2016-01-03

Forecasting Bike Rental Demand Using New York Citi Bike Data

Wen Wang
Dublin Institute of Technology

DUBLIN INSTITUTE OF TECHNOLOGY

MASTERS THESIS

Forecasting bike rental demand using New York Citi Bike data

Author:

Wen Wang

Supervisor:

Andrea Curley

A thesis submitted in fulfilment of the requirements

for the degree of MSc. In Computing (Data Analytics)

in the

School of Computing

College of Science of Health

January 2016

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the test of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed: Wen Wang

Date: ***2-Jan-2016***

ABSTRACT

The idea of this project is from a Kaggle competition "Bike Sharing Demand"[①] which provides dataset of Capital Bikeshare in Washington D.C. and asked to combine historical usage patterns with weather data in order to forecast bike rental demand. This dissertation will extend this work, working with a broader range of project not only just focusing on the phrase of model building but all phases of KDD (Knowledge Discovery in Databases).

This dissertation focuses on Citi Bike which is one of the biggest bike share projects in the world, collects Citi Bike data, weather data and holiday data from three different databases, and integrates the data to a model ready format. Four basic predictive models are built and compared using multiple modelling algorithms, five techniques are used to enhance the accuracy of random forest model, and the final model's RMSLE (with 10-fold cross validation) decreases from 0.499 to 0.265.

This paper learns many experience from case study of Kaggle Bike Sharing Demand, and seek to build optimize predictive model with smallest error rate. This project generally answers a question of "How many bikes will meet users' demand in a future certain time", the future work of this project will be to focus on each docking station's activity. The realistic meaning of this dissertation is to provide an overview solution for bike rebalance problem, and helps to better manage Citi Bike program.

Key words: bike share, Citi Bike, predictive modelling, random forest

[①] https://www.kaggle.com/c/bike-sharing-demand

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my supervisor, Andrea Curley, for all her guidance and help during the whole dissertation.

I would like to thank all my DT228A teachers and my dear classmates in DIT, for the wonderful experience during my Master study.

Finally, I want to say thank you to my parents and my husband, for all the support and encouragement you have given to me.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. INTRODUCTION

Overview of Project Area

The idea of this project is from a Kaggle competition "Bike Sharing Demand". Kaggle is the world's largest community for predictive modelling and analytics competitions[①], researchers and companies can put their data and demand on this platform, and data analysts from all over the world compete to solve the problems using their data mining skills. Bike Sharing Demand[②] provides dataset of Capital Bikeshare in Washington D.C. and asked to combine historical usage patterns with weather data in order to forecast bike rental demand.

This dissertation will extend this work, working with a broader range of project not only just focusing on the phrase of model building but all phases of KDD (Knowledge Discovery in Databases), examine the impact of factors such as weather on bike user activity, and seek to build optimize predictive model with smallest error rate.

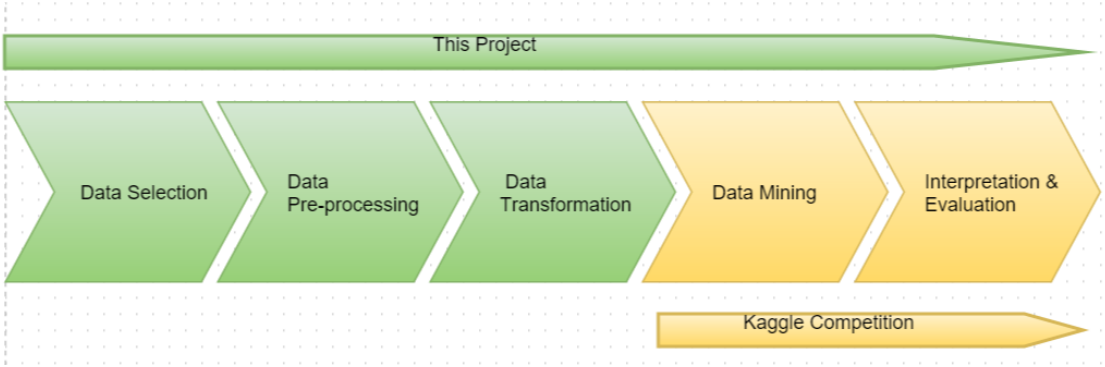


Figure 1 KDD process of Kaggle competition and this project

Citi Bike is a privately owned bike-share program in New York which opened to the public in May 2013[③], this project chooses Citi Bike program as the study object, not only because it is one of the largest bike share programs in the world[④], but also there are many problems and customer complains reported recently.

① https://www.kaggle.com/solutions/competitions

② https://www.kaggle.com/c/bike-sharing-demand

③ http://abc7ny.com/archive/9117001/

④ https://www.citibikejc.com/

This project will focus on the bike share program's rebalancing issue, aiming to answer a question: "How many bikes will meet users' demand in a future certain time." The aim of this project is to build mathematical models to forecast bike rental demand by combining historical usage patterns with the related information of users, weather, holiday and weekend.

To solve the bike rebalancing issue, this dissertation will finish the first part of the solution which provides an overview of total Citi Bike rental demand. The second part (details in Chapter 5.5) is to explore each docking station's rental demand. The work need to be taken is only to replace the total Citi Bike historical data with each docking station's data in the predictive model built by this project.

1.1 Background

Currently, there are over 500 bike-sharing programs around the world (Larsen, 2013) and the number keeps on increasing. More and more people prefer to rent bikes because it is a cheap, convenient, healthy and environment friendly method for short trips (Shaheen, Guzman, & Zhang, 2010). Current bike share systems are automatic rental systems, which are supported by information systems (Raviv & Kolka, 2013), information systems can automatically record the data about each trip, such as trip time and docking station. By automatic rental systems, users can pick up and give back bikes quickly and easily.

Citi Bike is a privately owned bike-share program in New York, which opened to the public in May 2013 with 6,000 bikes at 330 docking stations in Manhattan and parts of Brooklyn[①], this bike-share program will be expanded to over 700 stations and 12,000 bikes by the end of 2017[②]. Citi Bike is the largest bike share program in the United States and one of the largest in the world[③], which took an average of 34,176 users daily in August 2014[④], and the number is still increasing. With the number of riders

① http://abc7ny.com/archive/9117001/

② http://a841-tfpweb.nyc.gov/dotpress/2014/10/citi-bike-program-in-new-york-city/#more-339

③ https://www.citibikejc.com/

④ https://www.citibikenyc.com/assets/pdf/august_2014_citi_bike_monthly_report.pdf

grows, there are many problems appeared, one of those most complained is: No place to park.

WNYC News[①]: (September 29, 2015)
"So many commuters jump on a Citi Bike in the morning that it's incredibly hard to find an open dock to park one in parts of Manhattan. In the afternoon, the problem reverses: too many workers are heading home, and all the available bikes disappear."

CBS New York[②]: (August 24, 2015)
"Just the mismatch between docks and bikes", "Sometimes these racks are empty; sometimes they're packed, and that forces me to fend, when I've got to ditch a bike or I need to be somewhere fast."

Abc7NY[③]: (December 12, 2014)
"And another daily biker says he has trouble finding bikes in some spots, like near the Port Authority, you have to walk up 10 blocks or so."

Citi Bike's rebalancing issue is a perennial challenge for all bike-sharing programs. Expanding the docking station and improving the number of bikes is obviously a method to solve the problem, but it is also obviously a costly and inefficient way. Rebalancing bikes, which move bikes from "busy" docking stations to "free" stations to maintain a reasonable distribution across docking stations(Fishman, Washington, & Haworth, 2014), is an effective and efficient method to handle this problem. "Rebalancing is a burgeoning sub-topic within bike share research. (Fishman, 2015)"

Because bike users' demand is changing with different time and location, and it is also affected by many factors, such as weather, holiday, weekend, traffic. Researchers base on different focus using various methods to examine bike users' demand (Fishman, 2015), such as "identify a relationship between weather and user activity", "examined the impact of topography on station activity". Generally speaking, to rebalance bikes,

① http://www.wnyc.org/story/citi-bike-deserts/

② http://newyork.cbslocal.com/2015/08/24/citi-bike-customers-seeing-red-over-broken-bikes/

③ http://abc7ny.com/traffic/new-york-city-bike-share-audit-reveals-problems/433473/

the key challenge is to integrate user preference and historical data by appropriate data mining models to predict users' demand, and test the accuracy of solutions(Fishman, 2015).

1.2 Research Project

This project examines the impact of factors such as weather on bike user activity, aims to build mathematical models to forecast bike rental demand of the Citi Bike in New York by combining historical usage patterns with the related information of users, weather, holiday and weekend.

This project focuses on the bike share program's rebalancing issue, aims to answer a question: "How many bikes will meet users' demand in a future certain time."

1.3 Research Objectives

This project will collect data from different resources such as Citi bike website, weather API, and then extract, transform and load them for this project. The predictive model built for forecasting bike rental demand is to help Citi Bike managers to better understand their users' behaviour patterns thereby better manage their systems, especially on bike rebalance issue.
The research objectives of this project are:

- To explore and study the related state-of-the-art approaches in the area of bike sharing programs, bike rebalance and data analytics technologies.
- To review and summarize similar projects to learn their evidence of model building and evaluation methods.
- To collect data from different resources and ETL the datasets.
- To explore and visualize the dataset to find out data quality issues and interesting patterns between variables.
- To build predictive models using multiple methods such as decision tree, random forest, and select one model with best performance for further enhancing.
- To enhancing the selected model by multiple techniques.
- To evaluate successful and failed evidence during model building and enhancing.
- To find out future work could be potentially improve this project.

1.4 Research Methodologies

In order to accomplish the research aim of this project will use both second research and empirical research to forecast bike rental demand.

A literature review is conducted in the field of bike sharing programs and a similar project: Kaggle Bike Sharing Demand, evidences will be summarized by case studies. Based on case studies, a review of appropriate predictive modelling algorithms and evaluation methods is conducted for data mining process.

Multiple tools such as Java, MySQL, Excel, Impala are used for data collection and ETL process. R is used for data exploration to find the quality issues and interesting patterns of the dataset. Regression, decision tree, neural network and random forest models will be built to predict the rental demand, and model enhancing techniques such as feature selection and target variable log transformation is used to improve model's accuracy. Model's accuracy is measured by RMSLE and across evaluated by ten folders.

KDD (Knowledge Discovery in Databases) is the whole process that discover knowledge from data, which provides a standard process for empirical research of this project (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). KDD process conations five core steps: data selection, pre-processing, transformation, data mining and Evaluation (Fayyad et al., 1996). Data mining is only an essential step of KDD, and the whole KDD process of this project is design below:

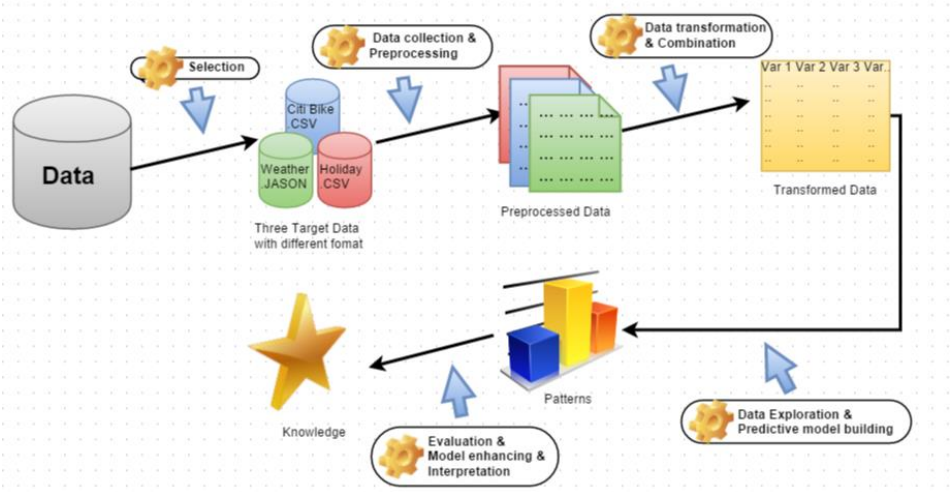


Figure 2 Project design based on KDD process

1.5 Scope and Limitations

This project is based on the whole users' rental demand of Citi Bike, and seeks to build optimize predictive model with smallest error rate, by combining historical usage patterns with the related information of users, weather, holiday and weekend. It does not include details on each customer's usage pattern, or each docking station's activity.

Besides the historical data of Citi Bike, the data collected in this project only includes weather data and holiday data, other factors (such as traffic, government policy, pollution and so on) are not included.

This project investigates similar project case studies, and selects four modelling algorithms of regression, neural network, decision tree and random forest, and all the model enhancing techniques are based on random forest model.

The predictive models of this project are built based on the data collected whilst writing this paper, there is a time limit of the use of models for three or four months, the model should be updated time by time with adding new data.

1.6 Document Outline

This dissertation is organized in the following sections:

Chapter 2 begins by an overview of bike sharing programs and a similar project: Kaggle Bike Sharing Demand, by summarizing Kaggle competitors' model building methods and evaluation methods to find out some useful experience for this project. Then this chapter provides a literature review of modelling algorithms such as decision tree, linear regression and neural network, then focus on an ensemble machine learning algorithm: random forest. Model evaluation and selection methods are also discussed in this chapter.

Chapter 3 firstly presents how to collect the dataset for this project, CSV and JASON format data from three different resources are processed by a series of tools such as Mysql and Java, Hadoop Impala is also used to process the big size data. This chapter then explores and visualizes the dataset to find out data quality issues and some

interesting patterns. The design of model building and evaluation methodology is also discussed in this chapter.

Chapter 4 compares the accuracy of 4 basic models built by four different methods, and then focus on random forest model, details the five methods to enhancing the accurate of RF models.

Chapter 5 reviews the whole project, summarizes the contributions of this project and limitation of modelling building and evaluation methods, future experiments are recommended.

2. STATE-OF-THE-ART

2.1 Introduction

This chapter will review research literature in the field of bike sharing programs and a similar project: Kaggle Bike Sharing Demand. Some case studies will be discussed with focus on the key methodologies of predictive model building and model evaluation, by summarizing these cases, the techniques of predictive model building and evaluation would potentially be implemented in an experiment in this project.

This chapter will also provide a literature review of modelling algorithms such as decision tree, linear regression and neural network, especially, focus on random forest which is an ensemble machine learning algorithm. In addition, model evaluation and selection methods will be discussed in this chapter also.

2.2 Bike Share Program

Until now, there is more than 50 years since the first bike share program was launched in Holland in 1965 (Shaheen, Zhang, Martin, & Guzman, 2011). Contemporary bike share programs refer to the provision of bicycles to enable short-term rental from one docking station to another (Fishman, Washington, & Haworth, 2013). By renting a bike, people can reach a short destination easily. Currently, there are over 500 bike-sharing programs around the world, urban transport advisor Peter Midgley notes that "bike sharing has experienced the fastest growth of any mode of transport in the history of the planet. (Larsen, 2013)" Figure below shows the growing of total number of bike share cities and systems from 1998 to 2013.

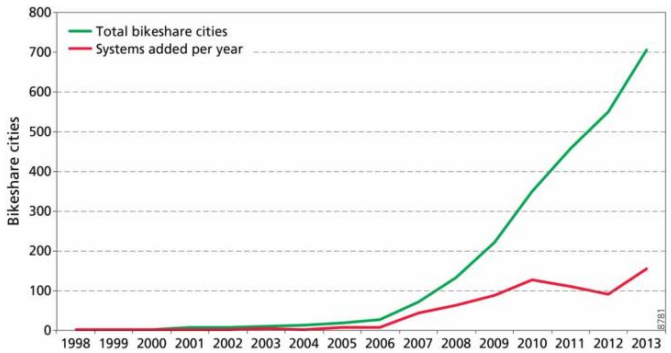


Figure 3 Growth in bike share cities (1998 - 2013) (Fishman et al., 2013)

Some researchers have categorized the evolution of bike share systems into four "generations" (Parkes, Marsden, Shaheen, & Cohen, 2013). The first generation is the White Bikes program in Amsterdam launched in 1965, which characterized by no payment or security functions (Fishman et al., 2013). Compared with the first generation, the second generation added a coin deposit "system" which was similar to trolleys at a supermarket. The first two generation's common problem is security. The third generation of bike share systems are characterized by dedicated docking stations (in which bicycles are picked up and returned), as well as automated credit card payment and other technologies to allow the tracking of the bicycles (Shaheen, Cohen, & Martin, 2013). The features of fourth-generation systems are not quite so clear, but are said to potentially include dock-less systems, easier installation, power assistance and transit smartcard integration (Parkes et al., 2013).

Current bike share systems are mostly the third generation which are supported by information systems, which collect data about the state of the system (i.e., the number of bicycles and lockers available at each station), this information is accessible online via the World Wide Web and in data kiosks at the stations (Raviv & Kolka, 2013, p. -). Based on these data, data mining technologies can help people to better understand activity patterns of bike share system and help to make further decision to manage the system. Patrick Vogel et al. (Vogel, Greiser, & Mattfeld, 2011) hypothesized that bike activity and demanding customers depend on the stations' locations, with the help of Geo BI, exploratory and cluster analysis of ride data reveal spatio-temporal dependencies of pickup and return activity patterns at bike stations, which supported their stated hypothesis. Pierre Borgnat et al. (Borgnat et al., 2011) studied the Lyon's shared bicycle program data using data mining technics such as time series, PCA analysis, clustering analysis to exhibit some features of the system and to answer some economical questions linked to such a community system such as the main flows between different parts of the city.

2.3 Similar Project: Kaggle Bike Sharing Demand

Kaggle is the world's largest community for predictive modelling and analytics competitions[①], researchers and companies can put their data and demand on this

[①] https://www.kaggle.com/solutions/competitions

platform, and data analysts from all over the world compete to solve the problems using their data mining skills. And data miners will know how well their solution is by the ranking of all the competitors. Bike Sharing Demand[①] is a competition in Kaggle which provide dataset of Capital Bikeshare in Washington D.C. and asked to combine historical usage patterns with weather data in order to forecast bike rental demand. By the end date of this competition, 3252 teams from all over the world provided their solutions.

Yin et al.[②] Du et al.[③] and Lee et al.[④] chose this competition as their finial project of Machine Learning module in Stanford University in 2014. Yin et al. tried four method to build predictive models and focused on optimize model with smallest RMSLE (Root Mean Squared Logarithmic Error). Compared with Ridge Regression, Support Vector Regression and Gradient Boosted Tree, Random Forest method is found the best "in terms of both prediction accuracy and training time", which the performance RMSLE by 10-fold cross-validation is 0.31. Before the built predictive models, they digitalized some features, for example variable Season (1=spring, 2=summer, 3=autumn, 4=winter), they spilt Season to four new variables x_{spring}, x_{spring}, x_{spring}, and x_{spring} with Boolean type. And they transformed the feature "hour" into cos(2π/24*hour) and sin(2π/24*hour)".

Du et al. tried more methods which are Basic Linear Regression, Generalized Linear Models with Elastic Net Regularization, Generalized Boosted Model, Principal Component Regression, Support Vector Regression, Random Forest and Conditional Inference Trees, and they found Random Forest and Conditional Inference Trees are the best with smallest RMSLE: 0.50 and 0.46. Compare with Yin's team, Du et al.'s solution get bigger RMSLE. They also did data preprocessing: besides converting categorical variables into binary variable, they add some more features such as peak

[①] https://www.kaggle.com/c/bike-sharing-demand

[②] http://cs229.stanford.edu/proj2014/Yu-chun%20Yin,%20Chi-Shuen%20Lee,%20Yu-Po%20Wong,%20Demand%20Prediction%20of%20Bicycle%20Sharing%20Systems.pdf

[③] http://cs229.stanford.edu/proj2014/Jimmy%20Du,%20Rolland%20He,%20Zhivko%20Zhechev,%20Forecasting%20Bike%20Rental%20Demand.pdf

[④] http://cs229.stanford.edu/proj2014/Christina%20Lee,%20David%20Wang,%20Adeline%20Wong,%20Forecasting%20Utilization%20in%20City%20Bike-Share%20Program.pdf

hour and day of the week. And they remove some features such as “temp” and “holiday”.

Lee et al.[①] tried four methods to build predictive models, which are Neural Network, Poisson Regression, Markov Model and Mean Value Benchmark. Neural Network was found out the best method with RMSLE 0.49. Compared with Yin’ group and Du’s group the error is relative bigger.

Patil et al.[②] focused on using R code to build predictive models by Random Forest method. Their RMSLE is about 0.49 which is not as good as Yin et al., but the data exploration part is good. Patil et al. using R to visualize the dataset, try to find the relationship between the count of rental bike and other variables such as weather and holiday.

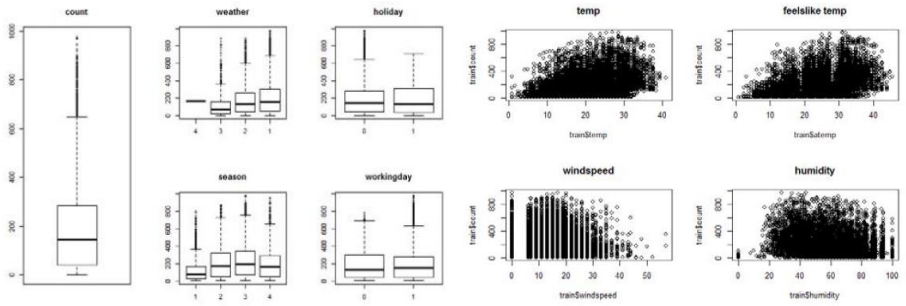


Figure 4 Visualization of Capital Bikeshare[③]

Jing et al.(2015) focused on how other features affect target variable count. The conclusion is variable “atemp” (temperature people feels) is the most important variable. The graphs clearly describe the relationship of variable count, temperature and humidity.

