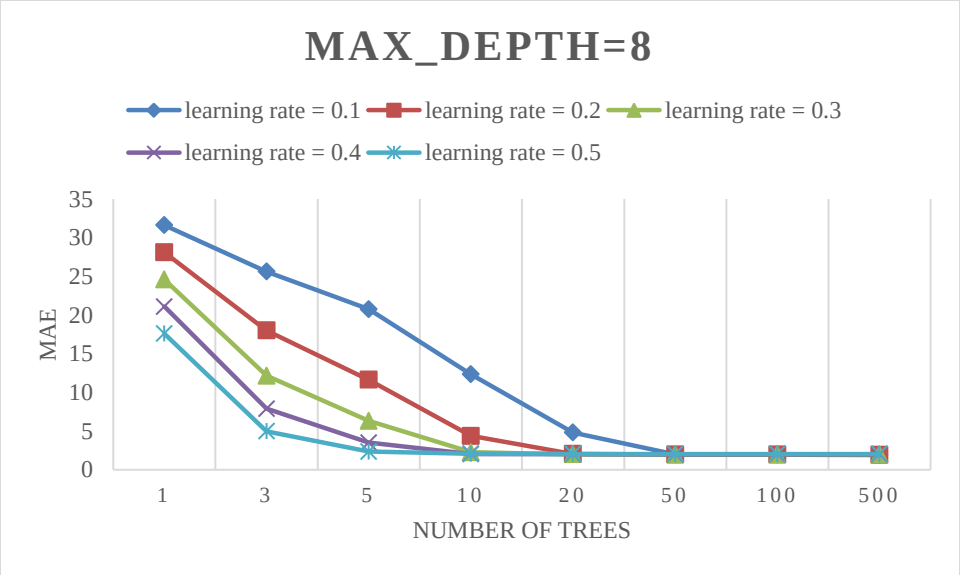

Figure 5.4: XGBoost Travel Time Prediction Model Outputs with the Max_depth=8

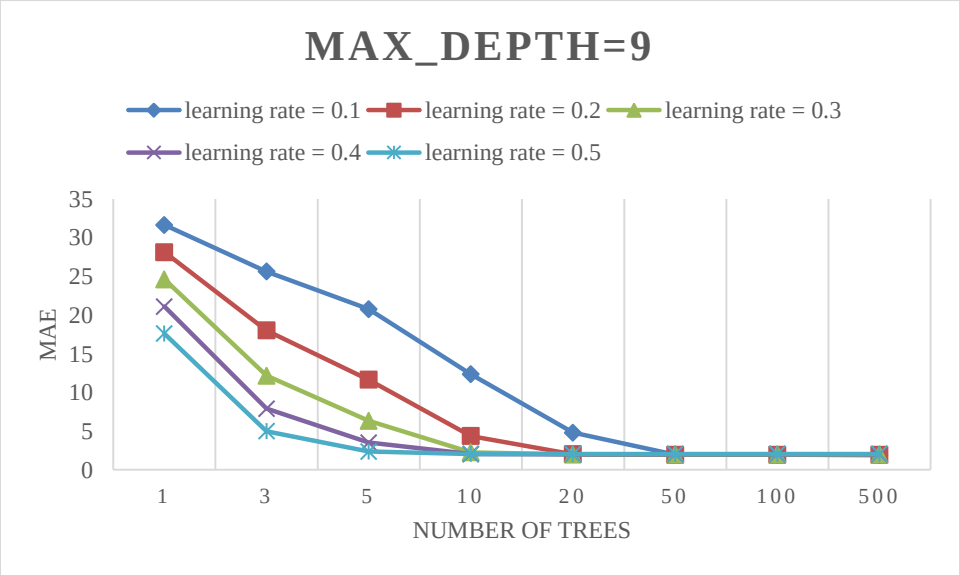

Figure 5.5: XGBoost Travel Time Prediction Model Outputs with the Max_depth=9

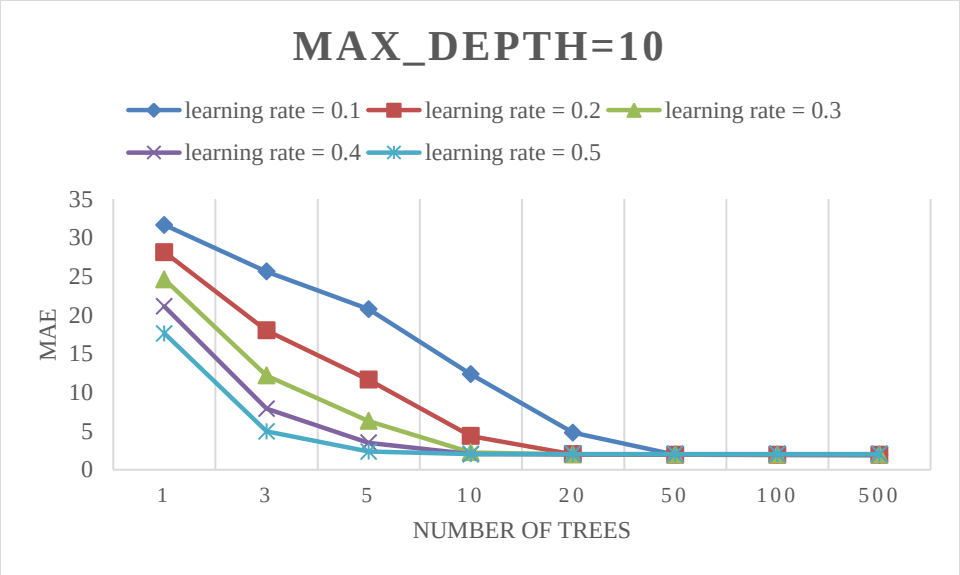

Figure 5.6: XGBoost Travel Time Prediction Model Outputs with the Max_depth=10

Based on Figure 5.1 to Figure 5.6 above, it can be concluded that the MAE value decreases as the number of trees increases, and the slopes of different learning rates are also different. In general, the lower the learning rate is, the higher the initial MAE value (with the number of trees = 1) will be. For example, when the learning rate equals to 0.1, the initial MAE value is about 36.2. In comparison, the figures show that the MAE values are about 17.6 when the number of trees is 1 and the learning rate is 0.5.

Based on the figures above, when the number of trees reaches 50, the value of MAE becomes nearly the same. However, the data in Table 5.3 indicate that the results can still be optimized a little bit if the number of trees keeps increasing. Overfitting is a general problem of traditional ensemble learning methods. For example, the prediction error usually increases when the number of trees increases after it reaches the optimized point in the gradient boosting model (Zhang and Haghani, 2015). In the XGBoost model, the overfitting problem can be solved as the iteration will be stopped when there is no performance improvement after 50 iterations. Therefore, the value 'NA' in Table 5.3 means that the computation already stopped before the number of trees reached those values.

It could be seen that the parameter max_depth does not influence the prediction results significantly since the trends of the errors are nearly the same. However, the data in Table 5.3 shows that as the max_depth increases, the MAE decreases a little bit (the optimized MAEs of max_depth from 5 to 10 are 2.02, 1.98, 1.95, 1.93, 1.91, 1.90, respectively). The data in Table 5.4 shows that as the max_depth increases, the average computation time of the model also decreases a lot, which means the larger value of max_depth can not only increase the accuracy of the model a little bit but also increase the efficiency.