Sunil Ray[④] got in top 5 percentile of participants in this Kaggle competition and he shared his solutions in web[①]. He used R to build Random Forest model and got score

① http://cs229.stanford.edu/proj2014/Christina%20Lee,%20David%20Wang,%20Adeline%20Wong,%20Forecasting%20Utilization%20in%20City%20Bike-Share%20Program.pdf

② International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015. URL: http://ijiset.com/vol2/v2s4/IJISET_V2_I4_195.pdf

③ Figure from URL: http://ijiset.com/vol2/v2s4/IJISET_V2_I4_195.pdf

④ Sunil Ray: a business analytics and Intelligence professional
URL: http://www.analyticsvidhya.com/blog/author/sunil-ray/

0.38675 on Kaggle leader board. He hypothesis some variables such as “hour”, “day”, “rain” are significant affect the target variable “cnt” and using visualization method to prove his hypothesis are true. He also hypothesis traffic and pollution are also significant variables but cannot test that because no related variables in the given dataset. Based on his hypothesis and decision tree models, some new variables are added into dataset to improve the prediction power of model. By analysing Ray’s code, there are three methods to improving his model: (1) Adding some new independent variables. (2) Using random forest model to predict 0 value of variable wind speed. (3) Log transformation of the target variable. All the three methods will be tested in this project, to find out whether it also works in this task.

Wayne Liu also shared his solution to web[②], the method he used are Regression and Generalized Boosted Models. His score in Kaggle leader board is 1.38378, which is not as good as Sunil Ray. But his data exploration and visualization is as good as Patil et al.

Table 1 Summary of similar projects in Kaggle competition

Reference	Focus	Modeling Methodology	Evaluation Methodology	Conclusion	Details
Yin et al.	Optimize predictive model with smallest RMSLE	(1) Ridge Regression (2) Support Vector Regression (3) Random Forest (4) Gradient Boosted Tree	RMSLE by 10-fold cross-validation	**Random Forest** is found to perform the best which is 0.31.	Good RMSLE Data Preprocessing: (1) Digitization (2) Periodic Function
Du et al.	Optimize predictive model with smallest RMSLE	(1) Basic Linear Regression (2) Generalized Linear Models with Elastic Net Regularization (3) Generalized Boosted Models (4) Principal Component Regression (5) Support Vector Regression (6) Random Forest (7) Conditional Inference Trees	RMSLE by 10-fold cross-validation	Two of the Tree based models is found to perform the best: **CTree**: 0.46 **Random Forest**: 0.50	Relative big RMSLE compared with Yin et al. group. Data Preprocessing: (1) Digitization (2) Add some features (3) Remove some feature
Lee et al.	Optimize predictive model with smallest RMSLE	(1) Neural Network (2) Poisson Regression (3) Markov Model (4) Mean Value Benchmark	RMSLE	**Neural Network** is found to perform the best which is 0.49. (// Not building random forest model)	Relative big RMSLE compared with Yin et al. group.
Patil et al.	Using R code to build predictive models using Random Forest	(1) Random Forest (2) Enhancing RM model using TuneRF (3) Conditional Inference Trees (4) Generalized Boosted	RMSLE	**Random Forest** with TuneRF performance is 0.49	Relative big RMSLE compared with Yin. Good data exploration

[①] http://www.analyticsvidhya.com/blog/2015/06/solution-kaggle-competition-bike-sharing-demand/

[②] http://beyondvalence.blogspot.ie/2014/06/predicting-capital-bikeshare-demand-in.html

		Models			
Jing et al.	What factors affect the bike sharing system operation	(1) Multiple linear regression (2) Poisson and Topology method	P values	Variable atemp(temperature people feels) is the most important variable	Good visualization: 3D graphs
Ray	Using R code to build predictive models using Random Forest	(1) Random Forest	RMSLE	**Random forest** model with RMSLE 0.38675 on Kaggle leader board top 5%.	Good Score Three methods to improve model Using data visualization and Decision Tree to help create new variables
Wayne Liu	Using R code to build predictive models using Regression and Generalized Boosted Models	(1) Regression (2) Generalized Boosted Models	RMSLE	Score 1.38378 on Kaggle leader board (// Error rate too high)	Not as good as Ray Good data exploration

2.4 Modelling Algorithms

2.4.1 Linear Regression

Regression is one of the most wildly used models for data mining (Seber & Lee, 2012). Linear regression assumes the target variable y on features x_1, x_2, x_3 … x_k is linear. Linear regression tries to find the "best" line to fit the distribution of target variable with other variables, this line is the model used to predict the target variable (Han, Kamber, & Pei, 2011). For example of a simple linear regression, there are two variable x and y, there relationship can be described by a line $y = ax + b$. Using this line, y can be predicted by a given value of x. In this model, y is called a response variable or dependent variable, and x is called a predictor variable or independent variable (Han et al., 2011).

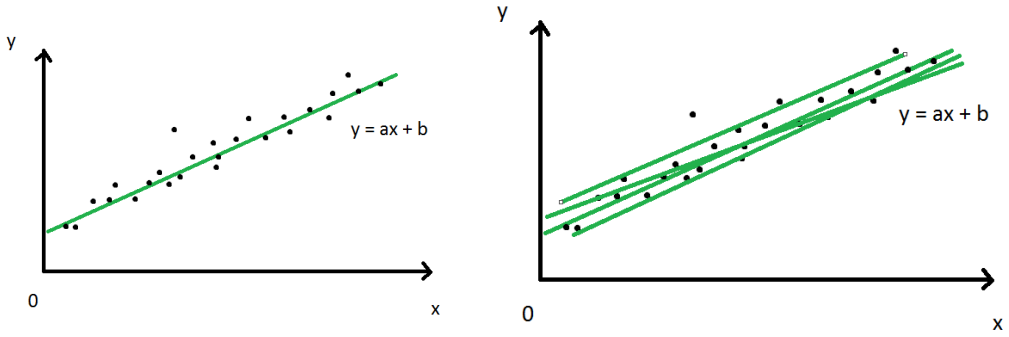


Figure 5 A simple linear regression example

But there are many lines can fit the distribution, to find out how good a line fit, residuals should be explored. For an observation $A(x_1,y_1)$, using line y = ax + b can predict response $A'(x_1,ax_1+b)$, the difference between A and A' is called a residual which is equal to $y_1 - (ax_1+b)$ (Montgomery, Peck, & Vining, 2012). The Residual Sum of Squares (RSS) is defined as:

$$RSS = (y_1 - ax_1 - b)^2 + (y_2 - ax_2 - b)^2 + \ldots + (y_n - ax_n - b)^2$$

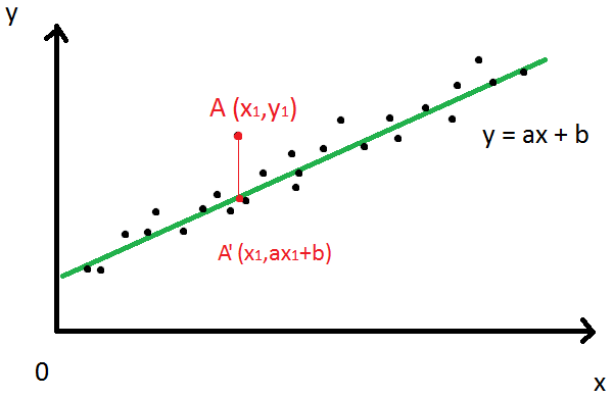


Figure 6 A residual example

Least RSS indicates most accurate model. The least squares approach chooses a and b to minimize the RSS (Montgomery et al., 2012):

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$b = \bar{y} - a\bar{x}$$

Where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. $\bar{x}$ and $\bar{y}$ are the mean of all the observations.

In multiple linear regression, there are k independent variables (Tranmer & Elliot, 2008):

$$y = a_1x_1 + a_2x_2 + a_3x_3 + \ldots + a_kx_k + b$$

Where a_1 to a_k are the coefficients relating the k independent variables to the variables of interest, b is an constant value. Simple linear recession is a special case of multiple linear regression which k equals to 1.

2.4.2 Neural Network

Neural network is a popular model which get more and more attention by researchers in recent years 2(Rojas, 2013). A neural network is "an interconnected assembly of simple processing elements, whose functionality is loosely based on the animal neuron, the processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns. (Gurney, 1997)" Processing elements are unites or nodes in network, and interconnects are the directed links which connect the processing elements (Bavarian, 1988).

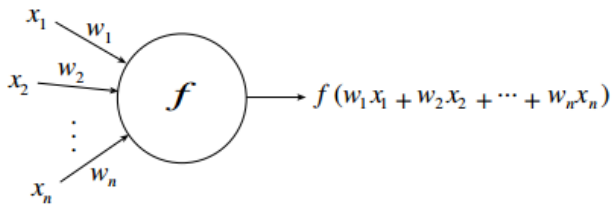


Figure 7 An abstract neuron (Rojas, 2013)

An abstract neuron has n inputs, each input channel t can transmit an incoming information x_t, w_t is corresponding weight of channel t, f is the primitive function, "the transmitted information is integrated at the neuron and the primitive function is then evaluated (Rojas, 2013)".

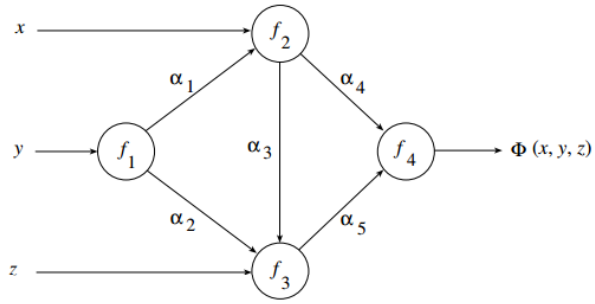


Figure 8 Functional model of an artificial neural network (Rojas, 2013)

Neural network combines n primitive functions. In this example there are 4 primitive functions: f_1, f_2, f_3, f_4, the network function Φ is evaluated at the point (x, y, z) and

produced by combining functions of f_1, f_2, f_3 and f_4 (Rojas, 2013), “different selections of the weights α_1, α_2, . . ., α_5 make different network functions (Rojas, 2013).”

A multilayer feed-forward neural network contains three kinds layers: input layer, hidden layer and output layer (Han et al., 2011), all the layers are made by neurons. The inputs to the network correspond to the attributes measured for each training tuple, pass through the input layer, then they are weighted and fed simultaneously to hidden layers, the output of last hidden layer is the weighted input of output layer which emits the network's prediction for given tuples (Han et al., 2011). “Given enough hidden units and enough training samples, multilayer feed-forward neural networks can closely approximate any function. (Han et al., 2011)”

2.4.3 Decision Tree

Decision Tree is a popular classification algorithm which is widely used in many areas (Abdelhalim & Traore, 2009) such as business (Sohn & Moon, 2004) (Duncan, 1980) and medical (Gambhir, Hoh, Phelps, Madar, & Maddahi, 1996) (Fan, Chang, Lin, & Hsieh, 2011). Because it is simulate human’s decision making process(Han et al., 2011), it is easy to understand. For example, this decision tree helps a student to make a decision to rent a bike or drive a car to go to university in a morning: If it is raining he will drive a car. If not raining he will look at what time he will leave home, if it is early enough he will ride a bike, if it is after 8:00 he will drive a car.

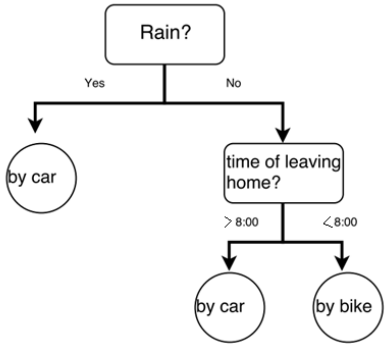


Figure 9 Example of decision tree

In this sample, there are two base hypothesis $g_t(x)$ "by car" and "by bike" which are the leaf at the end of path. "weather" and "time" are the condition $b_t(x)$ which is the branching criteria. A decision tree algorithm[①] can be described as below:

Table 2 Algorithm of decision tree

Algorithm : Basic Decision Tree	
01: **function DecisionTree(data D)**	// recursive function
02: if termination criteria met	
03: return base hypothesis $g_t(x)$	// $g_t(x)$: leaf at end of path t
04: else	
05: learn branching criteria b(x)	// b(x): branching criteria
06: split D to C parts D_c by b(x)=c	
07: build sub-tree G_c ←DecisionTree(D_c)	// $G_c(x)$: sub-tree hypothesis at the c-th branch
08: return G(x) = $\sum_{c=1}^{C}[b(x)=c]G_c(x)$	// G(x): full-tree hypothesis
09: **end function**	

But how to spit dataset D into smaller branches D_c best? For example, there is a small training dataset collected by a retail company, the dataset records some information of their customers and whether they have responded to promotions the company has run. This company want to know what kind of new customers will respond to promotional mailings[②].

[①] Algorithm from Coursera: Machine Learning Technology by professor Hsuan-Tien Lin in National Taiwan University. URL: https://class.coursera.org/ntumltwo-001/lecture/77

[②] Example from DIT exam paper of machine learning in 2012-2013, URL: http://www.dit.ie/library/a-z/exampapers/

ID	Income	Age	Newspaper	Health Foods	Respond
C-01	<40	81	no	low	No
C-02	<40	76	no	high	No
C-03	40-60	86	no	low	Yes
C-04	>60	84	no	low	Yes
C-05	>60	45	yes	low	Yes
C-06	>60	66	yes	high	No
C-07	40-60	41	yes	high	Yes
C-08	<40	68	no	low	No
C-09	<40	32	yes	high	Yes
C-10	>60	56	yes	low	Yes
C-11	<40	58	yes	high	Yes
C-12	40-60	52	no	high	Yes
C-13	40-60	90	yes	low	Yes
C-14	>60	69	no	high	No

Figure 10 Example of a training dataset[①]

To answer this question, there are two key tasks should to be considered: The first one is to determine which attribute is most useful for discriminating: income, age, newspaper or health foods. The second one is to determine how to split the dataset to branches. Information Gain describes how important a given feature is in training dataset (Quinlan, 1986), and higher Information Gain is, the more information the feature contains, which means the feature is more important (Han et al., 2011).

Information Gain = Entropy (parent) – Average Entropy (children)

Entropy is a method to measure the impurity of each partition after split (Han et al., 2011).

$$\text{Entropy} = -\sum_{i=1}^{n} p_i log_2(p_i)$$

Where p_i is the probability of class i. Take the retail company dataset for example, the importance of feature Newspaper can be measured as below:

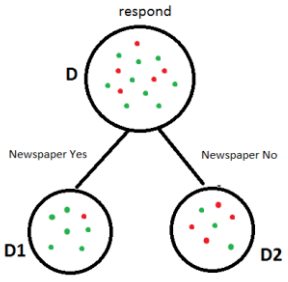
Entropy (parent)

$= -\frac{5}{14} log_2 \frac{5}{14} - \frac{9}{14} log_2 \frac{9}{14} = 0.94$ (bits)

Average Entropy (children)

$= p_1 \times$Entropy (children D_1) + $p_2 \times$Entropy (children D_2)

$= \frac{7}{14}(-\frac{1}{7} log_2 \frac{1}{7} - \frac{6}{7} log_2 \frac{6}{7}) + \frac{7}{14}(-\frac{4}{7} log_2 \frac{4}{7} - \frac{3}{7} log_2 \frac{3}{7})$



[①] Figure from DIT exam paper of machine learning in 2012-2013, URL: http://www.dit.ie/library/a-z/exampapers/

= 0.79 (bits)

Information Gain of Newspaper

= Entropy (parent) – Average Entropy (children) = 0.15 (bits)

Figure 11 Classification by Newspaper

So the Information Gain of the feature Newspaper at the root node of the tree is 0.15 bits. It is also calculated that Information Gain of Income is 0.247 bits, which means feature Income contains more information than feature Newspaper. By this way, each feature's importance can be measured.

For an feature that is continuous-valued such as Age, it should be find out the "best" split-point where the split-point is a threshold (Han et al., 2011). In this case, when sorted Age in advance, there are two threshold which are between age 58, 66 and age 81, 84. So this dataset can by split to two groups by age: $\{62 \leq \text{age} \leq 82\}$ and {age <62 or age > 82}. Beside Information Gain, Gain Ratio and Gini Index are also popular methods used in feature selection process when building a decision tree model (Han et al., 2011).

Table 3 Sample data sorted by age

Age	Respond	Threshold
32	Yes	.
41	Yes	.
45	Yes	.
52	Yes	.
56	Yes	.
58	Yes	.
66	No	√
68	No	.
69	No	.
76	No	.
81	No	.
84	Yes	√
86	Yes	.
90	Yes	.

"The Information Gain measures prefers to select attributes having a large number of values (Karegowda, Manjunath, & Jayaram, 2010)", which means it tend to make more classification. For example, using feature ID to split the training dataset, it will tend to split a very big splits by each ID value, the Information Gain is really good, but it is obvious useless. Gain Ratio attempts to overcome this bias. The gain ratio is defined as (Karegowda et al., 2010):

$$\text{Gain Ratio} = \text{Information Gain} / \text{SplitInfo(D)}$$

$$\text{SplitInfo(D)} = -\sum_{i=1}^{n}(|s_i| \,/\, |s|) log_2((|s_i| \,/\, |s|)$$

"The attribute with the maximum gain ratio is selected as the splitting attribute. (Han et al., 2011)" The Gini Index is used in C&RT which is introduced in chapter 2.4.2 modelling algorithm of Random Forest.

Fully grown tree (unpruned decision tree) may "reflect anomalies in the training data due to noise or outliers (Han et al., 2011)", Tree pruning can "cut" the least-reliable branches, not only make the tree smaller, faster, but also more accurate. Tree pruning is a method to handle over-fitting problem. In summary, decision tree is a popular

modelling algorithm which is easy to assimilate by people. It is relative effective and accurate, but "successful use may depend on the data at hand (Han et al., 2011)".

2.4.4 Random Forest

Random forest is a well-known machine learning technique which combine both "bagging" and un-pruned C&RT (Aung & Hla, 2009).

- Bootstrap & Bagging

Bootstrap is a statistical method for accessing accuracy (Efron & Tibshirani, 1994). A bootstrap sample is created by sampling a given dataset uniformly with replacement (Kohavi & others, 1995). Because bootstrap is sample with replacement, each tuple in bootstrap sample has same possibility with other tuples to be sampled again (Han et al., 2011).

Bagging is also called Bootstrap aggregating which is a machine learning method to increasing stability and accuracy by generating multiple versions of a predictor in classification and regression (Yao, Zhao, Liu, & Cai, 2014). Bagging use bootstrap method to re-sample the dataset D to "simulated" sub-dataset D_t time by time, after k times bootstrapping, there will be k sub-training sets. Because Bagging is a repeatedly replacement sampling process, some of the original tuples of *D* may be sampled several times, while some tuples may not be included in any sample D_t. The probability of any tuple which is not chosen after k bootstrapping samples is $(1-\frac{1}{k})^k \approx e^{-1} \approx 0.368$ (Kohavi & others, 1995).

- C&RT

A basic C&RT (Classification and Regression Tree) is a fully grown (unpruned) binary tree which only split to two branch at each node(Chen, Hsu, Chiu, & Rau, 2011).

- Gini index

In classification, Gini index (Breiman, Friedman, Stone, & Olshen, 1984) is used to choose the important attribute to split each node of the tree best, the algorithm(Aung & Hla, 2009) of Gini index is described as below:

$$Gini(attr) = 1 - \sum [B_i]^2$$

$$Gini_{split} = \sum_{attr=1}^{m_{try}} \frac{n_{attr}}{N} Gini(attr)$$

In which B_i is the relative frequency of the attribute ($attr$) at class i, m_{try} is number of attributes, $n_{attr}$is the number of randomly selected training records, N is the total number of training records of dataset D. Gini index measures the impurity of each split, so best split is based on the minimum of $Gini_{split}$. In regression, minimum regression error is for best splitting(Svetnik et al., 2003).

- Algorithm of random forest

Random forest using bagging to draw T_{trees} samples with replacement from original dataset D, for each sample D_t, grow an unpruned (maximum depth) classification or regression tree using best split method. "This method is not choosing the best split among all predictors, but randomly sample m_{try} of predictors and choose the best split from among those variables. (Liaw & Wiener, 2002)" Each m_{try} is measured by Gini index (in classification) or regression error (in regression), smallest Gini index (or regression error) indicates the best split method. The table below describes the algorithm of random forest (Joelsson, Benediktsson, & Sveinsson, 2005).

Table 4 Algorithm of random forest

Algorithm[①] : Basic Random Forest	
01: **function RandomForest(D)**	
02: for t = 1,2,3,…,T	
03: request size N' data D_t by bootstrapping with D	// Select a N' size bootstrap sample D_t from dataset D
04: obtain tree g_t by **DTree(D_t)**	// Grow an un-pruned tree on this bootstrap
05: return G = Uniform({g_t})	
06: **end function**	
07:	
08: **function DTree(D)**	
09: if termination return base g_t	
10: else learn b(x) and split D to D_c by b(x)	// using recursive to build sub-tree
11: bulid D_c ←DTree(D_c)	// b(x) means at each node randomly

[①] Algorithm from Coursera: Machine Learning Technology by professor Hsuan-Tien Lin in National Taiwan University. URL: https://class.coursera.org/ntumltwo-001/lecture/83

12:	return G(x) = $\sum_{c=1}^{C}[b(x)=c]G_c(x)$	sample m_{try} and learn to find out the best split method
13: **end function**		

- OOB

Because bagging process is randomly sample with replacement, it is possible that some of the original tuples of *D* may not be include in any sample D_t, these tuples are called out-of-bagging (OOB) (Breiman, 2001). The possibility of OOB (after N times' drawings of N tuples) is $\frac{1}{e}N$, which means about 36.8% original dataset would be OOB (Liaw & Wiener, 2002).

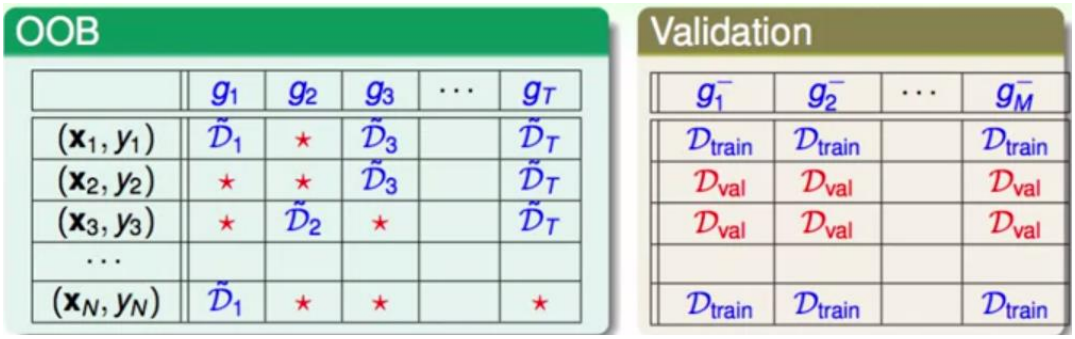


Figure 12 Description of OOB[①]

Because OOB is not selected to training to obtain sub-tree g_t (which is fully grow tree by sample data D_t) it can be used to validation g_t. Take (x_n,y_n) tuple for example, it can be used as g_2, g_3, …, g_T's validation, but not for g_1. OOB can be used to describe validation dataset D^- (besides D_1 of all dataset D): $D_N^-(x) = average\left(g_{2,}g_{3,\dots,}g_{T,}\right)$. OOB error of all the dataset D is defined as below:

$$E_{oob}(D) = \frac{1}{N}\sum_{n=1}^{N} err\left(y_n, D_n^-(x_n)\right)$$ [②]

OOB error can be used for random forest self-validation, and no need to cut the dataset to training part and validation part. "Our experience has been that the OOB estimate of error rate is quite accurate, given that enough trees have been grown. (Bylander, 2002) (Liaw & Wiener, 2002)"

- Importance of feature

[①] Figure from Coursera: Machine Learning Technology by professor Hsuan-Tien Lin in National Taiwan University. URL: https://class.coursera.org/ntumltwo-001/lecture/85

[②] Figure from Coursera: Machine Learning Technology by professor Hsuan-Tien Lin in National Taiwan University. URL: https://class.coursera.org/ntumltwo-001/lecture/85

In some dataset contains redundant features, which means these features provide same information or related information, such as date of birth and age, woman and female. Some features are irrelevant features which provide useless information, such as weather related variables for tax fraud prediction. So feature selection is useful and important. To test how important a feature is, random forest using permutation test to select features.

$$\text{importance}(f_i) = \text{performance}(D) - \text{performance}(D_p)$$
$$= E_{oob}(D) - E_{oob}^{(p)}(D)^{①}$$

Where $E_{oob}^{(p)}(D)$ comes from replacing each request of $x_{n,i}$ by a permuted OOB value.

Random forest is an effective predictive method and at the same time, it is not as easy as decision tree to over-fitting because of the Law of Large Numbers (Breiman, 2001). The case studies of similar project in Kaggle show that four of seven cases' conclusions are random forest model perform relative best (see chapter 2.3).

2.5 Model Evaluation and Selection

2.5.1 Data Split

In data mining, source dataset is split to two or three sub-set for different use. The training set is used to build the predictive models, the testing set is unseen data when building models, which is used to evaluate the models performance[②]. In some cases e.g. (Schüldt, Laptev, & Caputo, 2004) source dataset is split to three dataset, training set, testing set and validation set. Validation set is used to review the model (which is built by training set) and select the best performance, for example, to minimize over-fitting. Test set is to estimate the accuracy of the selected approach, to test the final solution in order to confirm the actual predictive power of the network.

2.5.2 K-fold Cross Validation

K-fold cross validation is a wildly used machine learning validation method for accuracy assessment (Alippi & Roveri, 2010). In k-fold cross validation, dataset D will

① Algorithm from Coursera: Machine Learning Technology by professor Hsuan-Tien Lin in National Taiwan University. URL: https://class.coursera.org/ntumltwo-001/lecture/87

② https://webdocs.cs.ualberta.ca/~zaiane/courses/cmput690/notes/Chapter1/

be split into k equal size "folds" D_t randomly (Kohavi & others, 1995), and repeated k times (Alippi & Roveri, 2010, p. -). For example, in 10-fold cross validation, 1500 records dataset will be split to ten 150 records folds randomly, and this process will be repeated for 10 times. At each time, D_t will be split to two groups: one group D_j contains k-1 folds and one group D_i contains the rest fold (Rodriguez, Perez, & Lozano, 2010). D_j is used as training group and D_j is used to validate the accuracy of models, each time will get an error by train D_j, the total process will get k estimate errors. Finally, the k-fold cross validation error is the average of the k errors (Alippi & Roveri, 2010).

2.5.3 RMSLE

The Root Mean Squared Logarithmic Error (RMSLE) is to compare the predictive value with the true value (cnt value in this project), and is "calculated as the square root of the squared bias plus squared standard error" (Mickey & Greenland, 1989):

$$\text{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(a_i + 1) - \log(b_i + 1))^2}$$

Where n is the total number of observations in the testing dataset, a_i is predicted rental bike number, and b_i is the actual number. RMSLE is a method to measure the error rate, so smaller RMSLE value indicates more accurate model.

This project chooses this method because it is used by Kaggle competition[①②], all the solutions provided by data miners from all over the world is evaluated and ranked by RMSLE score, so it allows this project's results to compare with them.

2.6 Conclusion

This chapter has summarized 7 case studies of similar project in Kaggle, found out the neural network, decision tree and random forest which are the best performance models in their studies. In addition, all of the predictive cases use RMSLE (Root Mean Squared Logarithmic Error) as their evaluation method.

① https://www.kaggle.com/c/bike-sharing-demand

② http://cs229.stanford.edu/proj2014/Yu-chun%20Yin,%20Chi-Shuen%20Lee,%20Yu-Po%20Wong,%20Demand%20Prediction%20of%20Bicycle%20Sharing%20Systems.pdf

Various models have been discussed including decision tree, linear regression, neural network and random forest. Random forest, which is an ensemble machine learning algorithms both combined decision tree and bagging, is found an effective method and it have more advantages compared with the other three.

3. DESIGN / METHODOLOGY

3.1 Introduction

This chapter will present the details of how to collect and process the dataset. Three resources will provide different format of Citi bike data, weather data and holiday data respectively, a series of tools such as Mysql and Java will be used to extract, transform and loading the data. Especially, Hadoop Impala is also used to process the big size data.

Then this chapter will summarize and fix data quality issues of this dataset. Relationship of variables and some interesting patterns will be explored and visualized using R Studio. The design of model building and evaluation methodology will be also discussed in this chapter.

3.2 Data Preparation

The aim of this project is to predict bike sharing demand by combining historical usage patterns with weather data, users' information, and holiday data. In another word, this project tries to answer a question: How many bikes will be rented at a particular time based on specific condition such as weather, user patterns, and holiday? Before answer this question. It is important to consider what factors will affect people's behaviours to rent a bike. For example: How many bikes were rented in a certain time? Who rented the bikes? What was the weather like? What was the temperature? Was that a working day? Was that a holiday? Based on these factors, the dataset of this project need to contain four main information: (1) Time and rented bike number; (2) User information; (3) Weather information; (4) Holiday information.

There are three data source for this project: (1) Citi Bike website[①] provides New York City's bike sharing system data which contains rented bike information and user information from July 2013 to September 2015. (2) Weather Underground Website[②]

[①] https://www.citibikenyc.com/system-data

[②] http://www.wunderground.com

provides New York historical weather information such as humid, temperature, rain, snow. (3) "OfficeHolidays" website[①] provides holiday information of New York. And "Timeanddate" website[②] provides weekend information of USA.

Because the Citi Bike program began to operate in mid-2013, and the dataset's duration is from July 2013 to September 2015, this project will use the datasets of 2014 and 2015 to build the predictive models. Figure 3.1 shows the overview of three datasets preparation process, the detail of each dataset extracting and processing will be discussed one by one.

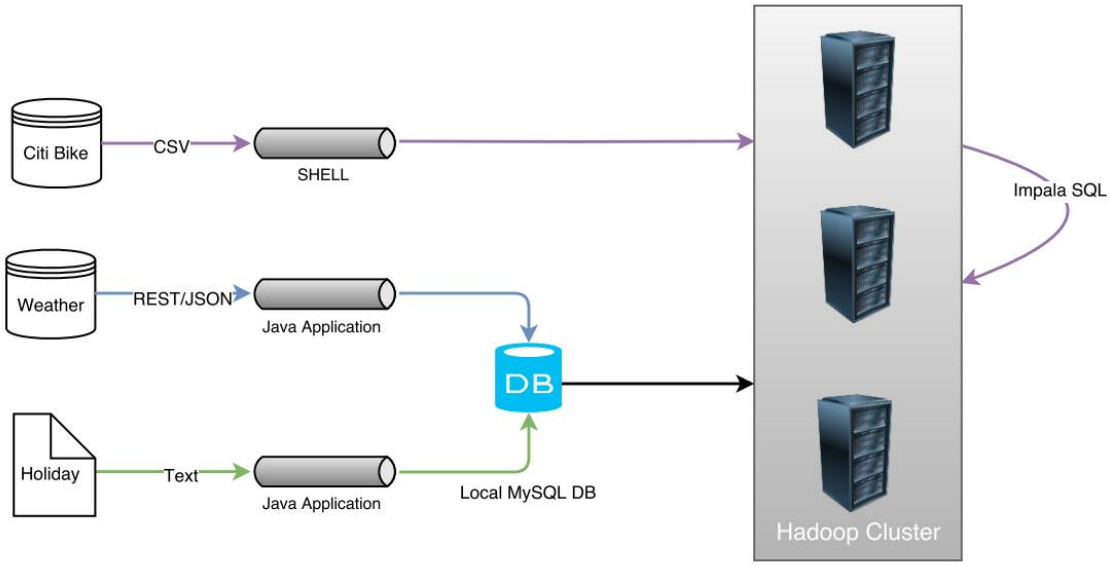


Figure 13 Overview of data preparation process

3.2.1 New York Citi Bike Dataset

The Citi Bike website[③] provides CSV (comma-separated values) files of Citi Bike trip histories. The source dataset contains 15 variables including 4 key variables: "starttime", "usertype", "birth year" and "gender" which are related to the first two main information of this project: : (1) Time and rented bike number; (2) User information. Table 3.1 shows the details of this dataset:

[①] http://www.officeholidays.com/countries/usa/regional.php?list_year=2014&list_region=New%20York

[②] http://www.timeanddate.com/calendar/?country=1&year=2014

[③] https://www.citibikenyc.com/system-data

Table 5 Description of Citi Bike dataset

Field Name	Data Type	Description	Information
tripduration	Numeric	Trip duration (by seconds)	.
starttime	Date	Start time and date eg. 1/1/2014 01:06:20	(1)
stoptime	Date	Stop time and date	.
start station id	Numeric	Start station ID	.
start station name	Character	Start station name	.
start station latitude	Numeric	Start station latitude	.
start station longitude	Numeric	Start station longitude	.
end station id	Numeric	End station ID	.
end station name	Character	End station name	.
end station latitude	Numeric	End station latitude	.
end station longitude	Numeric	End station longitude	.
bikeid	Numeric	Bike ID	.
usertype	Character	User type (Customer = 24-hour pass or 7-day pass user; Subscriber = Annual Member)	(2)
birth year	Date	User's birth year	(2)
gender	Numeric	User's gender (0 = unknown; 1 = male; 2 = female)	(2)

The target Citi bike table is designed to provide information of total rented bike number per hour, at the main time, provide users' information. Based on the Citi Bike source dataset, the target table can be designed as table 3.2. The "starttime" can be split to "year", "month", "day" and "hour", the total number of rented bike (field "count") can be calculated by grouping "starttime" by hour.

Table 6 Description of target Citi Bike table

Source Name	Field Name	Data Type	Description
starttime	start_date	Date	Date (D/M/YYYY)
	cnt	Numeric	Number of total rented bike
	year	Numeric	Year
	month	Numeric	Month
	day	Numeric	Day
	hour	Numeric	Hour
usertype	cnt_subscriber	Numeric	Number of annual member
	cnt_customer	Numeric	Number of short term customer
birth year	cnt_birth_1930s	Numeric	Number of users born in 1930s
	cnt_birth_1940s	Numeric	Number of users born in 1940s
	cnt_birth_1950s	Numeric	Number of users born in 1950s
	cnt_birth_1960s	Numeric	Number of users born in 1960s
	cnt_birth_1970s	Numeric	Number of users born in 1970s
	cnt_birth_1980s	Numeric	Number of users born in 1980s
	cnt_birth_1990s	Numeric	Number of users born in 1990s
	cnt_birth_2000s	Numeric	Number of users born in 2000s
	cnt_birth_outlier	Numeric	Number of null value and outlier value
gender	cnt_unknown	Numeric	Number of gender unknown users
	cnt_male	Numeric	Number of male users
	cnt_female	Numeric	Number of female users

To generate the target table, there are two main steps: data pre-process and SQL process.

(1) Data pre-process

The source dataset contains header and double quote, the first task is to remove header and double quote. Because the datasets are downloaded by month which means there are 12 files in each year, it is slow to open each file which is more than 100MB by Excel, so the preliminary process will be done by SHELL in Linux.

(2) SQL process

The observation of source dataset is recorded by the time of each rented bike, the total datasets is relative big (more than 1.4G), so the main process will be done by Impala which is a SQL like database supported by Cloudera in Hadoop. To generate the target table, three tables are created by Impala: The first one is Citi Bike raw table which is the same format as source dataset. The second table is an intermediate table which split start_time to start_date and start_hour, and at the same time select three key variables related to users' information. The third table is the target table, this table is created by count the occurrence of each identified variables from step two by hour. Based on the fact that people born before 1930s are unlikely able to rent a bike, therefore those values are counted as cnt_birth_outlier variable. The table details and key SQL code are described as figure 3.2.

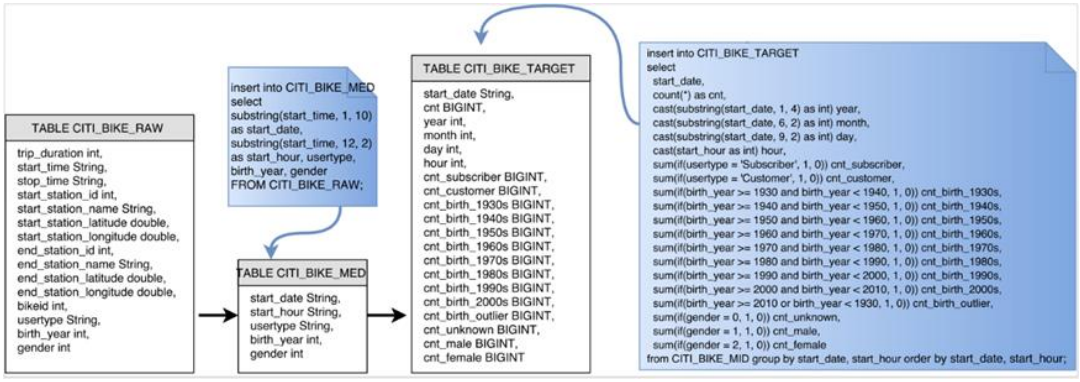


Figure 14 Details of Impala process

3.2.2 New York Weather Dataset

New York historical weather information can be collected by sending Http request to an API (Application Programming Interface) from Weather Underground Website[①]. The dataset is in JSON format which contains date information and weather

[①] http://api.wunderground.com/api/f0bc8c8eb066140c/history_20140101/q/US/Newyork.json

information, and each observation is recorded by hour. Figure 3.3 shows a sample data in Jan 1st, 2014. The sample data contains 24 observations on hourly basis, and each observation provides 30 weather related variables.


```
{
    "response": {
        "version": "0.1",
        "termsofService": "http://www.wunderground.com/weather/api/d/terms.html",
        "features": {
            "history": 1
        }
    },
    "history": {
        "date": {
            "pretty": "January 1, 2014",
            "year": "2014",
            "mon": "01",
            "mday": "01",
            "hour": "12",
            "min": "00",
            "tzname": "America/New_York"
        },
        "utcdate": {
            "pretty": "January 1, 2014",
            "year": "2014",
            "mon": "01",
            "mday": "01",
            "hour": "17",
            "min": "00",
            "tzname": "UTC"
        },
        "observations": [
            {
                "date": {
                    "pretty": "12:51 AM EST on January 01, 2014",
                    "year": "2014",
                    "mon": "01",
                    "mday": "01",
                    "hour": "00",
                    "min": "51",
                    "tzname": "America/New_York"
                },
                "utcdate": {
                    "pretty": "5:51 AM GMT on January 01, 2014",
                    "year": "2014",
                    "mon": "01",
                    "mday": "01",
                    "hour": "05",
                    "min": "51",
                    "tzname": "UTC"
                },
                "tempm": "-3.9",
                "tempi": "25.0",
                "dewptm": "-14.4",
                "dewpti": "6.1",
                "hum": "45",
                "wspdm": "5.6",
                "wspdi": "3.5",
                "wgustm": "-9999.0",
                "wgusti": "-9999.0",
                "wdird": "260",
                "wdire": "West",
                "vism": "16.1",
                "visi": "10.0",
                "pressurem": "1025.4",
                "pressurei": "30.28",
                "windchillm": "-6.3",
                "windchilli": "20.7",
                "heatindexm": "-9999",
                "heatindexi": "-9999",
                "precipm": "-9999.00",
                "precipi": "-9999.00",
                "conds": "Clear",
                "icon": "clear",
                "fog": "0",
                "rain": "0",
                "snow": "0",
                "hail": "0",
                "thunder": "0",
                "tornado": "0",
                "metar": "METAR KNYC 010551Z AUTO 26003KT 10SM CLR M04/M14 A3030 RMK A02 SLP254 T10391144 10000 21039 51014"
            }
```


Figure 15 Sample of weather dataset

7 key variables are selected, based on they affect people's behaviour of renting a bike relatively more distinctly. They are temperature, humidity, wind speed, rain, snow, fog, and hail. Based on the weather dataset, the target weather table can be designed as below:

Table 7 Design of target weather table

Name	Type	Label
year	Numeric	Year
month	Numeric	Month
day	Numeric	Day
hour	Numeric	Hour
temperature	Numeric	Temperature (in Celsius)
humidity	Numeric	Relative humidity (%)
windspeed	Numeric	Wind speed (in Kph)
weatherType	Numeric	Snow = 1, Rain = 2, fog = 3, hail = 4, else = 0

To generate the target table, the JASON format weather data need to be download first, then extract useful information and upload to database. There are three Java class are created: The Weather class provides a data model. The DatabaseUtil class provides a function to persist data to database. The WeatherDataCollector class provides a function to download weather data by month, get each observation by hour, and store the extracted data into data model, and save it to database.

Table 8 Algorithm of retrieve Weather JSON data and persist to database

Algorithm 1: Retrieve Weather JSON Data And Persist To Database

```
01: class Weather
02:     all attributes                                  // including all fields in table 3.4
03:     constructor method                              // setting values
04:     function getYear()                              // getting year value
05:     …                                               // getting all other attributes values
06:     function isFog                                  // getting fog value
07: end class
08:
09: class DatabaseUtil
10:     load the JDBC driver
11:     function saveWeather(weather)
12:         conn ← connect to database                  //MySQL database
13:         sql ← map weather data into database
14:         connection.excute(sql)
15:     end function
16: end class
17:
18: class WeatherDataCollector
19:     function collect(year, month)
20:         create a calendar object
21:         get the number of the days of the month
22:         for each day of the month
23:             response = get weather data for the day
24:             observationList =  get the observations from response
25:                 for each hour of the day
26:                     prepare weather data
27:                     databaseUtil.saveWeather(weather)
28:                 end for                             //save weather data to database
29:         end for
30:     end function
31: end class
```

3.2.3 New York Holiday Dataset

Website OfficeHolidays[①] provides holiday information of New York. Take year 2014 for example there are 10 national holidays in total. A variable of weekend will also be included in the target holiday table, based on a hypothesis that bike activity and demanding customers are in different patterns in work day and weekend.

[①]http://www.officeholidays.com/countries/usa/regional.php?list_year=2014&list_region=New%20York

Day	Date	Holiday	Comments
Wednesday	January 01	New Years Day	
Monday	January 20	Martin Luther King Day	Third Monday in January
Sunday	February 02	Groundhog Day	Not a National Holiday
Wednesday	February 12	Lincoln's Birthday	Connecticut, Illinois, Missouri, New Jersey, New York.
Monday	February 17	Presidents Day	3rd Monday in February
Sunday	May 11	Mothers Day	2nd Sunday in May. Not a public holiday.
Monday	May 26	Memorial Day	Last Monday in May
Sunday	June 15	Fathers Day	Third Sunday of June. Not a National Holiday
Friday	July 04	Independence Day	
Monday	September 01	Labor Day	First Monday in September
Monday	October 13	Columbus Day	Second Monday in October.
Wednesday	November 05	Election Day	New York only.
Tuesday	November 11	Veterans Day	
Thursday	November 27	Thanksgiving	Fourth Thursday in November
Friday	November 28	Day after Thanksgiving	Fourth Friday in November
Thursday	December 25	Christmas Day	
Friday	December 26	Day after Christmas	Federal Employees only. State Holiday in six states.