Table 5.3: MAEs of Different Learning Rates, Number of Trees and Max_depth

Learning rate	MAE							
Max_depth=5								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6232	25.6101	20.7441	12.3545	4.86309	2.1066	2.08573	2.01681
0.2	28.105	17.9992	11.6453	4.43742	2.16435	2.11039	2.08105	2.03219
0.3	24.5887	12.1685	6.37425	2.36127	2.14303	2.11073	2.09237	2.05376
0.4	21.0772	7.91567	3.5855	2.19242	2.1688	2.13987	2.11298	NA
0.5	17.5817	5.01915	2.47101	2.22655	2.20399	2.16189	2.13814	NA
Max_depth=6								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6239	25.6113	20.7459	12.352	4.84463	2.05099	2.03103	1.97875
0.2	28.1064	17.9985	11.6379	4.42059	2.09807	2.05496	2.02066	1.98881
0.3	24.5905	12.1655	6.35543	2.32393	2.09369	2.06605	2.04824	2.02001
0.4	21.0791	7.9191	3.55346	2.12642	2.11301	2.08441	2.06326	NA
0.5	17.5816	5.01106	2.4425	2.16502	2.14012	2.11357	2.11321	NA
Max_depth=7								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6246	25.6126	20.7487	12.3503	4.83108	2.01236	1.9901	1.95178
0.2	28.1076	18.0042	11.6389	4.40351	2.0696	2.01503	1.997	1.97402
0.3	24.5923	12.1681	6.33938	2.29773	2.05536	2.02392	2.01867	2.00145
0.4	21.0818	7.90891	3.52957	2.07265	2.06057	2.04777	2.0422	NA
0.5	17.5864	4.99475	2.39171	2.08885	2.07452	2.07155	2.06401	NA
Max_depth=8								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6262	25.6164	20.7538	12.3515	4.82079	1.98521	1.96533	1.92991
0.2	28.1107	18.0094	11.6371	4.38646	2.04197	1.984	1.968	1.94855
0.3	24.5969	12.1703	6.33313	2.28103	2.01744	1.99892	1.99763	NA
0.4	21.0879	7.90641	3.51711	2.04708	2.02954	2.01933	2.0176	NA
0.5	17.5936	4.9815	2.36822	2.07031	2.05874	2.06077	NA	NA
Max_depth=9								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6291	25.6203	20.7573	12.3507	4.80693	1.96347	1.94109	1.91498
0.2	28.1168	18.0162	11.6391	4.3746	2.01152	1.95592	1.94509	1.93497
0.3	24.606	12.1708	6.32246	2.25596	1.99238	1.97705	1.97125	NA
0.4	21.0991	7.90241	3.51192	2.02939	2.01599	2.01326	NA	NA
0.5	17.6059	4.97653	2.35518	2.0485	2.03841	2.05002	NA	NA

Learning rate	MAE							
Max_depth=10								
	Number of trees							
	1	3	5	10	20	50	100	500
0.1	31.6307	25.6248	20.7631	12.352	4.80107	1.94157	1.91908	1.89544
0.2	28.1198	18.0198	11.6369	4.37085	2.00348	1.94777	1.94338	NA
0.3	24.6101	12.1721	6.31815	2.24795	1.98632	1.97365	1.97672	NA
0.4	21.1044	7.89797	3.50731	2.02597	2.01108	2.01454	NA	NA
0.5	17.6118	4.96814	2.3521	2.03704	2.04147	2.05903	NA	NA

Table 5.4: Optimized Prediction Results and Computation Times

Learning rate	Optimized Result (MAE)	Number of Iterations	Computation Time
Max_depth =5			
0.1	2.01681	500	25 mins
0.2	2.03219	500	25 mins
0.3	2.05376	500	25 mins
0.4	2.079	481	23 mins
0.5	2.11782	217	9 mins
Max_depth =6			
0.1	1.97875	500	25 mins
0.2	1.98881	500	25 mins
0.3	2.02001	500	25 mins
0.4	2.05099	405	20 mins
0.5	2.10818	107	5 mins
Max_depth =7			
0.1	1.95178	500	25 mins
0.2	1.97402	500	25 mins
0.3	2.00145	500	25 mins
0.4	2.03456	231	12 mins
0.5	2.06401	81	4 mins
Max_depth =8			
0.1	1.92991	500	25 mins
0.2	1.94855	500	25 mins
0.3	1.99435	281	17 mins
0.4	2.0176	98	6 mins
0.5	2.05619	73	4 mins
Max_depth =9			
0.1	1.91498	500	25 mins
0.2	1.93497	500	25 mins
0.3	1.96895	167	8 mins
0.4	2.01224	80	4 mins
0.5	2.03841	70	4 mins
Max_depth =10			
0.1	1.89544	500	25 mins

Learning rate	Optimized Result (MAE)	Number of Iterations	Computation Time
0.2	1.93876	352	18 mins
0.3	1.97233	156	8 mins
0.4	2.00963	74	4 mins
0.8	2.03704	60	4 mins

According to the experimental results, it can be concluded that:

The accuracy level of slower learning rate with a larger number of trees in the model is higher than that of a faster learning rate with a smaller number of trees. The number of trees needed to get optimized result for the model with faster learning rate is also lower than those with slower learning rates.

There is also a need to consider the tradeoff between prediction accuracy and computational time. Since a large number of trees is being fitted, model complexity also increases and requires more computational time. Therefore, the selection of the parameters such as max_depth and number of stopping round is important in the real world.

In addition, the maximum depth of the tree also affects the optimized selection. When the learning rates and number of trees are the same, a higher maximum depth of the tree leads to the lower error rates. A higher max_depth is also more efficient than a lower value since the number of iterations needed to achieve optimized results is lower. In general, a higher max_depth value means a more complex tree model and requires fewer trees to be fitted with a given learning rate.

5.4 Summary

This chapter describes the validation process of the XGBoost-based travel time prediction model. The detailed information about the input features is presented. The parameters of the XGBoost model are also introduced. In order to achieve a better model performance, the parameter tuning process is discussed. The experimental results could give a clear picture of how the analyzed parameters impact the prediction performance.

Chapter 6. Prediction Results Analysis

6.1 Introduction

This chapter presents the evaluation of the proposed XGBoost model based on the results described in Chapter 5. Section 6.2 presents the analysis of the optimized prediction results from XGBoost model. Section 6.3 presents the performance comparison between the XGBoost model and gradient boosting model. Finally, section 6.4 concludes this chapter with a summary.

6.2 Modelling Results Analysis

In machine learning area, the predictor variables, which are the features mentioned in Chapter 5, usually have significant impacts on the prediction results. Exploring the influence on the individual feature can help understand the data better. Higher relative importance indicates a stronger influence in predicting travel time.

Table 6.1 presents the relative importance of each feature in the optimized XGBoost model. Each predictor variable has a different impact on the predicted travel time. Based on the importance rank of each variable, it can be found that the variable T_{t-1}, which is the travel time at time step t-1 (15 minutes before), contributes the most to the predicted travel time. This result is expected and consistent with a previous study (Zhang and Haghani, 2015), which demonstrates that the immediate previous traffic condition will influence the traffic condition in the future. Therefore, this feature T_{t-1} is the most important and highly correlated with the prediction value.

The results in Table 6.1 show that time of day is the second ranked variable with the relative importance value of 34.85%, and this result is also expected. As mentioned by other studies, the travel time variability is also highly correlated with the time of day. The travel time usually increases a lot during peak hours and becomes stable during non-peak hours.

The third ranked variable is the segment ID with the relative importance value of 12.65%. The potential reason behind this ranking could be that the segment ID indicates which segment it is. The segment ID contains a lot of potential information such as the geographic location of the segment. Based on the travel time variability analysis results of other studies, different segment locations contribute to different travel time variability characteristics. Therefore, the segment ID is also a necessary and important feature in the model.

Day of week is the 4th ranked variable in the model; the relative importance value of day of week is 3.76%. The variable day of week is also important in the model since the travel time is highly correlated with which day of the week it is. Based on previous studies, the traffic congestion on weekends is less frequent than on weekdays (Chen et al., 2017, Chen et al., 2018). The travel time during peak hours on Friday is usually higher than those on other weekdays (Wang et al., 2017). Therefore, the variable day of

week is important in the model; this result is consistent with a previous study (Zhang and Haghani, 2015).

Weather is also considered in the model with a relative importance value of 1.72%. Based on the results of other studies, inclement weather conditions may have a drastic impact on travel time variability. Therefore, the weather information is also useful in travel time prediction as adverse weather usually increases travel time. This finding is consistent with previous studies (Koesdwiady et al., 2016; Qiao et al., 2016).