National Holiday
Not a Public Holiday
Government/Public Sector

Figure 16 New York holiday in 2014

Based on the requirement, the target holiday table is designed as below:

Table 9 Design of target holiday table

Name	Type	Label
year	Numeric	Year
month	Numeric	Month
day	Numeric	Day
holiday	Numeric	Public holiday (1 = holiday, 0 = not a holiday)
weekday	Numeric	Mon =1, …, Sun = 7

To generate the target holiday table, a calendar date table which contains holiday and weekend information need be created first, then persist the data to database. The DatabaseUtil class provides a function to persist data to database. The CalendarCollector class provide a function to identify each holiday and weekend of year 2014 and 2015, then save each calendar date to database.

Table 10 Algorithm of create calendar data and persist to database

Algorithm 2: Create Calendar Date And Persist To Database

```
01: class DatabaseUtil
02:     load the JDBC driver
03:     function saveCalendarDate(calendarDate)
04:         conn ← connect to database                    //MySQL database
05:         sql ← map calendar date into database
06:         connection.excute(sql)
07:     end function
08: end class
09:
10: class CalendarCollector
11:     function collect
12:         create 2014holiday ArrayList
13:         create 2015holiday ArrayList
14:         for each day of 2014.1.1 and 2016.1.1
15:             set year, month, day value to bikeDate
```

```
16:             if  the date in either ArrayList
17:                 set holiday = ture
18:             else if the date is weekend
19:                 set weekend = ture
20:             else
21:                 set weekend = false
22:              databaseUtil.saveCalendarDate(bikeDate)      //save calendar date to
23:         end for                                           database
24:     end function
25: end class
```

3.2.4 Datasets Combination

Citi Bike target table and weather target table contains four common variables: year, month, day and hour. Based on the four variables, left join can be used to join this two tables. All the three target tables contain three common variables except hour, so the third target table (holiday) can be joined by the same way. The final table can be produced by using SQL left join in Impala. The details of the final table are described as below:

Table 11 Details of the final table

Name	Type	Label	Source
cnt	Numeric	Number of total rented bike	1
year	Numeric	Year	1
month	Numeric	Month	1
day	Numeric	Day	1
hour	Numeric	Hour	1
cnt_subscriber	Numeric	Number of total annual member	1
cnt_customer	Numeric	Number of total short term customer	1
cnt_birth_1930s	Numeric	Number of total users born in 1930s	1
cnt_birth_1940s	Numeric	Number of total users born in 1940s	1
cnt_birth_1950s	Numeric	Number of total users born in 1950s	1
cnt_birth_1960s	Numeric	Number of total users born in 1960s	1
cnt_birth_1970s	Numeric	Number of total users born in 1970s	1
cnt_birth_1980s	Numeric	Number of total users born in 1980s	1
cnt_birth_1990s	Numeric	Number of total users born in 1990s	1
cnt_birth_2000s	Numeric	Number of total users born in 2000s	1
cnt_birth_outlier	Numeric	Number of total null value and outlier value	1
cnt_unknown	Numeric	Number of gender unknown users	1
cnt_male	Numeric	Number of male users	1
cnt_female	Numeric	Number of female users	
temperature	Numeric	Temperature (in Celsius)	2
humidity	Numeric	Relative humidity (%)	2
windspeed	Numeric	Wind speed (in Kph)	2
weatherType	Numeric	Snow=1, Rain=2, fog=3, hail=4, else=0	2
holiday	Numeric	Public holiday (1 = holiday, 0 = not a holiday)	3
weekday	Numeric	Mon =1, …, Sun = 7	3

(1: Citi Bike dataset 2: Weather dataset 3: Holiday dataset)

3.2.5 Tools Used

In the data preparation process, there are five main tools are used to produce the final table. (1) MySQL database is used create table and store data locally before uploading to Impala. For example, before generating the target holiday table, an empty Holiday table needs to be created in MySQL database, and a Java program inserts holiday information into this table. (2) Hadoop cluster is the main repository of all the datasets. It provides powerful data processing environment for the project. The Hadoop cluster built for this project consists of three virtual machines. (3) Cloudera Impala is a SQL like analytic database sitting on Hadoop cluster environment. The Citi Bike dataset is mainly processed on this platform. To aggregate the hourly data of 1.4 GB it only takes less than ten seconds to process. (4) Java is used to create an application that requests weather data from remote API and filling the target weather table. And Jackson Library is used to processing the JSON data in the response. (5) Linux SHELL is used to preliminarily process the CSV data before uploading to Hadoop. For example, it removes the header and special double quotes in the source datasets.

3.3 Data Exploration / Visualization

After all the data collection and preparation process, the dataset of this project contains 14641 records and 25 variables, which are all numeric variables. The target variable is cnt which record the total number of rental bike of each hour from Jan 1, 2014 to June 30, 2015.

Table 12 Description of dataset of this project

Name	Type	Label
year	Numeric	Year
month	Numeric	Month
day	Numeric	Day
hour	Numeric	Hour
cnt	Numeric	Number of total rented bike
cnt_subscriber	Numeric	Number of total annual member
cnt_customer	Numeric	Number of total short term customer
cnt_birth_1930s	Numeric	Number of total users born in 1930s
cnt_birth_1940s	Numeric	Number of total users born in 1940s
cnt_birth_1950s	Numeric	Number of total users born in 1950s
cnt_birth_1960s	Numeric	Number of total users born in 1960s
cnt_birth_1970s	Numeric	Number of total users born in 1970s
cnt_birth_1980s	Numeric	Number of total users born in 1980s
cnt_birth_1990s	Numeric	Number of total users born in 1990s
cnt_birth_2000s	Numeric	Number of total users born in 2000s
cnt_birth_outlier	Numeric	Number of total null value and outlier value
cnt_unknown	Numeric	Number of gender unknown users

cnt_male	Numeric	Number of male users
cnt_female	Numeric	Number of female users
temperature	Numeric	Temperature (in Celsius)
humidity	Numeric	Relative humidity (%)
windspeed	Numeric	Wind speed (in Kph)
weatherType	Numeric	Snow=1, Rain=2, fog=3, hail=4, else=0
holiday	Numeric	Public holiday (1 = holiday, 0 = not a holiday)
weekday	Numeric	Mon =1, …, Sun = 7

3.3.1 Null Value and Outliers

Some of bike users' information is incomplete: Variable cnt_unknown count the number of gender unknown users. Variable cnt_birth_outlier count the number of null value and outlier values. For example, some users registered their birth year in 19th century which is obviously error values. In this project, it is hypothecs that people who is more than 85 is unlikely to rent a bike, so the birth years before 1930s are treated as outliers. Besides this, there are 244 null values in total, which are all from weather related variables. Each of temperature, humidity, windspeed, weatherType variable contains 61 null values. Especially, variable windspeed contains 1064 records value of -9999, which are obvious error value. Besides these issues above the dataset is clean, generally speaking the quality of this dataset is relative good.

Table 13 Overview of data quality issue

Variable	Issue	Number
temperature	missing value	61
humidity	missing value	61
weatherType	missing value	61
windspeed	missing value	61
	error value -9999	1064

To handle these problems, firstly, because cnt_birth_outlier and cnt_unknown (gender) will not be used for predictive model building, therefore the null value will be kept as where they are. Secondly, the value -9999 in windspeed is obviously error value so they will be replaced by 8.76 which is the mean of all windspeed values (Han et al., 2011). The 244 weather related missing values will be replaced by the average value of nearest four hours' value. For example, there is a temperature null value in 16:00 Jan 1, 2014, this value will be replaced by the average temperature value of 14:00, 15:00, 17:00 and 18:00, which is the most probable value(Han et al., 2011). The tool used is Microsoft Excel.

3.3.2 Overview of Rental Bike Count

The distribution of bike count is look like a part of standard normal distribution, it is make sense, firstly because the rental bike number is impossible to be a negative value, secondly because at whole night time there were only a few bike rented, so the distribution is looks like a “part” of the normal distribution. The skewness of the distribution is 1.243036, which means the data is highly skewed to the right. The kurtosis is 4.102326 implying that the distribution is very platykurtic. During January 2014 and June 2015, the minimum number is 1 rental bike per hour, which all happened in the night in cold days. There is one maximum number of 4365 rental bikes per hour which happened in 18:00 June 29, 2015. Usually it is about 100 to 500 bikes are rented per hour.

The total rental bike number of January to June in 2015 is almost the same as the whole year of 2014, which means more people registered as Citi bike users. The distribution by month shows that users are not prefer to rent a bike in winter months, it is not comfortable to ride a bike in cold days. More bikes are rented in working days (Monday to Friday), and each day the peak hour is 8:00 in the morning and 17:00 to 18:00 in the afternoon. It is the time people go to work and after work. It seems that working people (or students) are one of the main user groups of Citi Bike.

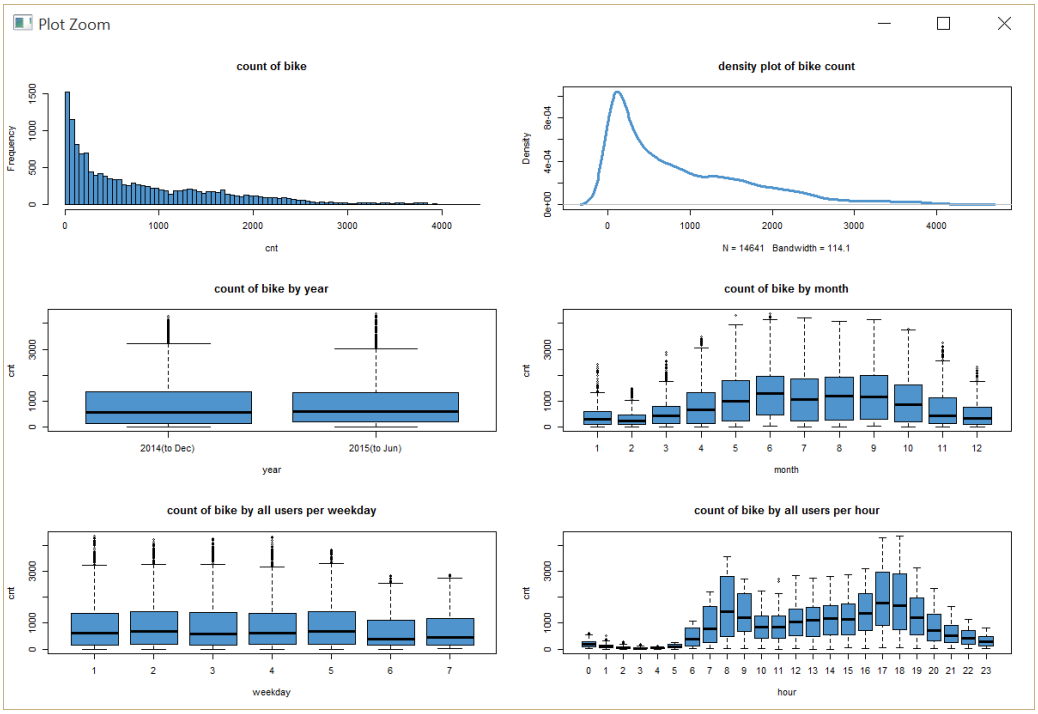


Figure 17 Visualization group 1

Notice that there are some outliers while plotting the weekday count, they are not error values, they might be a result of groups of people taking up cycling for bicycle events, or maybe bus drivers or train drivers strike, so they are considered as natural outliers.

3.3.3 Information of Users

When plotting the count by user group (long term and short term), the patterns are dramatically different. The subscriber group who are registered for one year shows "working" feature: more people rent a bike at 8:00 am and 17:00 to 18:00 pm. While the customer group who are registered for 24 hours or 7 days do not show "working" feature: they rented bikes mainly in daytime, and prefer 9:00 to 19:00. The plots are like a normal distribution. Notice that there are many outliers in short term users' plot and it might because a group of people who not registered rented bikes.

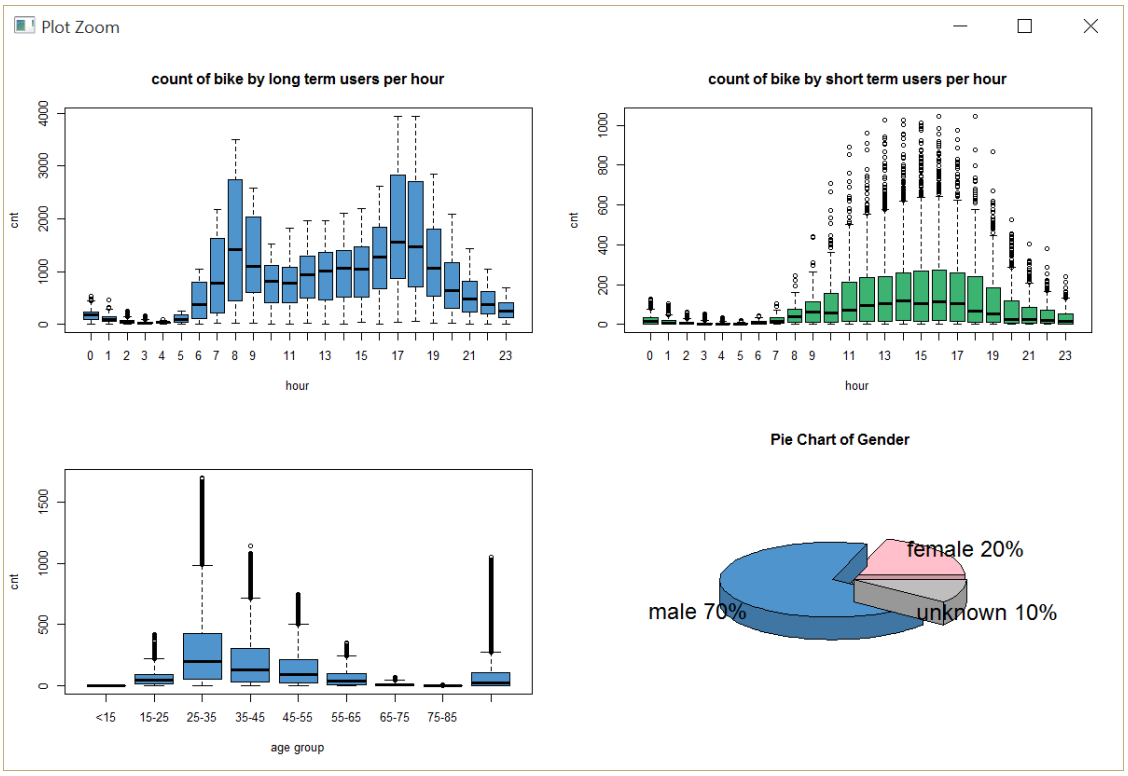


Figure 18 Visualization group 2

For the gender, there are at least 70% Citi Bike users are male. For the age, the users are mainly in 25 to 45, there is no value in <15 age group, the reason might be people under 18 cannot registered as Citi Bike users, or might be they do not hold a credit card to register the bike system. It is glad to see that users who are 65 to 85 years old prefer to rent bikes.

3.3.4 Information of Weather

When plotting the count by weather related variables, it is found that temperature and wind speed affect the behaviour of renting a bike more dramatically. With the temperature grows the number of rental bikes grows. It seems that about 17 to 26 degree Celsius is the comfortable temperature for people to ride bikes, and the number goes down when temperature is more than 28. People feel uncomfortable to ride a bike in big wind, it seems that many people are not prefer to ride bikes when wind speed is more than 20 kilometres per hour, and only a few people rent bikes when wind speed is more than 30 kilometres.

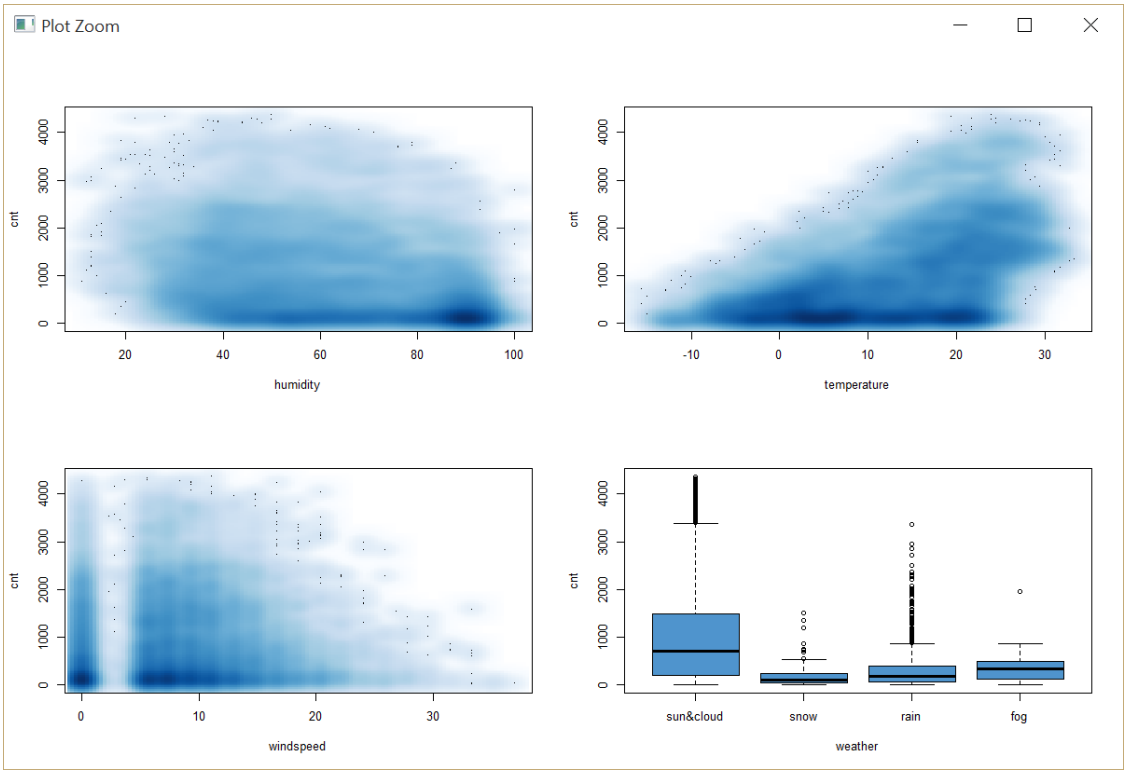


Figure 19 Visualization group 3

When the weather is very dry, which the relative humidity is less than 20%, people feel uncomfortable and not prefer to rent bikes. Weather seems an important factor, if there is fog, rain or snow, the number of rental bikes decrease a lot, especially snow.

It is interested to find out that short term users are more affected by temperature. It seems that no matter cold or hot, working people insiste on renting bikes, while casual users prefer rent bikes in confortalbe weather.

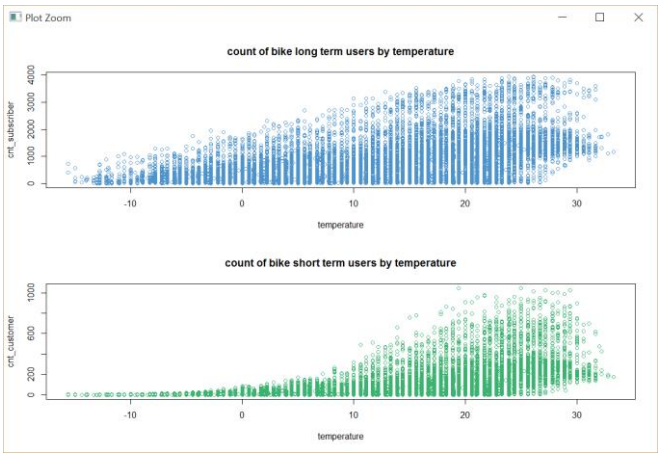


Figure 20 Count of different groups users by temperature

3.3.5 Information of Holiday and Weekend

The distribution of working days, weekends and holidays are very different. In working days the plot shows obvious "working" features, the rental bike number reach the peak at 8:00 am and 17:00 to 18:00 pm, which is the time working people (or maybe students) go to work and after work. In weekend, it looks like normal distribution and the peak time is 14:00 pm. It is interested to find out that in 13:00 pm the count is relative low, maybe it is the time people prefer to take a break to enjoy a cup of tea. In holidays, the peak time is 18:00 pm and it seems that in holidays people prefer to go out have a dinner.

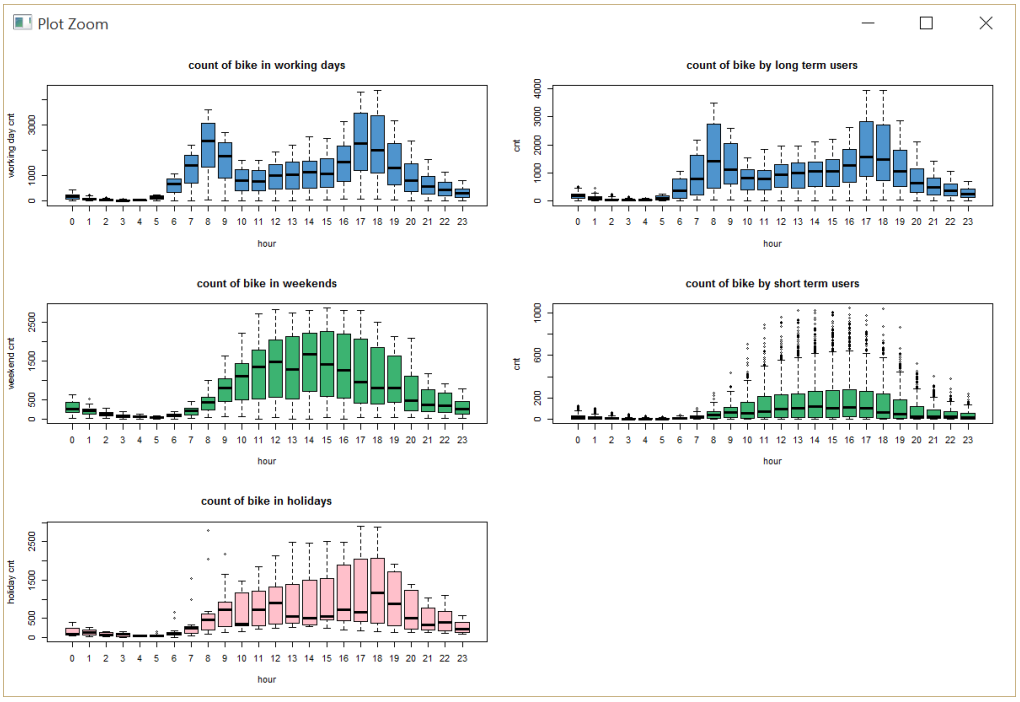


Figure 21 Visualization group 4

It is interested to find out that the distribution in working days and by long term users are similar. It can be hypothesis that annual members of Citi Bike include many working people who prefer to rent bikes as a transport to go to work and go back home. At the same time, the distribution in weekends and by short term users are similar, it can be hypothesis that short term users are tend to rent bikes for casual use such as touring.

3.3.6 Correlation

Before building predictive models, it is important to find out the relationship of all features with target variable (cnt). The correlation plot (more details in appendix A) below shows that cnt has a strong relationship with temperature, and relative strong relationship with hour, month, humidity, weather type. When building models, these features will get more attention.

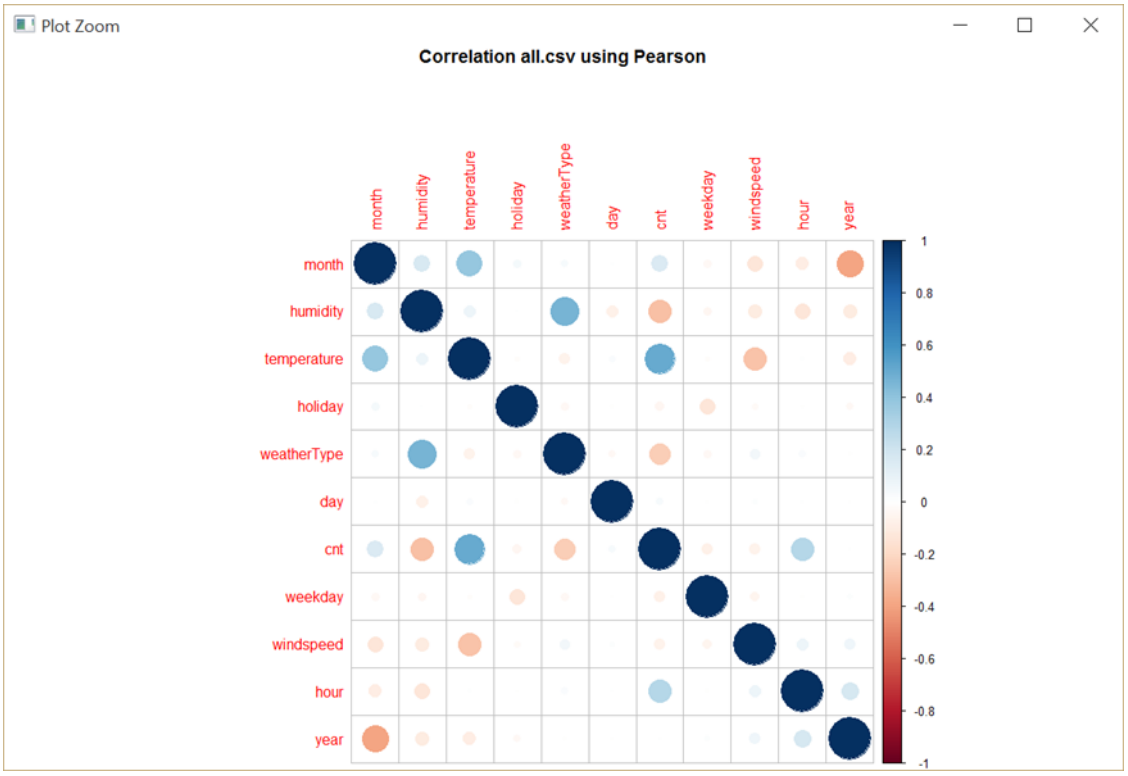


Figure 22 Correlation of features

3.3.7 Conclusion

Generally speaking, besides noisy values in variable windspeed, and some missing values in weather related variables, the quality of the source dataset is good. It is found that there is no person born after year 2000 to rent a bike, and no hail weather during

Jan 2014 to Jun 2015, so variable cnt_birth_2000s and value 4 in variable weather will be removed.

About feature importance, time related variables (such as hour, weekday, month, and year), weather related variables (such as humidity, temperature, wind speed, rain, fog and snow) are all significant factors to affect people's behavior to rent a bike. All the variables should be included when building models. Specifically, variable temperature, hour, month, humidity, and weather type have relatively strong relationship with target variable cnt, so they should get more attention.

It is also found that the behaviors of long term users are significant different from short term users, the count of the two types' users can be predicted separately. The same method is also used by Ray[①] who was in top 5% of participants in Kaggle competition.

Compared with the Kaggle competition: Bike Sharing Demand[②], there are some similar conclusions when doing the data exploration:
(1) Similar distribution[③] by weather related variables.

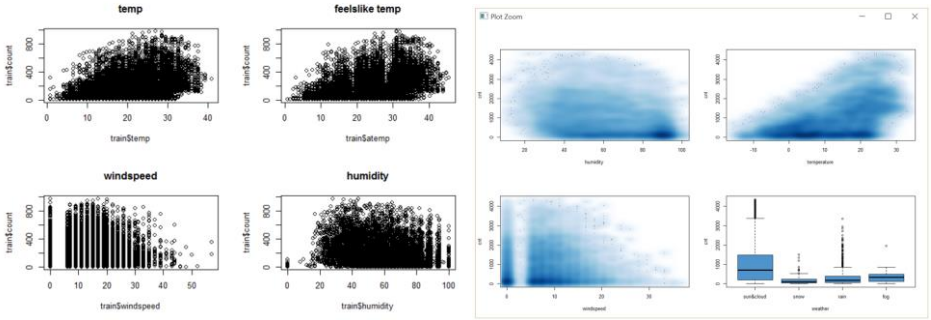


Figure 23 Comparison of weather distribution

(2) Similar distubution[④] by diffenrt type of users: Kaggle: regsisterd users and caual users; This project: long term users and shout term users.

[①] http://www.analyticsvidhya.com/blog/2015/06/solution-kaggle-competition-bike-sharing-demand/

[②] Kaggle Bike Share Demand: https://www.kaggle.com/c/bike-sharing-demand

[③] Figure from: International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015. URL: http://ijiset.com/vol2/v2s4/IJISET_V2_I4_195.pdf

[④] Figure from: http://www.analyticsvidhya.com/blog/2015/06/solution-kaggle-competition-bike-sharing-demand/

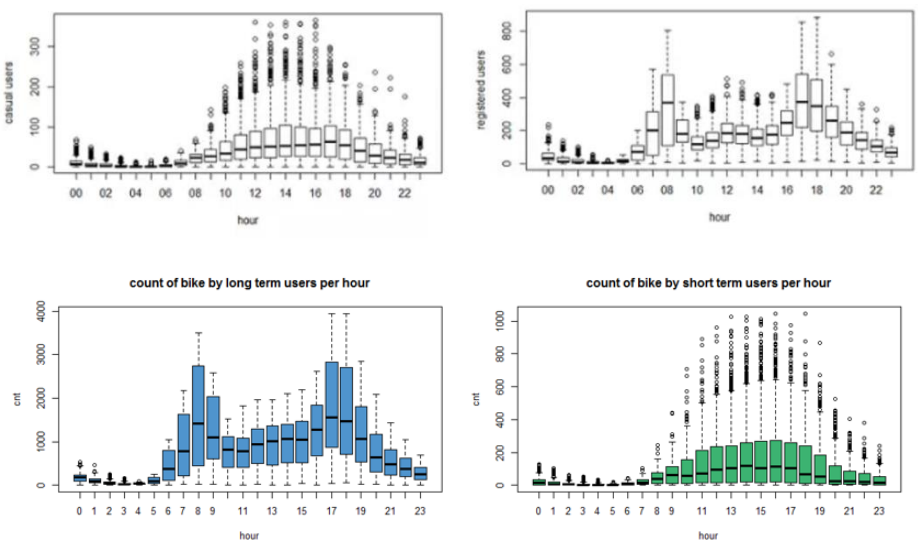


Figure 24 Comparison of user groups destitution by hour

Also, there are also some different conclusions when doing the data exploration:

(1) Variable holiday is not significant in Kaggle competition, and Du et al.[①] even removed this variable before they built models. But it is a significant variable in this project.

(2) More user related information such as gender and age groups are explored in this project.

It should be noticed that variable "windspeed" contains 1064 noisy values and 61 missing values, this variable may potentially affect the accuracy of predictive models. Furthermore, the distribution of this variable shows that there is a big "gap" between 0 and 5.6 Kph.

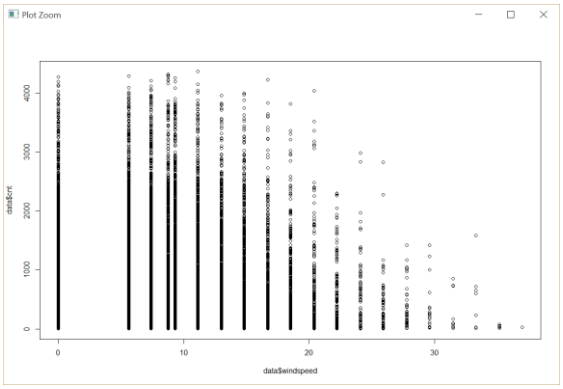


Figure 25 Distribution of wind speed by count

[①] http://cs229.stanford.edu/proj2014/Jimmy%20Du,%20Rolland%20He,%20Zhivko%20Zhechev,%20Forecasting%20Bike%20Rental%20Demand.pdf

All the reviewed papers in this project are not mentioned this issue, but in Sunil Ray[①]'s code, he splits the dataset to two part: wind_0 which the wind speed is 0, and wind_1 which the wind speed is not 0. He uses wind_1 dataset to build a random forest model, and then predicts the wind_0 value. This method is based on a hypothec that 0 value of wind speed is "missing" value. It is reasonable, because it is the wind speed value in one hour, it seems impossible that there is absolutely no wind during one hour. This project will test this hypothec when building predictive models.

3.4 Building Models

Based on the case study of modelling algorithms in similar project of Kaggle competition (see table 1 in chapter 2.3), four of six cases found random forest perform best, Du et al.[②] found tree based models show best performance, the error rate of which are Ctree: 0.46 and random forest 0.50. Lee et al.[③] found Neural Network performs the best with 0.49 error rate, but she did not build random forest model. Beside them, Ray's[④] random forest model gets top 5% in Kaggle Competition leader board. All the details are summarised as below:

Table 14 Summarize of best performance modelling method by case study

Reference	Modeling Methodology	Best model	Performance	Details
Yin et al.	(1) Ridge Regression (2) Support Vector Regression (3) Random Forest (4) Gradient Boosted Tree	Random forest	RMSLE 0.31	Good solution: Smallest error rate
Du et al.	(1) Basic Linear Regression (2) Generalized Linear Models with Elastic Net Regularization (3) Generalized Boosted Models (4) Principal Component Regression (5) Support Vector Regression (6) Random Forest (7) Conditional Inference Trees	Ctree Random forest	RMSLE 0.46 0.50	Two tree based models perform best

① http://www.analyticsvidhya.com/blog/author/sunil-ray/

② http://cs229.stanford.edu/proj2014/Jimmy%20Du,%20Rolland%20He,%20Zhivko%20Zhechev,%20Forecasting%20Bike%20Rental%20Demand.pdf

③ http://cs229.stanford.edu/proj2014/Christina%20Lee,%20David%20Wang,%20Adeline%20Wong,%20Forecasting%20Utilization%20in%20City%20Bike-Share%20Program.pdf

④ Sunil Ray: a business analytics and Intelligence professional
URL: http://www.analyticsvidhya.com/blog/author/sunil-ray/

Lee et al.	(1) Neural Network (2) Poisson Regression (3) Markov Model (4) Mean Value Benchmark	Neural network	RMSLE 0.49	Not building random forest model
Patil et al.	(1) Random Forest (2) Conditional Inference Trees (3) Generalized Boosted Models	Random forest	RMSLE 0.49	Random Forest with TuneRF mothod
Ray	(1) Random Forest	Random forest	RMSLE 0.38675	Good solution: Kaggle leader board top 5%
Wayne Liu	(1) Regression (2) Generalized Boosted Models	Regression	RMSLE 1.38378	Not good solution

Based on the case study of similar project of Kaggle Competition, it is found that random forest shows best performance in 4 cases. Neural network show best performance in 1 case but random forest is not tested in this case. Regression is tested in one case but not good performance compared with other cases.

This project will learn from similar researchs' experience to focus on random forest model. Besides random forest, this project will also try other three data mining classic models: regression, decision tree and neural network, by comparing their performance to choose the best modelling algorithm.

During the data exploration process, it is found that long term users' rental pattern is dramatically different from short term users' rental pattern, and short term users' rental behaviour are more likely affected by weather and temperature factors, so two models will be built separately base on these two groups. The aim of modelling is to predict the total rental demand, sum of the two model's result are the total users' rental demand.

In order to enhancing the performance of model, several techniques will be tried and to improve model's accurate. For example, based on the distribution of hour and users' count, it is found that there are two peaks of long term users, this pattern will be explained by adding a new variable which is built by decision tree model. Based on the distribution of short term users by hour, it is found there are many outliers, log transformation will be used for reduce the variability of data and outlying observations (Feng et al., 2014). All the processes of model building are designed as below:

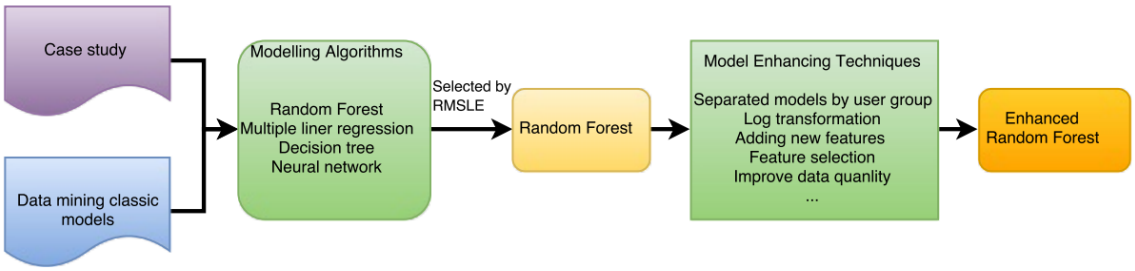


Figure 26 Model building process design

3.5 Evaluation Methods

There are many method to compare the predictive value with the true value in regression, such as Root Mean Squared Error (RMSE) (Willmott, 1981):

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\log(a_i - b_i))^2}$$

Because the similar project in Kaggle competition use RMSLE (Root Mean Squared Logarithmic Error) to measure the accuracy of each solution, and all competitors of case study are use the same method, the table below shows the evaluation methodology from case studies:

Table 15 Summarize of evaluation method by case study

Reference	Evaluation Methodology
Yin et al.	(1) 70/30 : training set / testing set (2) RMSLE by 10-fold cross validation
Du et al.	(1) 70/30 : training set / testing set (2) RMSLE by 10-fold cross validation
Lee et al.	(1) 70/30 : training set / testing set (2) RMSLE
Patil et al.	(1) 70/30 : training set / testing set (2) RMSLE
Ray	(1) 70/30 : training set / testing set (2) RMSLE
Wayne Liu	(1) 70/30 : training set / testing set (2) RMSLE

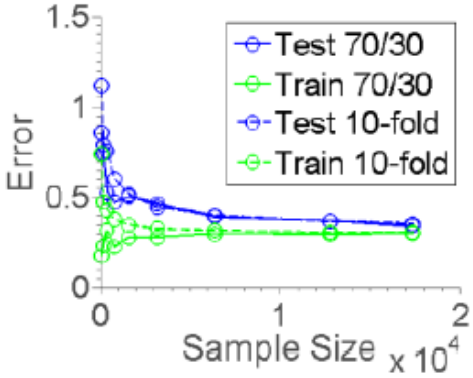
This research will use RMSLE to measure error rate. RMSLE is "calculated as the square root of the squared bias plus squared standard error" (Mickey & Greenland, 1989):

$$\text{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(a_i + 1) - \log(b_i + 1))^2}$$

Where n is the total number of observations in the testing dataset, a_i is predicted rental bike number, and b_i is the actual number, smaller RMSLE value indicates better model.

70/30 method will be used to split data to training set and testing set. The training set is used to build the predictive models, the testing set is unseen data when building models, which is used to evaluate the models performance.

Figure 27 Performance of 70/30 method and 10-fold CV by sample size increase[①]



While 10-fold cross validation is a better method for model selection because it is lower bias(Kohavi & others, 1995) and lower over-fitting risks. Based on the case study that some related researches use 10-fold cross validation, and Yin et al.[①] mentions that generally 10-fold cross validation perform better, while with the sample size increase, the 70/30 method and 10-fold validation show similar performance. This project will also use RMSLE with 10-fold cross validation to test the performance of models.

3.6 Conclusion

This chapter has summarized the whole data collection and integration process, explored and visualized data quality issues and some interesting patterns between independent variables and dependent variable. Model building and evaluation methods are discussed and designed.

[①] http://cs229.stanford.edu/proj2014/Yu-chun%20Yin,%20Chi-Shuen%20Lee,%20Yu-Po%20Wong,%20 Demand%20Prediction%20of%20Bicycle%20Sharing%20Systems.pdf

4. IMPLEMENTATION / RESULTS EVALUATION

4.1 Introduction

This chapter will outline how the dataset was pre-processed for model building, and then compare the accuracy of basic models built by decision tree, multiple linear regression, neural network and random forest. Finally relative best model will be chosen, focus on improve model's performance by different technologies and methods. Some model building and enhancing experiences learning from similar project case studies will be tested in this chapter.

4.2 Data Pre-processing

Before building the predictive models, there are some possible ways (Yin, Lee, & Wong, n.d.) to potentially increase the accuracy of models:

Firstly, remove useless variables. Customer birth and gender related variables such as cnt_male, cnt_birth_1930s will be removed, because they are not provide useful information for modelling process.

Secondly, because the distributions by users' group (cnt_subscriber and cnt_customer) are dramatically different (chapter 3.4.3), so this project will predict the count of long time users and short time users separately, the total count of rental bikes are the sum of the two groups users' count.

Secondly, variable factorization will be incorporated. Some variables' values have real meaning in the real world, such as 10 Kph of wind speed, while some variables are not, for example, 2 of month. We know 2 means February but R do not know, to let R know 2 is not means "how big the month is" but indicates to a certain month, variable factorization is necessary. Yin et al.[①] and Du[②] split categorical variables (such as

[①] http://cs229.stanford.edu/proj2014/Yu-chun%20Yin,%20Chi-Shuen%20Lee,%20Yu-Po%20Wong,%20Demand%20Prediction%20of%20Bicycle%20Sharing%20Systems.pdf

[②] http://cs229.stanford.edu/proj2014/Jimmy%20Du,%20Rolland%20He,%20Zhivko%20

season) into binary indicator variables, in this project, factorization process will be done by “as.factor( )” function in R.

4.3 Basic Models Built on Regression, Neural Network, DT & RF

Rattle (Williams, 2011) is a R package which provides an GUI (Graphical User Interface) to many other R packages for data mining users, by using Rattle, R users can easily and quickly to build multiple models and compare their performance. In this project, there are four steps to build models using Rattle:

Table 16 Steps of four basic models built by Rattle

Step	Details
1	Import data and set 70/30/0 as train set and validation set.
2	Select appropriate independent variables for model building and set variables as appropriate format.
3	Build regression model, decision tree model, neural network model and RF model.
4	Validation all the models and export outcomes

The outcome of the four basic models’ validation are described by distribution of predicted values and observation values, the figure 28 shows the distribution of models.

Because decision tree model is a classification model, the result is many “parallel lines”. Compared with other models, the distribution of predicted value and observation values in random forest model is most like a “line”, which means random forest model’s prediction are more accurate than other models. Next process will focus on random forest model to enhance the performance.