The travel time at time step t-1 (15 minutes before) is not the only variable with the consideration of temporal correlation. Several variables such as the travel time of the two steps and three steps ahead (with the relative importance value of 0.40% and 0.33%, respectively) and the travel time change value of the three time steps ahead (with the relative importance value of 0.24%, 0.47% and 0.27%, respectively) are considered in the model. These variables are also used in the model of previous studies which had used gradient boosting models to predict freeway travel time (Zhang and Haghani, 2015; Cheng et al., 2018). The time change variables are considered in this study because they could indicate the travel time change trends of the segments. However, the influences of these variables are relatively small. The outcome is similar to the outcome of a previous study (Cheng et al., 2018).

With the consideration of spatial impact, several variables such as the travel time of the two upstream segments (with the relative importance value of 0.29% and 0.40%, respectively) and the travel time of the two downstream segments (with the relative importance value of 0.26% and 0.60%, respectively) one time-step ahead are considered in the model. With respect to the travel time change value, the relative importance values of the two upstream segments are both 0.28%, and the relative importance values of the two upstream segments are 0.36% and 0.69%, respectively. Based on these results, it could be found that the relative importance values of the downstream segments are higher than those of upstream segments. It could be explained by the spatial characteristics of the roadway. If a bottleneck occurs at the downstream segment, the upstream segment will be influenced shortly.

Table 6.1: Relative Importance of Each Variable and Their Ranks in the Model

Variable	Relative Importance (%)	Rank
ID	12.65	3
L	0.24	19
TOD	34.85	2
DOW	3.76	4
Month	2.10	5
Weather	1.72	6
T_{t-1}	38.87	1

Variable	Relative Importance (%)	Rank
T_{t-2}	0.40	10
T_{t-3}	0.33	13
ΔT_{t-1}	0.24	19
ΔT_{t-2}	0.47	9
ΔT_{t-3}	0.27	17
T_{t-1}^{i-1}	0.29	14
T_{t-1}^{i-2}	0.40	10
ΔT_{t-1}^{i-1}	0.28	15
ΔT_{t-1}^{i-2}	0.28	15
T_{t-1}^{i+1}	0.26	18
T_{t-1}^{i+2}	0.60	8
ΔT_{t-1}^{i+1}	0.36	12
ΔT_{t-1}^{i+1}	0.69	7

6.3 Model Comparison

In order to examine the accuracy and effectiveness of the XGBoost model, this section comprehensively evaluates the modeling results of the XGBoost model and compares the results with those of the gradient boosting model. The prediction result of the gradient boosting model is also optimized using a grid search method. For clarity, the mean absolute percentage error (MAPE) is used to evaluate and compare the performance of the two models.

The equation of the MAPE is provided below:

$$MAPE = \frac{100\%}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

where,

m = The total number of the data.

y_i = The actual travel time value in the test dataset of record i.

$\hat{y}_i$= The predicted travel time value in the test dataset of record i.

Table 6.2 below presents the comparison between prediction results of the optimized XGBoost model and gradient boosting model. Based on the comparison, it could be concluded that the XGBoost model outperforms the gradient boosting model with both the consideration of accuracy and efficiency. The potential reason behind this could be as follows:

In general, the XGBoost model is a more regularized form of the gradient boosting model. XGBoost uses advanced regularization terms, which improve model generalization capabilities. Therefore, the prediction results of the XGBoost model is more accurate than those of the gradient boosting model. At the same time, the computation time of the XGBoost model (25 mins) is much faster than that of the gradient boosting model (2 hours). One important reason behind the better performance of the XGBoost model could be the parallel processing function. The gradient boosting model is extremely difficult to parallelize since it has sequential characteristics. In comparison, XGBoost can allow us to do the boosting work using distributed processing engines.

Another key reason is the XGBoost model implements the early stopping function, which means that one can stop model assessment when additional trees (see Chapter 5) offer no improvement to the prediction results. This function can help us not only prevent overfitting problem, but also improve the efficiency of the model significantly.

Table 6.2: Performance Comparison between XGBoost Model and Gradient Boosting Model

Number of Trees	MAPE XGBoost (%)	MAPE Gradient Boosting (%)
3	14.64	35.10
10	5.22	24.33
20	5.22	16.78
50	4.87	13.56
100	4.82	11.11
200	4.74	9.38
500	4.72	5.67
Average computation time	11.8 mins	Over one hour

6.4 Summary

This chapter describes the numerical results of the developed XGBoost model. The relative importance of each variable in the model is presented and interpreted. In order to examine the accuracy and effectiveness of the proposed model, this chapter also evaluates the optimized modeling results of the proposed XGBoost travel time prediction

model and compares them with those of the gradient boosting model. The results demonstrate that the developed XGBoost travel time prediction model significantly improves the computation accuracy and efficiency.

Chapter 7. Summary and Conclusions

7.1 Summary

Travel time is an important performance measure for assessing freeway traffic conditions and the extent of highway congestion. Anonymous vehicle probe data is a reliable source for freeway travel time analysis since it greatly improves both data coverage and data fidelity. With the development of machine learning technologies, various novel algorithms have been developed during recent years (Jordan and Mitchell, 2015). Typically, these new technologies aim to increase the accuracy and efficiency of the data prediction. One of the representative technologies is the XGBoost model. In recent years, the XGBoost model has gained popularity by winning many data science competitions (e.g., Kaggle competition). Therefore, the XGBoost model has the potential to be applied in transportation-related data analysis fields such as traffic flow, travel speed and travel time prediction.

The primary objective of this research is to develop a methodology for conducting the XGBoost model-based travel time prediction. A real-world freeway corridor is selected as the case study to examine the XGBoost prediction model so that the gaps between the theoretical research and the application of the developed model can be bridged.

The rest of this chapter is organized as follow: Section 7.2 presents a summary of conclusions of the numerical results derived from the proposed XGBoost travel time prediction model; Section 7.3 gives a brief discussion of the limitations of the current approaches and provides future research directions.

7.2 Summary and Conclusions of Travel Time Prediction Results

Regarding the travel time prediction, it is found that the XGBoost model can provide reliable prediction results. The relationships between several important parameters in the model (e.g. number of trees, learning rate, and maximum depth of the tree) are discussed in this study. In detail, the accuracy level of a slower learning rate with a larger number of trees in the model is higher than that of a faster learning rate with a smaller number of trees. A higher max_depth value is also more efficient than a lower value since the number of iterations needed to achieve optimized results is smaller.

The relative importance of the features shows that the travel time one step ahead (15 minutes before) contributes the most to the predicted travel time. Features such as the time of day, day of the week and weather also have higher relative importance values in the model than other features.

The proposed XGBoost-based travel time prediction method has considerable advantages over the gradient boosting approach. The performance evaluation result shows that the XGBoost-based model can have better outcomes in terms of both prediction accuracy and efficiency.

7.3 Future Work Directions

Typically, the XGBoost-based travel time prediction model can provide reliable results with low error rates. However, the impacts of accidents and roadworks on travel time prediction are also worth exploring. In the future, how to incorporate these features in the model will be studied if the data can be made available.

Furthermore, the performance of the travel time prediction model is discussed under all conditions as a whole. In the future, the performances of the model under different traffic conditions (such as both non-congested and congested conditions) can be learned and compared.

References

1. Wu, C.H., Ho, J.M. and Lee, D.T., (2004). "Travel-Time Prediction with Support Vector Regression." *IEEE Transactions on Intelligent Transportation Systems*, *5*(4), pp.276-281.