Zhechev,%20Forecasting%20Bike%20Rental%20Demand.pdf

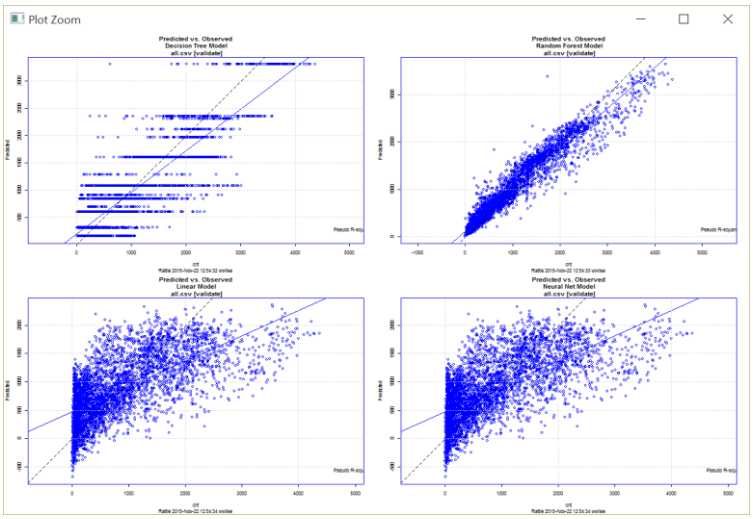


Figure 28 Comparison of four basic models' performance

4.4 Random Forest Model

Because it is found random forest perform best compared with other three basic models in Rattle, the following process is using R code to build a basic random forest model (Rattle is not good for model enhancing in this project) and use techniques to enhancing the performance. There are five main steps to build a basic random forest (RF) model and evaluate the performance by RMSLE with 10-fold cross validation using R:

Table 17 Steps of random forest model built by R

Step	Details	Core Code
1	Input dataset	setwd("C:/Users/wwlise/Desktop/CitiR") data <- read.csv("all.csv")
2	Encode a vector as a factor	e.g. data$weatherType=as.factor(data$weatherType) data$holiday=as.factor(data$holiday)
3	For loop for 10-fold cross validation	k=10 n=floor(nrow(data)/k) err = rep (NA,k) for(i in 1:k) { s1 = ((i-1)*n + 1) s2 = (i*n) subset=s1:s2 train = data[-subset,] test = data[subset,]
4	Build RF model	rf <- randomForest(as.factor(cnt)~ .,data=train,importance=TRUE) *## using package "randomForest"*
5	Evaluate RF model	test$prd<-predict(rf,test) rmsle(test$prd, test$cnt) } *## using package "Metrics"*

About the number of trees, the default tree number of random forest is 500. The error plot shows that when the tree number is smaller than 80, the error decrease dramatically, when bigger than 80, the line goes stable, when more than 300, the error is very stable, so this project set the tree number (ntree) to 300.

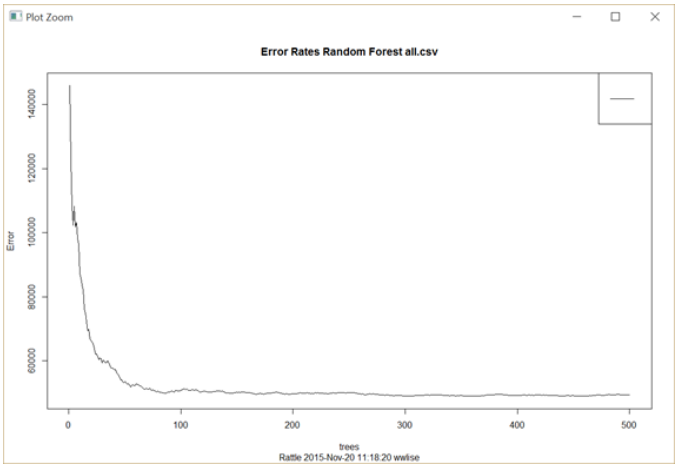


Figure 29 Plot of error with tree number

The RMSLE with 70/30 method of basic random forest model is 0.5531849 (result of screen shot see Appendix E), with 10-folder cross validation the RMSLE is 0.4989367, the decrease of error rate is reasonable, because in 10-fold cross validation runs 10 times random forest model, each time using 90% data as training set to build model and 10% data as testing set to validation, compared with 70/30 method it contains more information in training set.

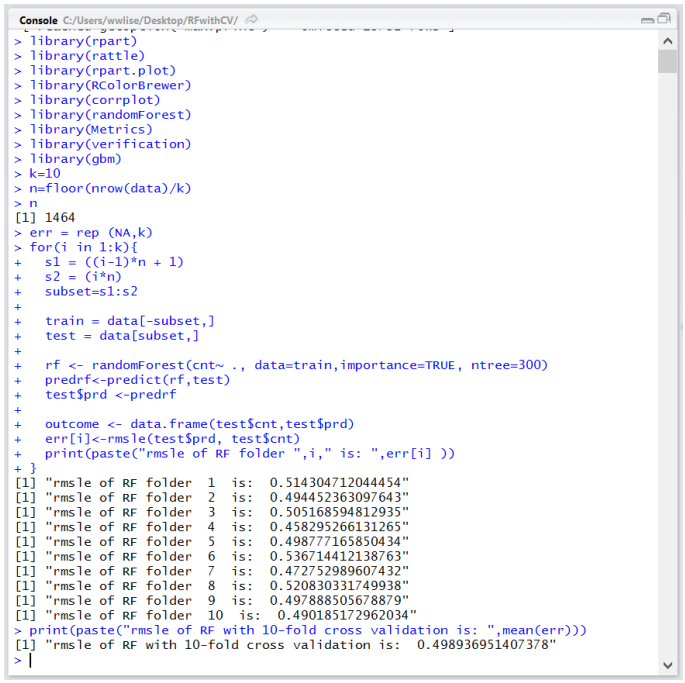


Figure 30 Result of basic RF model with 10-fold CV

The feature below shows the relationship between predictive values and observation values of target variable cnt. The plot generally seems like a thick line, which means the predictive model is good, but not good enough.

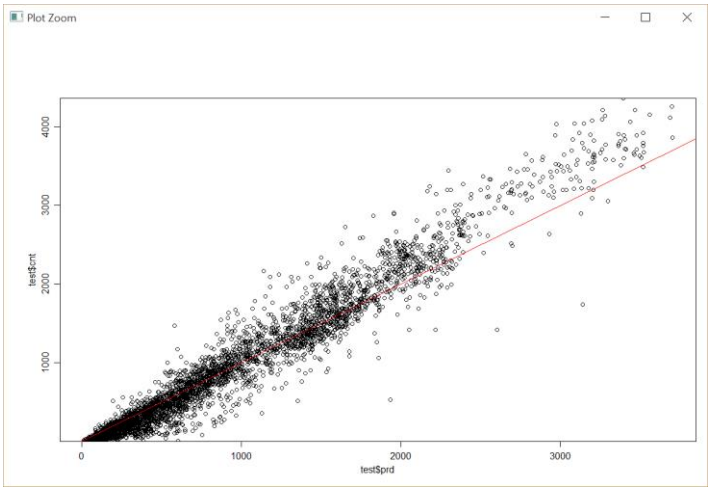


Figure 31 Initial random forest Pr vs Ob plot

"varImpPlot" function in randomForest package can help to see the important the independent variables. The %IncMSE graph shows that if a variable is assigned values by random permutation by how much will the MSE (mean squared error) increase[①]. Node purity is measured by Gini Index which is the difference between RSS (Residual Sum of Squares) before and after the split on that variable. Because there are only ten independent variables by describe different information, and no strong correlation with each other, there is no need to do feature selection in this basic model.

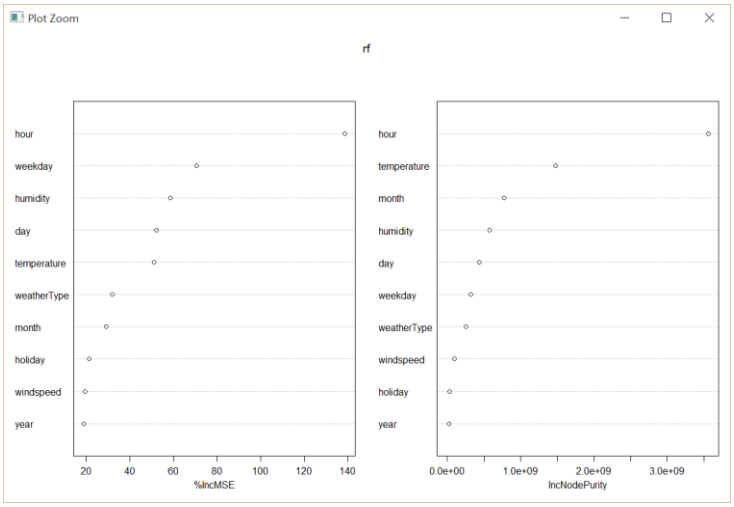


Figure 32 Importance of all the independent variables

① http://discuss.analyticsvidhya.com/t/how-to-extract-important-variables-from-random-forest-model-using-varimpplot-in-r/1325

4.5 Enhancing RF Model by Adding More Independent Variables

There are some conclusion have been found when exploring the dataset, for example, the distribution of long time users and short time users are significant different; the peak time of renting bikes is 8:00 in the morning and 17:00 to 18:00 in the afternoon. In the initial RF model, all these distinguishing features are not prominent. This project hypothec that adds some independent variables which are based on the distinguishing features will improve the accuracy of predictive model. Based on this hypothetic, some new variables are added in the dataset:

Table 18 Table of new variables

New Variable	Related variable	Describe
h_sub	cnt_subscriber ~ hour	6 groups by hour
h_cus	cnt_customer ~ hour	4 groups by hour
temp_sub	cnt_subscriber ~ temperature	4 groups by temperature
temp_cus	cnt_customer ~ temperature	4 groups by temperature
year_part	cnt ~ year + month	7 groups by year and month
day_type	holiday + weekday	3 groups of workday, weekend, holiday
weekend	weekday	2 groups of workday,weekend

Variables day_ type and weekend can be created directly by the value of holiday and weekday. All the others are created by decision tree models. Take h_sub for example, decision tree is built by variables cnt_subscriber and hour which means to classify the count of long term users by hour.

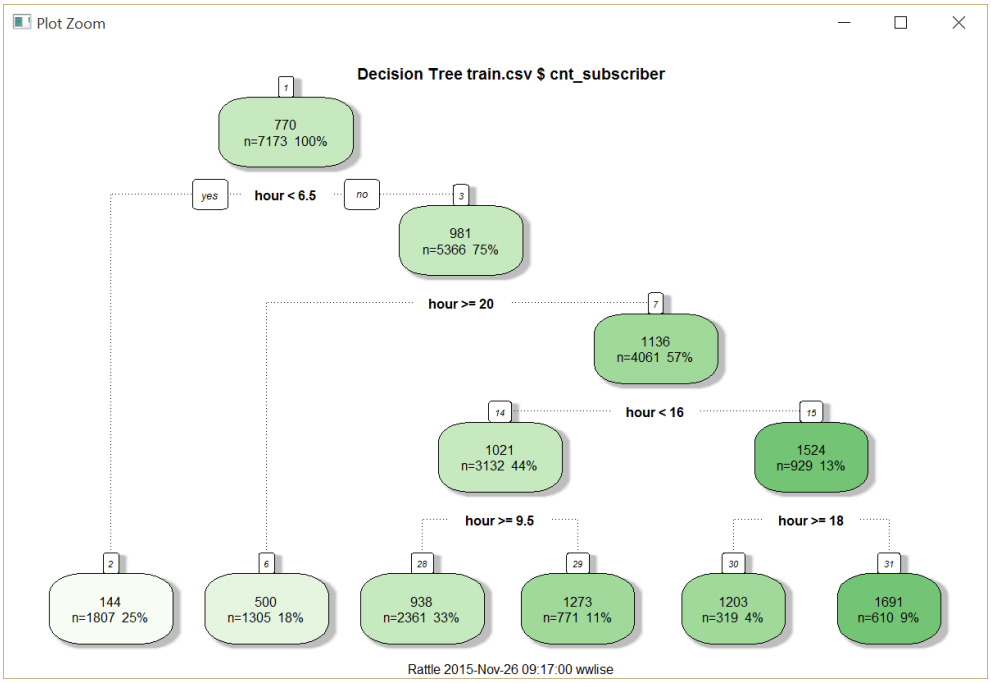


Figure 33 Decision tree: count of long time users by hour

The decision tree shows that the count of long time users is classified by hour to six groups:

Table 19 Describes of decision tree

Groups	Details
1	Hours: 1, 2, 3, 4, 5, 6
2	Hours: 20, 21, 22, 23, 24
3	Hours: 10, 11, 12, 13, 14, 15
4	Hours: 7, 8, 9
5	Hours: 16, 17
6	Hours: 18, 19

By using decision tree, independent variables cnt_subscriber is classified to six groups by hour, new variable h_sub is created. Variables h_cus, temp_sub, temp_cus, year_part are created by the same way, more details please see appendix B. Compared with initial model's RMSLE 0.5531849 (70/30 method), the new model's error rate decrease to 0.3945004 with 70/30 method (result see Appendix E) and 0.3534387 with 10-fold CV. The result of RMSLE decrease proves that adding more independent variables helps to improve the random forest model's accuracy in this project.

```
+
+   test$prd <- predrf1 + predrf2
+
+   outcome <- data.frame(test$cnt,test$prd)
+   err[i]<-rmsle(test$prd, test$cnt)
+   print(paste("rmsle of RF folder ",i," is: ",err[i] ))
+ }
[1] "rmsle of RF folder  1  is:  0.368166084254288"
[1] "rmsle of RF folder  2  is:  0.356599111416482"
[1] "rmsle of RF folder  3  is:  0.353116346110697"
[1] "rmsle of RF folder  4  is:  0.352734107019894"
[1] "rmsle of RF folder  5  is:  0.341047920281777"
[1] "rmsle of RF folder  6  is:  0.329389533242028"
[1] "rmsle of RF folder  7  is:  0.341799874009389"
[1] "rmsle of RF folder  8  is:  0.392831740706585"
[1] "rmsle of RF folder  9  is:  0.360291316075533"
[1] "rmsle of RF folder  10  is:  0.338410585635164"
> print(paste("rmsle of RF with 10-fold cross validation is: ",mean(err)))
[1] "rmsle of RF with 10-fold cross validation is:  0.353438661875184"
>
```

Figure 34 Result of improved RF model by adding new features (with 10-fold CV)

4.6 Enhancing RF Model by Improving Independent Variable's Quality

It has been noticed that variable windspeed contains 1064 noisy values and 61 missing values, this variable may potentially affect the accuracy of predictive models. Furthermore, the distribution of this variable shows that there is a big "gap" between 0 and 5.6 Kph. Sunil Ray[①]'s hypothetic that it is impossible there is no wind during one

① http://www.analyticsvidhya.com/blog/author/sunil-ray/

hour, Ray's solution is using random forest model to predict the 0 value of wind speed. This hypothetic will be tested in this project.

There are two quality issues of variable "windspeed": Missing values (noisy values of -9999 are treated as missing values) and 0 values. There are two methods to handle missing values: replaced by average value (Han et al., 2011) and using RF model to predict the missing values (Ray). This project built three models using 70/30 method to test which is the best method (results see Appendix C), the error rate of method three is the least with 0.3827817, which is a little smaller than method one 0.3945004, it seems Ray's hypothetic is true, but the result of 10-fold CV shows RMSLE of method three is 0.3550672, which increase 0.0017 than method one, the results indicates that method 3 does not work in this project.

Table 20 Outcome of three methods

methods	Describe	RMSLE (70/30 method)	RMSLE (10-fold CV)
1	Null value replaced by average value	0.3945004	0.3534387
2	Null value replaced by average and 0 value predicted by RF model	0.3938721	
3	Null and 0 value predicted by RF model	0.3827817	0.3550672

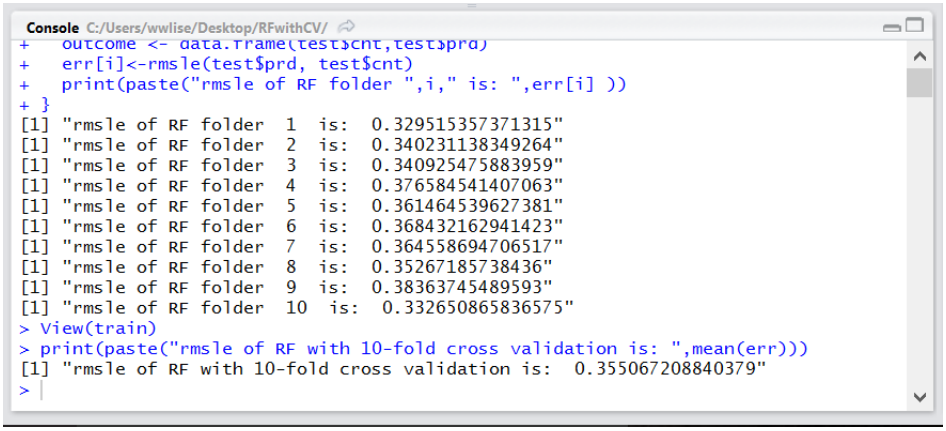


Figure 35 Result of improved RF model by handle missing/noisy value (with 10-fold CV)

4.7 Enhancing RF Model by Transforming the Dependent Variable

Log transformation is a popular method to reduce the variability of data, especially when dataset contains outlying observations (Changyong FENG et al., 2014). When building random forest models, the target variables are cnt_subscriber and

cnt_customer, the dataset contains some outliers, for example, the figure below shows there are many outliers when plotting cnt_customer by hour. To handle this problem, log transformation method will be tested is this project.

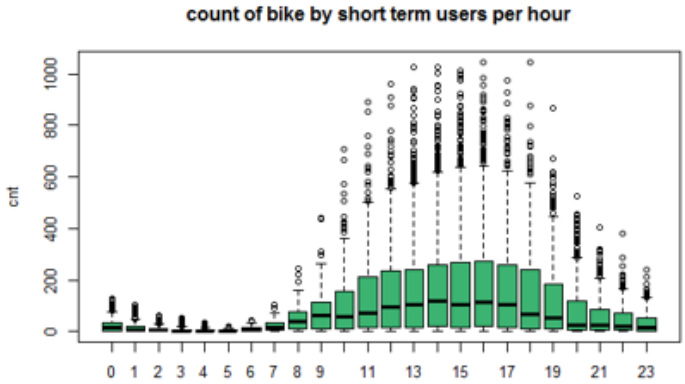


Figure 36 Sample plot of outliers

There are four steps to do the log transformation: Firstly is to plus one to target variable, this can make sure the value of target variable is not equal to 0. Secondly is to log the target variable plus 1, such as train$logcus=log(train$cus1). Thirdly is using the new target variable to build RF model. Fourthly is transforming the target variable to originally format, for example: test$precus=exp(test$logcus)-1.

After log transformation, the error rate of random forest model is decrease to 0.281201 with 70/30 method (see Appendix E) and 0.2649822 with 10-fold CV, which proves log transformation is a useful method to improve the accuracy of predictive models.

```
Console C:/Users/wwlise/Desktop/RFwithCV/
+   outcome <- data.frame(test$cnt,test$prd)
+   err[i]<-rmsle(test$prd, test$cnt)
+
+   print(paste("rmsle of RF folder ",i," is: ",err[i] ))
+ }
[1] "rmsle of RF folder  1  is:  0.266355818282657"
[1] "rmsle of RF folder  2  is:  0.25492093070453"
[1] "rmsle of RF folder  3  is:  0.294797304071379"
[1] "rmsle of RF folder  4  is:  0.262203909602855"
[1] "rmsle of RF folder  5  is:  0.262153329412423"
[1] "rmsle of RF folder  6  is:  0.274990358679286"
[1] "rmsle of RF folder  7  is:  0.249094508762032"
[1] "rmsle of RF folder  8  is:  0.284419410388736"
[1] "rmsle of RF folder  9  is:  0.253941866014786"
[1] "rmsle of RF folder  10  is:  0.246944106123296"
> print(paste("rmsle of RF with 10-fold cross validation is: ",mean(err)))
[1] "rmsle of RF with 10-fold cross validation is:  0.264982154204198"
>
```

Figure 37 Result of improved RF model by log transformation (10-fold CV)

4.8 Enhancing RF Model by Heat Index

Sometimes people feel the weather is hotter or colder than the actual temperature, which is not an illusion, but because of humidity affecting people's feeling of temperature. Heat index (please see appendix D[①]) is a measure of how weather "feels" to the people's bodies by combining humidity's impact to temperature.

The idea of using heat index to enhancing model is from the datasets of Kaggle competition[②]. In Kaggle's datasets there are two variables "temp" and "atemp" in which "atemp" is described as "feels like" temperature in Celsius, which is the heat index. This project hypothesizes that adding heat index will enhance the model's accuracy.

Brooke Anderson[③] shared the R source code to calculate the heat index in GitHub, it is very easy to calculate heat index for this project by the help of the source code. The result of RMSLE is 0.2799691 with 70/30 method (see Appendix E) and 0.2666055 with 10-fold cross validation, compared with method three in chapter 4.7, RMSLE with 70/30 decrease but with 10-fold CV increase. It is an interesting result, because 70/30 method shows adding heat index will enhance model's accuracy while 10-fold CV shows this enhancing method do not work. 10-fold cross validation shows the variable of Heat Index may have strong correlation with other variables. Feature selection will be discussed in chapter 4.9.

```
+
+   outcome <- data.frame(test$cnt,test$prd)
+   err[i]<-rmsle(test$prd, test$cnt)
+
+   print(paste("rmsle of RF folder ",i," is: ",err[i] ))
+ }
[1] "rmsle of RF folder  1  is:  0.295885364842247"
[1] "rmsle of RF folder  2  is:  0.289141612451369"
[1] "rmsle of RF folder  3  is:  0.255807694814792"
[1] "rmsle of RF folder  4  is:  0.246723841399322"
[1] "rmsle of RF folder  5  is:  0.275474277341226"
[1] "rmsle of RF folder  6  is:  0.253127204950834"
[1] "rmsle of RF folder  7  is:  0.276573771592875"
[1] "rmsle of RF folder  8  is:  0.25136075085086"
[1] "rmsle of RF folder  9  is:  0.248236216611649"
[1] "rmsle of RF folder  10  is:  0.273724629246412"
> print(paste("rmsle of RF with 10-fold cross validation is: ",mean(err)))
[1] "rmsle of RF with 10-fold cross validation is:  0.266605536410159"
>
```

Figure 38 Result of improved RF model by adding heat index (10-fold CV)

① http://www.srh.weather.gov/jetstream/downloads/heatindex_rh_f_20x12.pdf

② https://www.kaggle.com/c/bike-sharing-demand/data

③ https://github.com/cran/weathermetrics/blob/master/R/heat.index.R

4.9 Enhancing RF Model by Feature Selection

Because variable heat index is created by temperature and relative humidity, the picture below shows the correlation of the three variables and target variable.