2. Park, D. and Rilett, L.R., (1999). "Forecasting Freeway Link Travel Times with a Multilayer Feedforward Neural Network." *Computer‐Aided Civil and Infrastructure Engineering*, *14*(5), pp.357-367.
3. Van Lint, J., Hoogendoorn, S. and Van Zuylen, H. (2002), "Freeway Travel Time Prediction with State-Space Neural Networks: Modeling State-Space Dynamics with Recurrent Neural Networks." *Transportation Research Record: Journal of the Transportation Research Board,* 1811, pp.30-39.
4. Wisitpongphan, N., Jitsakul, W. and Jieamumporn, D., (2012), "Travel Time Prediction Using Multi-Layer Feed Forward Artificial Neural Network." In *2012 Fourth International Conference on Computational Intelligence, Communication Systems and Networks*. pp. 326-330.
5. Zheng, F. and Van Zuylen, H., (2013). "Urban Link Travel Time Estimation Based on Sparse Probe Vehicle Data." *Transportation Research Part C: Emerging Technologies*, *31*, pp.145-157.
6. Duan, Y., Lv, Y. and Wang, F.Y., 2016, November. Travel time prediction with LSTM neural network. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* pp. 1053-1058
7. Liu, Y., Wang, Y., Yang, X. and Zhang, L., (2017), "Short-Term Travel Time Prediction by Deep Learning: A Comparison of Different LSTM-DNN Models." In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* pp. 1-8
8. Wang, D., Zhang, J., Cao, W., Li, J. and Zheng, Y., (2018). "When Will You Arrive? Estimating Travel Time Based on Deep Neural Networks." In *Thirty-Second AAAI Conference on Artificial Intelligence*.
9. Wang, Z., Fu, K. and Ye, J., (2018). "Learning to Estimate the Travel Time." In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* pp. 858-866
10. Wei, W., Jia, X., Liu, Y. and Yu, X., (2018). "Travel Time Forecasting with Combination of Spatial-Temporal and Time Shifting Correlation in CNN-LSTM Neural Network." In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data* pp. 297-311.
11. Yu, B., Wang, H., Shan, W. and Yao, B., (2018). "Prediction of Bus Travel Time Using Random Forests Based on Near Neighbors." *Computer‐Aided Civil and Infrastructure Engineering*, *33*(4), pp.333-350.
12. Moonam, H.M., Qin, X. and Zhang, J., (2019). "Utilizing Data Mining Techniques To Predict Expected Freeway Travel Time From Experienced Travel Time. *Mathematics and Computers in Simulation*, *155*, pp.154-167.
13. Myung, J., Kim, D.K., Kho, S.Y. and Park, C.H., (2011). "Travel Time Prediction Using K-Nearest Neighbor Method with Combined Data from Vehicle Detector System and

Automatic Toll Collection System." *Transportation Research Record: Journal of the Transportation Research Board*, 2256, pp.51-59.
14. Hamner, B., (2010). "Predicting Travel Times with Context-Dependent Random Forests by Modeling Local and Aggregate Traffic Flow." In *2010 IEEE International Conference on Data Mining Workshops* pp. 1357-1359
15. Zhang, Y. and Haghani, A., 2015. "A Gradient Boosting Method to Improve Travel Time Prediction." *Transportation Research Part C: Emerging Technologies*, 58, pp.308-324.
16. Li, X. and Bai, R., (2016). "Freight Vehicle Travel Time Prediction Using Gradient Boosting Regression Tree." In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)* pp. 1010-1015
17. Fan, S.K.S., Su, C.J., Nien, H.T., Tsai, P.F. and Cheng, C.Y., (2018). "Using Machine Learning and Big Data Approaches To Predict Travel Time Based on Historical and Real-Time Data from Taiwan Electronic Toll Collection." *Soft Computing*, *22*(17), pp.5707-5718.
18. Gupta, B., Awasthi, S., Gupta, R., Ram, L., Kumar, P., Prasad, B.R. and Agarwal, S., (2018). "Taxi Travel Time Prediction Using Ensemble-Based Random Forest and Gradient Boosting Model." In *Advances in Big Data and Cloud Computing* pp. 63-78.
19. Quinlan, J.R. (1986). "Induction of Decision Trees." *Machine learning*, 1(1), pp.81-106.
20. Han, J., Pei, J. and Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.
21. LeBlanc, M. and Tibshirani, R. (1996). "Combining Estimates in Regression and Classification." *Journal of the American Statistical Association*, 91(436), pp.1641-1650.
22. Banfield, R.E., Hall, L.O., Bowyer, K.W. and Kegelmeyer, W.P. (2006). "A comparison of decision tree ensemble creation techniques." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1), pp.173-180.
23. Breiman, L. (1996). "Bagging predictors." *Machine learning*, 24(2), pp.123-140.
24. Breiman, L. (2001). "Random forests." *Machine learning*, 45(1), pp.5-32.
25. Strickland, J. (2015). Predictive Analytics Using R. Lulu. com.
26. Kearns, M. (1988). "Thoughts on Hypothesis Boosting." *Unpublished Manuscript*, 45, p.105.
27. Rajsingh, E.B., Veerasamy, J., Alavi, A.H. and Peter, J.D. (2018). Advances in Big Data and Cloud Computing (Vol. 645). Springer.
28. Freeman, E.A., Moisen, G.G., Coulston, J.W. and Wilson, B.T. (2015). "Random Forests and Stochastic Gradient Boosting for Predicting Tree Canopy Cover: Comparing Tuning Processes and Model Performance." *Canadian Journal of Forest Research*, 46(3), pp.323-339.
29. Friedman, J.H. (2001). "Greedy Function Approximation: A Gradient Boosting Machine." *Analysis of Statistics*, pp.1189-1232.
30. Ogutu, J.O., Piepho, H.P. and Schulz-Streeck, T. (2011). A Comparison of Random Forests, Boosting and Support Vector Machines For Genomic Selection. In BMC proceedings (Vol. 5, No. 3, p. S11).
31. Chen, T. and Guestrin, C. (2016). "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785-794.
32. Domingos, P.M. (2012). "A Few Useful Things to Know About Machine Learning." *Communications of ACM*, 55(10), pp.78-87.
33. Zheng, A. and Casari, A. (2018). Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. O'Reilly Media, Inc.