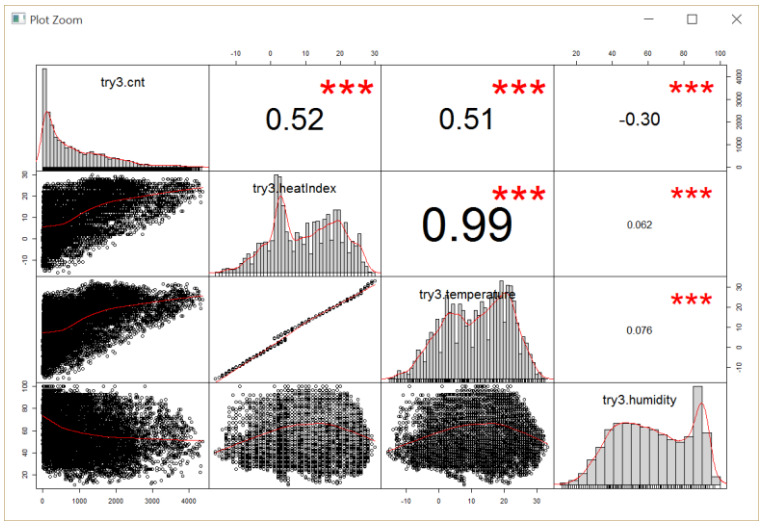


Figure 39 Correlations of four variables

Using Pearson method to calculate the correlation value of heat index with humidity is 0.06231202, with temperature is 0.9939419, the distribution of heat index and temperature is almost like a straight line which means they have very strong relationship. Based on the correlation values above, this project hypothetic that remove variable temperature is better than remove variable heat index.

The result of RMSLE with 10-fold CV is 0.2654593, which decrease slightly than not removing variable temperature, which means this variable indeed contains similar information as other variables. But this result is still bigger than not adding variable heat index and that is because heat index not only have strong relationship with variable temperature, but also with variable humid.

```
+
+
+   outcome <- data.frame(test$cnt,test$prd)
+   err[i]<-rmsle(test$prd, test$cnt)
+
+   print(paste("rmsle of RF folder ",i," is: ",err[i] ))
+ }
[1] "rmsle of RF folder  1  is:  0.266560825738195"
[1] "rmsle of RF folder  2  is:  0.256484655396122"
[1] "rmsle of RF folder  3  is:  0.297088451332161"
[1] "rmsle of RF folder  4  is:  0.262753166295954"
[1] "rmsle of RF folder  5  is:  0.2624155006142"
[1] "rmsle of RF folder  6  is:  0.276829837654644"
[1] "rmsle of RF folder  7  is:  0.248480652327747"
[1] "rmsle of RF folder  8  is:  0.280732974653511"
[1] "rmsle of RF folder  9  is:  0.254820012674277"
[1] "rmsle of RF folder  10  is:  0.248427152293245"
> print(paste("rmsle of RF with 10-fold cross validation is: ",mean(err)))
[1] "rmsle of RF with 10-fold cross validation is:  0.265459322898006"
>
```

Figure 40 Result of improved RF model by feature selection (10-fold CV)

4.10 Conclusion

This chapter has compared the four basic models: multiple linear regression model, decision tree model, neural network model, random forest model and selected random forest model to do the further enhancing based on RF model shows relative best performance.

Then this chapter has tested five techniques to improve the accuracy of RF model, using both 70/30 method and 10-fold cross validation, all the results are summarized as below:

Table 21 Enhancing methods and performance

Enhancing methods	Description	RMSLE (70/30 method)	RMSLE (10-fold CV)
0	Basic model	0.5531849	0.4989367
1	Adding more independent variables	0.3945004	0.3534387
2	Improving independent variable's quality	0.3827817	0.3550672
3	Log transforming the dependent variable	0.2812010	0.2649822
4	Add heat index	0.2799691	0.2666055
5	Feature selection	0.2793681	0.2654593

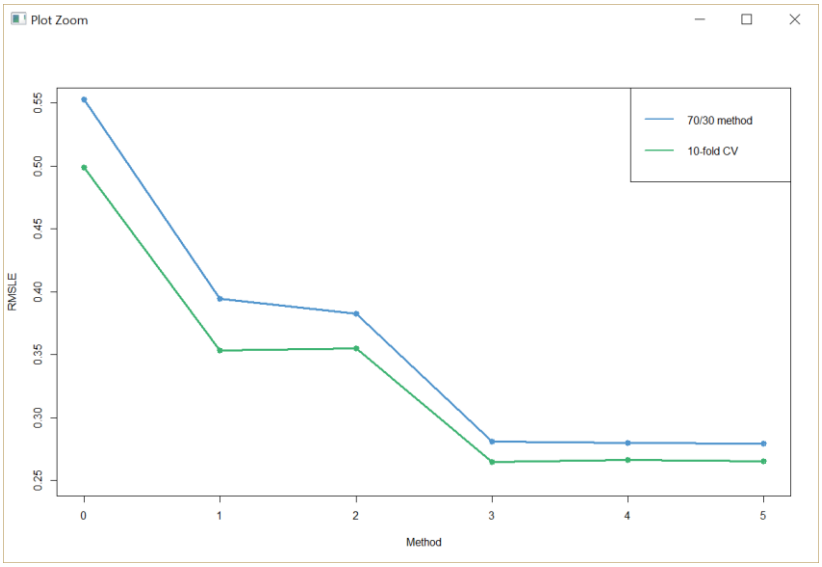


Figure 41 Compare of two error rate distribution by each enhancing method

The general performance by 10-fold CV is better than 70/30 method, besides 10-fold CV is more sensitive when adding a new variable which has correlation with other variables. 70/30 method shows all the five enhancing techniques can improve the model performance, while 10-fold cross validation shows in fact there are only two techniques work.

The RMSLE random forest model decrease from 0.499 to 0.265 after model enhancing process, which shows good performance of the final model. There are some experiences summarized from this chapter:

- Adding more independent variables based on the data's feature helps increase model's performance but needs to be aware of correlation problem.
- Log transforming of dependant variable helps to reduce the noisy caused by outliers.
- Always use K-fold cross validation when doing model selection.

5. CONCLUSION

5.1 Introduction

This chapter will review the whole project, summarize the contributions to the current bike sharing systems, and also find out the limitations during model building and evaluation processes. Future work and research will also be recommended.

5.2 Problem Definition & Research Overview

This project aims to forecast bike rental demand in a future certain time, this was done by building a random forest model that combines historical usage patterns with the related information of users, weather, holiday and weekend. To do this, a literature review of the modelling and evaluation algorithms was carried out, some case study of similar are taken to provide evidence of modelling building, model enhancing and model selection. This research enabled the following objectives to be achieved:

- Identified which modelling and evaluation algorithms performance best in similar project case study, and summarized their evidence in modelling building and selection process.
- Reviewed available literature on bike share program, predictive modelling algorithms and model evaluation methods.
- Used multiple tools such as Java, MySQL, Impala to collect data from three database resources. Designed a scalable ETL (Extract Transform Load) process to transform and combine the raw data in to a model ready format.
- Explored the quality issues and interested patterns of the dataset, and visualized the relationships between variables by R.
- Compared models built by four different methods and selected one by the performance.
- Designed and implement a five steps model enhancing process to improve the accuracy of random forest model.

5.3 Contributions to Bike Share Program

Based on the experiment of the project, the following findings that can be considered as contributions to bike share program both in data mining domain and realistic domain:

- Compared with related research in the field of forecast bike rental demand, this project not only focus on model building and evaluation process, but also followed the whole KDD process which including data selection and collection, ETL(extract, transform and load) process, and data pre-processing. Big data is handled during ETL process, and scalable methods and tools such as Hadoop, Impala, Java, MySQL are used for data collection, extraction, reduction and combination have been demonstrated.
- Visualize the interesting pattern of dataset during data exploration and model explanation process.
- Tested five methods to enhance the random forest model, compared the validation performance by 70/30 method and 10-fold CV and explored the reason.
- Decrease random forest model's RMSLE from 0.449 to 0.265.

- Dramatic increasing number of Citi Bike users in 2015, which not only need expand the docking station and improve the number of bikes, but also more effectively management especially on rebalancing bikes across bike docking stations.
- More than 70% users are male, there are strong signs that long term users (annual membership) show "working" features, and their behaviour are less affected by weather and temperature. While short term users (one day and 7 days membership) are casual users who prefer to rent bikes when they feel "comfortable", which means their behaviours are more affected by time, weather and temperature.
- The final random forest model can not only predict the whole users' demand of Citi bike in a certain time under certain weather, but also each docking station's bike rental demand, just replace the station's historical data to the total historical data.

5.4 Experimentation, Evaluation & Limitations

This project followed the whole KDD process, collected data (CSV format and Json format) from three different databases and integrated it to a model ready format, then built random forest model and enhanced it. There are some experiments from the whole process of this project:

- In-depth research on similar case studies were undertaken. This project summarized modelling algorithms, evaluation algorithms, model performance and conclusions from 7 similar cases.
- Random forest is a good ensemble algorithm which combines both decision tree and bagging. Whilst it takes more times running in R (especially using 10-fold CV), it is more accurate and less over-fitting.
- Adding more independent variables based on the data's feature (see chapter 3.3, 4.5 and 4.8) helps increase model's performance but needs to be aware of correlation problem.
- Log transforming of dependant variables helps to reduce the noisy caused by outliers.
- Always use K-fold cross validation when doing model selection.
- No predictive model is perfect, there will be always have better one.

The aim of this project is to forecast bike rental demand of the Citi Bike in New York. However, there are too many factors will affect bike users' behavior, for example, an event of bicycle racing or marathon, a short term government policy related to traffic, a tornado or typhoon. Actually, it is impossible to consider all the factors in one research, this project focus on the bike share program's rebalancing issue, seek to build a model to predict the number of rental demand in a future certain time based on weather, temperature, weekend and holiday.

Because the Citi Bike program was launched in mid-2013, the data of this project collected is from January 2014 to June 2015, therefore there is a time limit of the use of predictive model, which should be updated by using latest historical Citi Bike data every two or three months.

Bike rebalancing is a sub-topic within bike share research (Fishman, 2015), there are multiple methods to do this research such as “examine the impact of topography on station activity”, “examine the factors associated with higher and lower levels of docking station activity”, this research is more related to topic of “identified a relationship between weather and users’ activity” (Fishman, 2015). This project did not using clustering analysis because it is not focus on geographical position but focus on prediction of rental demand in a certain time, in other words, this project is focus on “when”, not “where”. However, if replace a certain docking station’s data to the total Citi bike historical data, the random forest model built in this project can predict each docking station’s rental demand, therefore can handle “where” problem.

5.5 Future Work & Research

There are many additional area of research worthy to do in the future:

The first area is to explore the rental activity of each docking station. This research already provide a final random forest model, which is the first stage to solve bike rebalance problem. The second stage is to replace the target docking station historical data to the whole Citi bike historical data, the model can predict users’ rental demand for each docking station. Another method is use cluster analysis to group all the docking station to several clusters by their activity, and predict each group’s bike rental demand.

The second area is to add more independence variables in this model by other factors which will affect users’ rental behavior. Users’ rental demand will affect by many factors, for example pollution, government policy. Especially when exploring the activity by each docking station, there are more factors, such as geographic location, company or shopping center nearby.

The third area is to try other enhancing techniques to improve the model’s performance. No predictive model is perfect, there is always better one. Beside this, the model should be updated using new historical Citi Bike data every two or three months, and based on new features of users’ activity, update the model.

BIBLIOGRAPHY

Abdelhalim, A., & Traore, I. (2009). A new method for learning decision trees from rules. In *Machine Learning and Applications, 2009. ICMLA'09. International Conference on* (pp. 693–698). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5381350

Alippi, C., & Roveri, M. (2010). Virtual k-fold cross validation: An effective method for accuracy assessment. In *The 2010 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–6). http://doi.org/10.1109/IJCNN.2010.5596899

Aung, W. T., & Hla, K. H. M. S. (2009). Random forest classifier for multi-category classification of web pages. In *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific* (pp. 372–376). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5394100

Bavarian, B. (1988). Special section on neural networks for systems and control. *IEEE Control Systems Magazine (April 1988), 3*.

Borgnat, P., Abry, P., Flandrin, P., Robardet, C., Rouquier, J.-B., & Fleury, E. (2011). Shared bicycles in a city: A signal processing and data analysis perspective. *Advances in Complex Systems*, *14*(03), 415–438.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Bylander, T. (2002). Estimating generalization error on two-class datasets using out-of-bag estimates. *Machine Learning*, *48*(1-3), 287–297.

Changyong FENG, Hongyue WANG, Naiji LU, Tian CHEN, Hua HE, Ying LU, & Tu, X. M. (2014). Log-transformation and its implications for data analysis.

Shanghai Archives of Psychiatry, *26*(2), 105–109. http://doi.org/10.3969/j.issn.1002-0829.2014.02.009

Chen, C.-M., Hsu, C.-Y., Chiu, H.-W., & Rau, H.-H. (2011). Prediction of survival in patients with liver cancer using artificial neural networks and classification and regression trees. In *Natural Computation (ICNC), 2011 Seventh International Conference on* (Vol. 2, pp. 811–815). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6022187

Duncan, R. (1980). What is the right organization structure? Decision tree analysis provides the answer. *Organizational Dynamics*, *7*(3), 59–80.

Efron, B., & Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=gLlpIUxRntoC&oi=fnd&pg=PR14&dq=bootstrap&ots=A8yzV9P6E3&sig=Z4KvNjUWaz3Ro-Y9KnZUoTBbEh4

Fan, C.-Y., Chang, P.-C., Lin, J.-J., & Hsieh, J. C. (2011). A hybrid model combining case-based reasoning and fuzzy decision tree for medical data classification. *Applied Soft Computing*, *11*(1), 632–644.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, *17*(3), 37.

Fishman, E. (2015). Bikeshare: A Review of Recent Literature. *Transport Reviews*, (ahead-of-print), 1–22.

Fishman, E., Washington, S., & Haworth, N. (2013). Bike share: a synthesis of the literature. *Transport Reviews*, *33*(2), 148–165.

Fishman, E., Washington, S., & Haworth, N. (2014). Bike share's impact on car use: evidence from the United States, Great Britain, and Australia. *Transportation Research Part D: Transport and Environment*, *31*, 13–20.

Gambhir, S. S., Hoh, C. K., Phelps, M. E., Madar, I., & Maddahi, J. (1996). Decision tree sensitivity analysis for cost-effectiveness of FDG-PET in the staging and management of non-small-cell lung carcinoma. *The Journal of Nuclear Medicine*, *37*(9), 1428.

Gurney, K. (1997). *An introduction to neural networks*. CRC press. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=HOsvllRMMP8C&oi=fnd&pg=PR9&dq=An+introduction+to+neural+networks+Kevin+Gurney&ots=SGLLwDcIVp&sig=JcwY-2th_JrQrWjNMy0VocNg1mU

Han, J., Kamber, M., & Pei, J. (2011). *Data mining: concepts and techniques: concepts and techniques*. Elsevier. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=pQws07tdpjoC&oi=fnd&pg=PP1&dq=data+mining+concept+and+&ots=tyHwZSlA0Z&sig=g97cJe2srE6cEiND6xjnw_VT9lE

Jing, C., & Zhao, Z. (2015). Research on Antecedents and Consequences of Factors Affecting the Bike Sharing System—Lessons From Capital Bike Share Program in Washington, DC. In *International Conference on Logistics Engineering, Management and Computer Science (LEMCS 2015)*. Atlantis Press. Retrieved from http://www.atlantis-press.com/php/paper-details.php?id=25838136

Joelsson, S. R., Benediktsson, J. A., & Sveinsson, J. R. (2005). Random forest classifiers for hyperspectral data. In *Geoscience and Remote Sensing Symposium, 2005. IGARSS'05. Proceedings. 2005 IEEE International* (Vol. 1, p. 4–pp). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1526129

Karegowda, A. G., Manjunath, A. S., & Jayaram, M. A. (2010). Comparative study of attribute selection using gain ratio and correlation based feature selection. *International Journal of Information Technology and Knowledge Management*, *2*(2), 271–277.

Kohavi, R., & others. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* (Vol. 14, pp. 1137–1145). Retrieved from http://frostiebek.free.fr/docs/Machine%20Learning/validation-1.pdf

Larsen, J. (2013). Bike-sharing programs hit the streets in over 500 cities worldwide. *Earth Policy Institute*, *25*. Retrieved from http://www.earth-policy.org/plan_b_updates/2013/update112

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R News*, *2*(3), 18–22.

Mickey, R. M., & Greenland, S. (1989). The impact of confounder selection criteria on effect estimation. *American Journal of Epidemiology*, *129*(1), 125–137.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). *Introduction to linear regression analysis* (Vol. 821). John Wiley & Sons. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=0yR4KUL4VDkC&oi=fnd&pg=PR13&dq=linear+regression+A+residual+is+the+difference+between+the+observed+response+&ots=p4owAjgUCl&sig=MN_PaoJVgiiMTVFYzz2nLAyTCzc

Parkes, S. D., Marsden, G., Shaheen, S. A., & Cohen, A. P. (2013). Understanding the diffusion of public bikesharing systems: evidence from Europe and North America. *Journal of Transport Geography*, *31*, 94–103.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Raviv, T., & Kolka, O. (2013). Optimal inventory management of a bike-sharing station. *IIE Transactions*, *45*(10), 1077–1093.

Rodriguez, J. D., Perez, A., & Lozano, J. A. (2010). Sensitivity Analysis of k-Fold Cross Validation in Prediction Error Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(3), 569–575. http://doi.org/10.1109/TPAMI.2009.187

Rojas, R. (2013). *Neural networks: a systematic introduction*. Springer Science & Business Media. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=4rESBwAAQBAJ&oi=fnd&pg=PA3&dq=neural+network+A+Systematic+Introduction&ots=VzghcS4XoQ&sig=x5dv5InPb08ay5xnNsbhRde3hX4

Schüldt, C., Laptev, I., & Caputo, B. (2004). Recognizing human actions: a local SVM approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on* (Vol. 3, pp. 32–36). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1334462

Seber, G. A., & Lee, A. J. (2012). *Linear regression analysis* (Vol. 936). John Wiley & Sons. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=X2Y6OkXl8ysC&oi=fnd&pg=PR5&dq=linear+regression&ots=sclVA5iShs&sig=RziNE5bZKur_qziVyTupFv7fW2M

Shaheen, S., Cohen, A., & Martin, E. (2013). Public bikesharing in North America: early operator understanding and emerging trends. *Transportation Research Record: Journal of the Transportation Research Board*, (2387), 83–92.

Shaheen, S., Guzman, S., & Zhang, H. (2010). Bikesharing in Europe, the Americas, and Asia: past, present, and future. *Transportation Research Record: Journal of the Transportation Research Board*, (2143), 159–167.

Shaheen, S., Zhang, H., Martin, E., & Guzman, S. (2011). China's Hangzhou public bicycle: understanding early adoption and behavioral response to bikesharing. *Transportation Research Record: Journal of the Transportation Research Board*, (2247), 33–41.

Sohn, S. Y., & Moon, T. H. (2004). Decision tree based on data envelopment analysis for effective technology commercialization. *Expert Systems with Applications*, *26*(2), 279–284.

Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., & Feuston, B. P. (2003). Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of Chemical Information and Computer Sciences*, *43*(6), 1947–1958.