34. Min, W. and Wynter, L. (2011). "Real-Time Road Traffic Prediction with Spatio-Temporal Correlations." *Transportation Research Part C: Emerging Technologies*, 19(4), pp.606-616.
35. Wang, Y., Zhang, Y., Piao, X., Liu, H. and Zhang, K. (2018). "Traffic Data Reconstruction via Adaptive Spatial-Temporal Correlations." *IEEE Transactions on Intelligent Transportation Systems*, 20(4), pp.1531-1543.
36. Dong, X., Lei, T., Jin, S. and Hou, Z. (2018). "Short-Term Traffic Flow Prediction Based on XGBoost." In 2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS). pp. 854-859
37. Chen, P., Tong, R., Lu, G., and Wang, Y. (2018). "Exploring Travel Time Distribution and Variability Patterns Using Probe Vehicle Data: Case Study in Beijing." *Journal of Advanced Transportation*, Article 3747632.
38. Chen, X. M., Chen, X., Zheng, H., and Chen, C. (2017). "Understanding network travel time reliability with on-demand ride service data." *Frontiers of Engineering Management*, 4(4), pp. 388-398.
39. Wang, Z., Goodchild, A., and McCormack, E. (2017), "A Methodology for Forecasting Freeway Travel Time Reliability Using GPS Data." *Transportation Research Procedia*, 25, pp. 842-852.
40. Koesdwiady, A., Soua, R. and Karray, F. (2016). "Improving Traffic Flow Prediction with Weather Information in Connected Cars: A Deep Learning Approach." *IEEE Transactions on Vehicular Technology*, 65(12), pp.9508-9517.
41. Qiao, W., Haghani, A., Shao, C.F. and Liu, J. (2016). "Freeway Path Travel Time Prediction Based on Heterogeneous Traffic Data Through Nonparametric Model." *Journal of Intelligent Transportation Systems*, 20(5), pp.438-448.
42. Cheng, J., Li, G. and Chen, X. (2018). "Research on Travel Time Prediction Model of Freeway Based on Gradient Boosting Decision Tree." *IEEE Access*, 7, pp.7466-7480.
43. Jordan, M.I. and Mitchell, T.M. (2015). "Machine Learning: Trends, Perspectives, and Prospects." *Science*, 349(6245), pp.255-260.