Tranmer, M., & Elliot, M. (2008). Multiple linear regression. *The Cathie Marsh Centre for Census and Survey Research (CCSR)*. Retrieved from http://www.cmist.manchester.ac.uk/medialibrary/archive-publications/working-papers/2008/2008-19-multiple-linear-regression.pdf

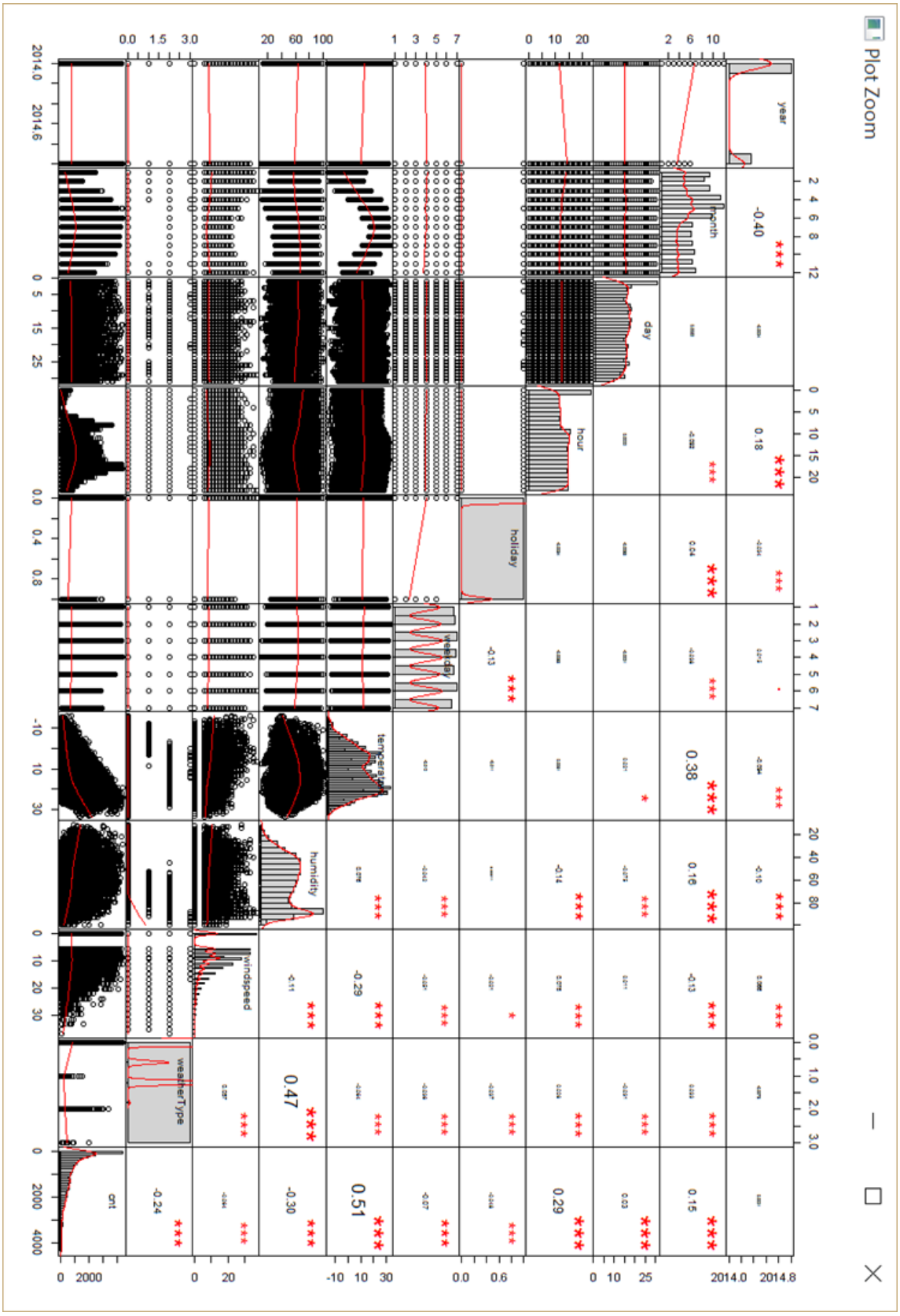
Vogel, P., Greiser, T., & Mattfeld, D. C. (2011). Understanding bike-sharing systems using data mining: Exploring activity patterns. *Procedia-Social and Behavioral Sciences*, *20*, 514–523.

Williams, G. (2011). *Data mining with Rattle and R: the art of excavating data for knowledge discovery*. Springer Science & Business Media. Retrieved from https://books.google.com/books?hl=zh-CN&lr=&id=mDs7OXj03V0C&oi=fnd&pg=PR3&dq=rattle+Rattle+(Williams,+2011)&ots=m_R2qiqX3j&sig=C7BHln8N6UiqkKmsG41t6UKqcQs

Willmott, C. J. (1981). On the validation of models. *Physical Geography*, *2*(2), 184–194.

Yao, B., Zhao, Z., Liu, K., & Cai, A. (2014). Bagging based metric learning for person re-identification. In *Multimedia and Expo (ICME), 2014 IEEE International Conference on* (pp. 1–6). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6890259

APPENDIX A – CORRELATION OF VARIABLES



APPENDIX B – DECISION TREE FOR NEW FEATURES

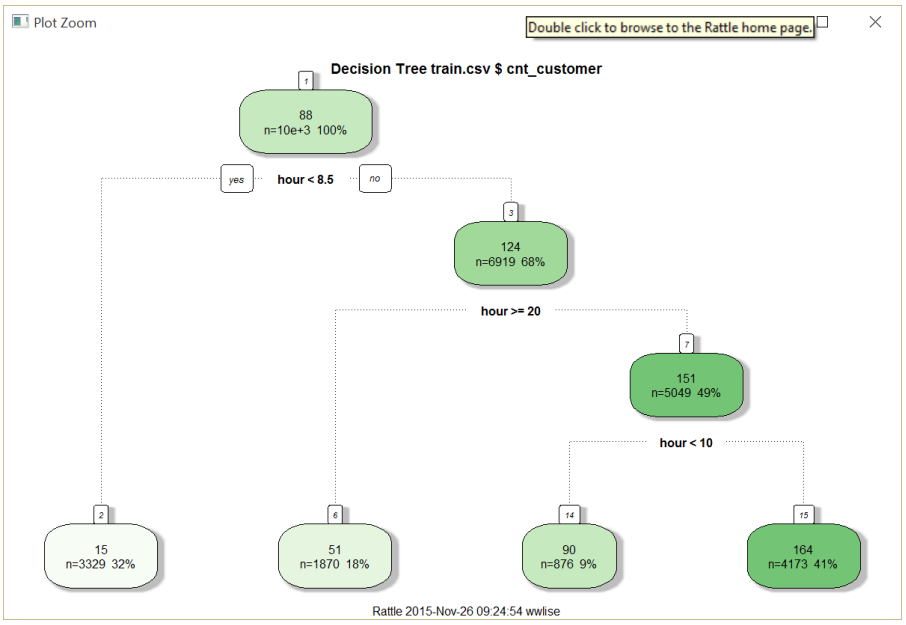


Figure 42 Decision tree: count of short time users by hour

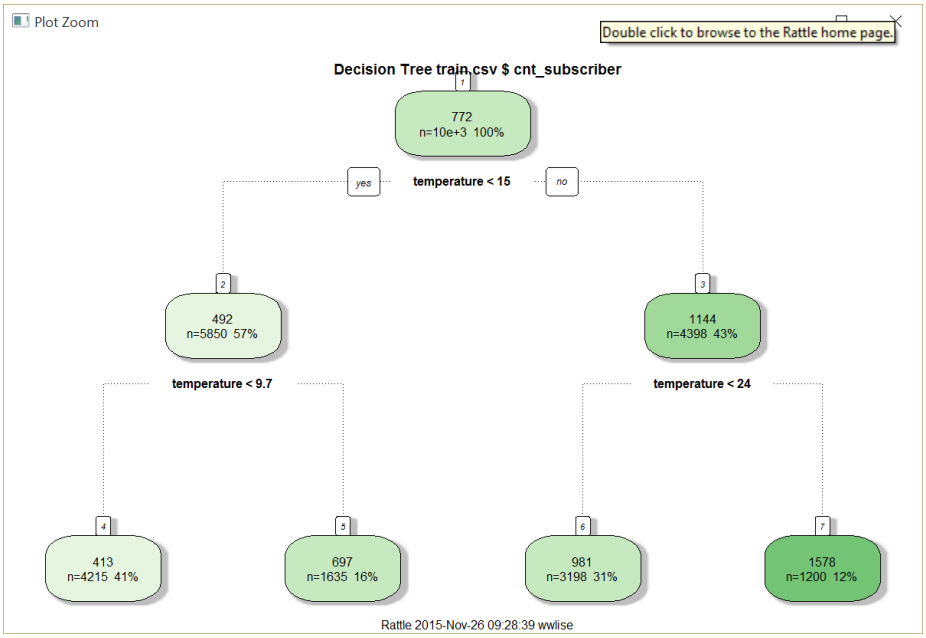


Figure 43 Decision tree: count of long time users by temperature

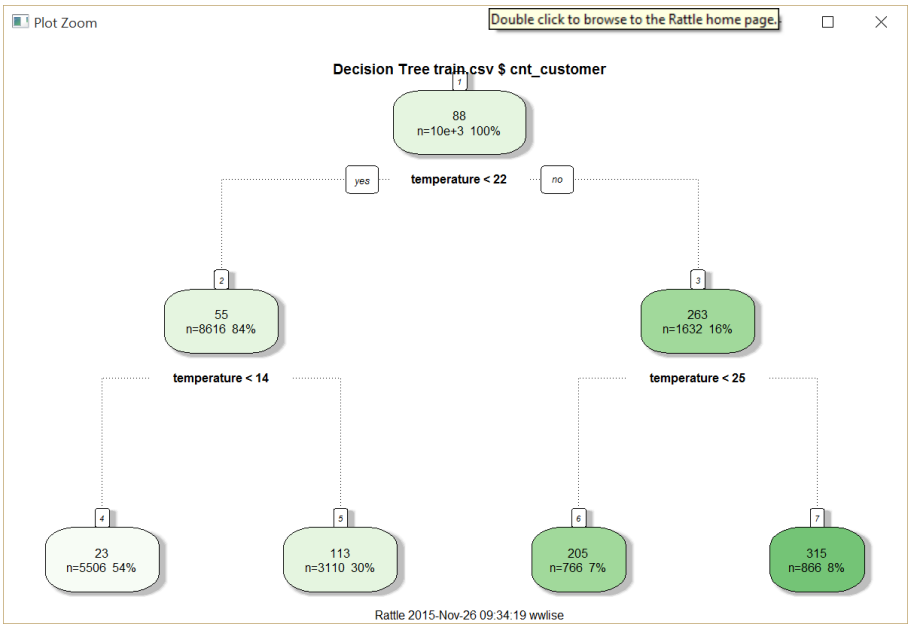


Figure 44 Decision tree: count of short time users by temperature

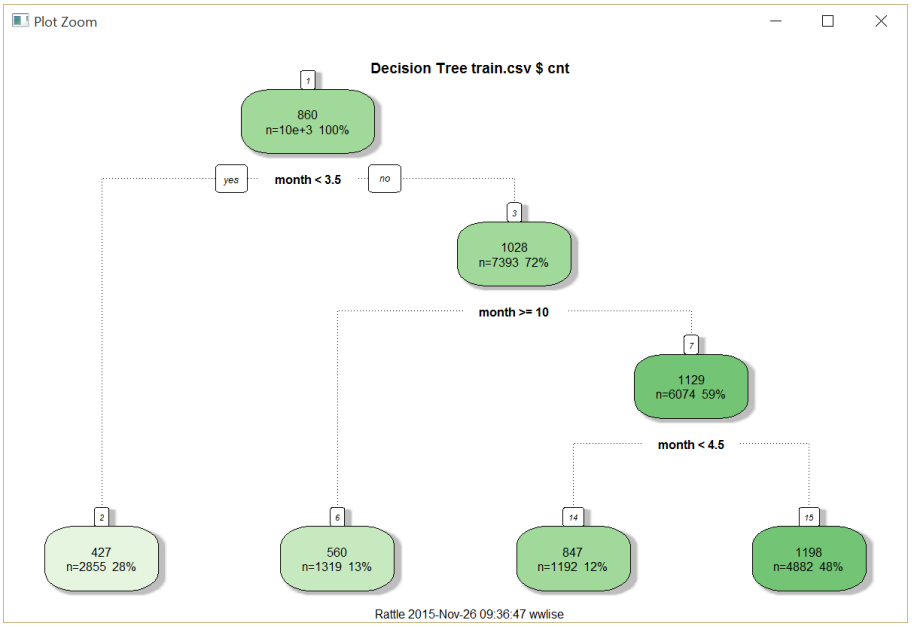


Figure 45 Decision tree: count of all users by year and month

APPENDIX C – RESULTS OF THREE METHODS

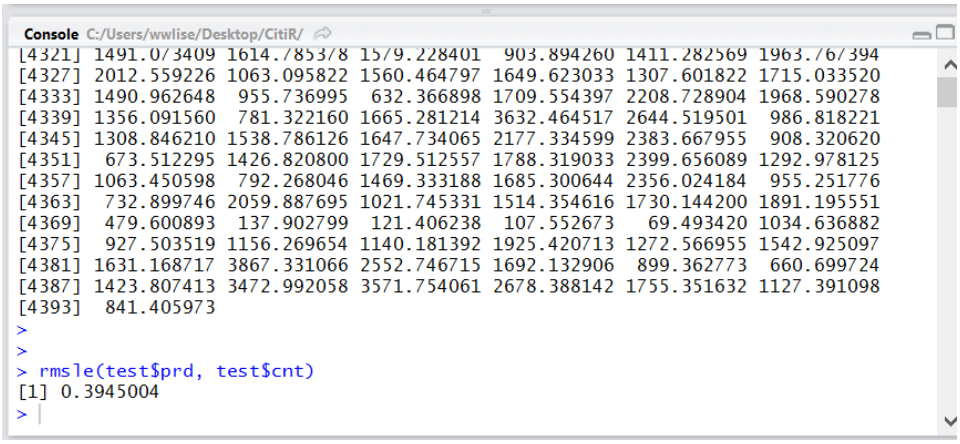


Figure 46 Outcome of method one: Null value replaced by average value

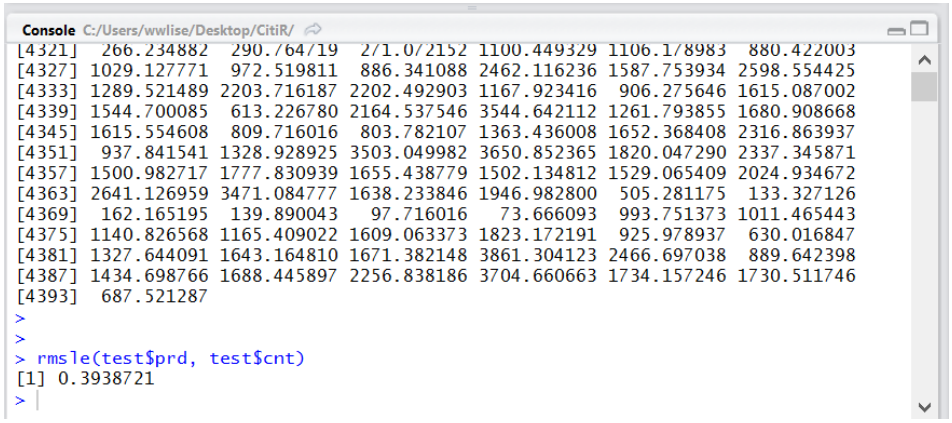


Figure 47 Outcome of method two: Null value replaced by average and 0 value predicted by RF model

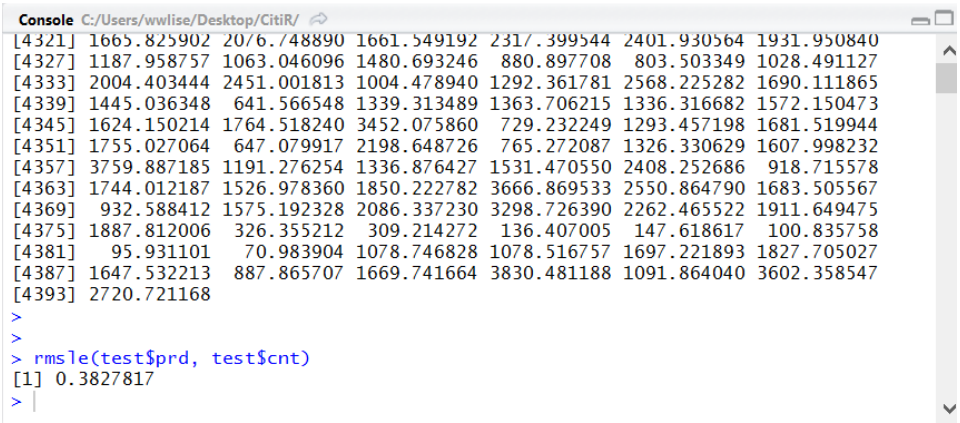
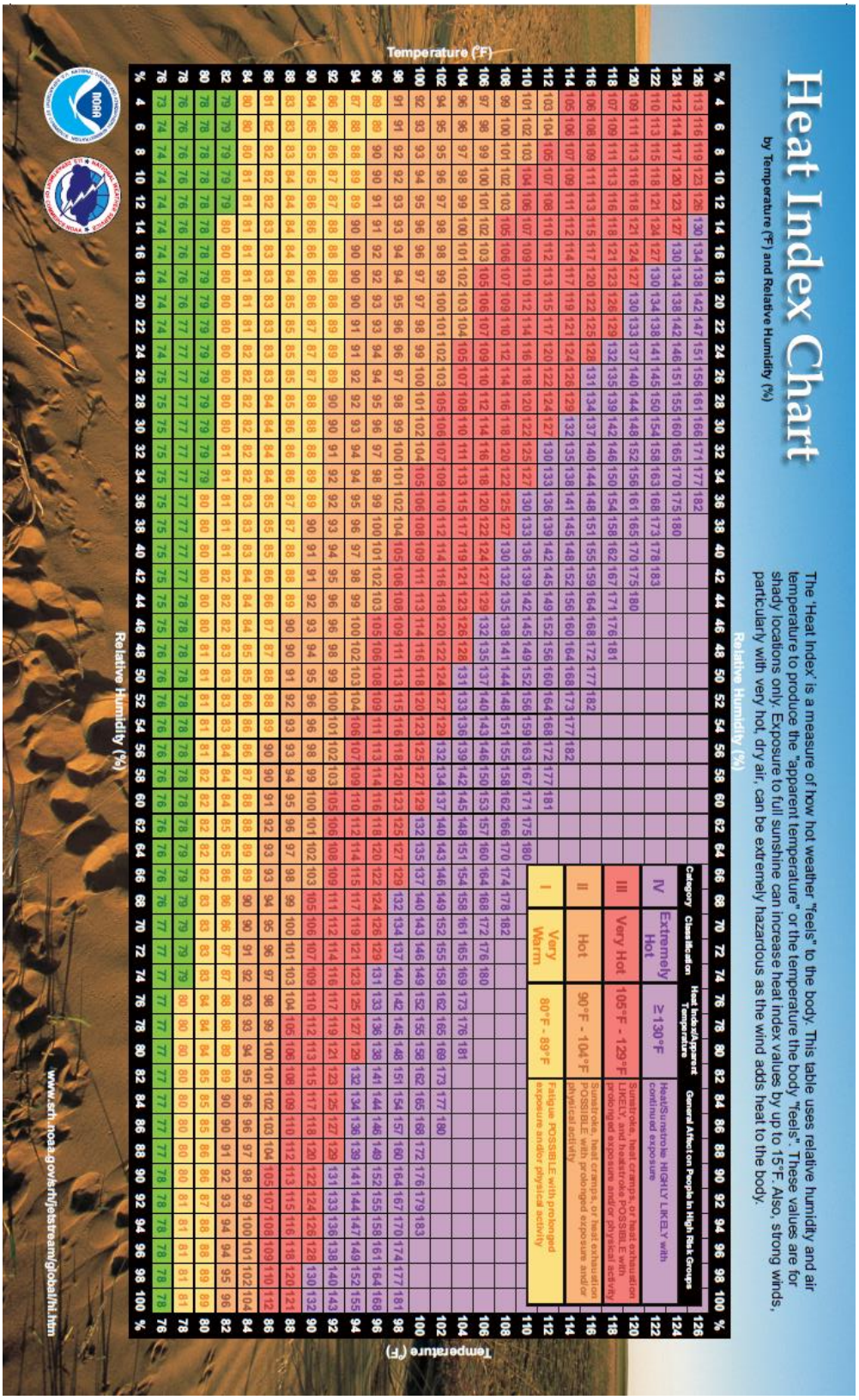


Figure 48 Outcome of method three: Null value predicted by RF model and 0 value predicted by RF model

APPENDIX D – HEAT INDEX CHART[1]



[1] http://www.srh.weather.gov/jetstream/downloads/heatindex_rh_f_20x12.pdf

APPENDIX E – RESULE OF FIVE ENHANCING RESULT BY 70/30 METHOD

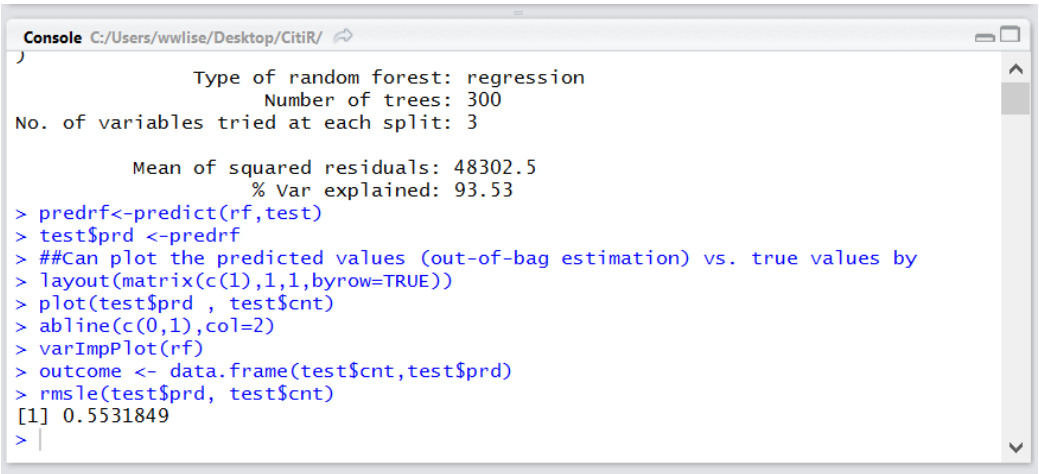


Figure 49 Result of basic RF model by 70/30 method

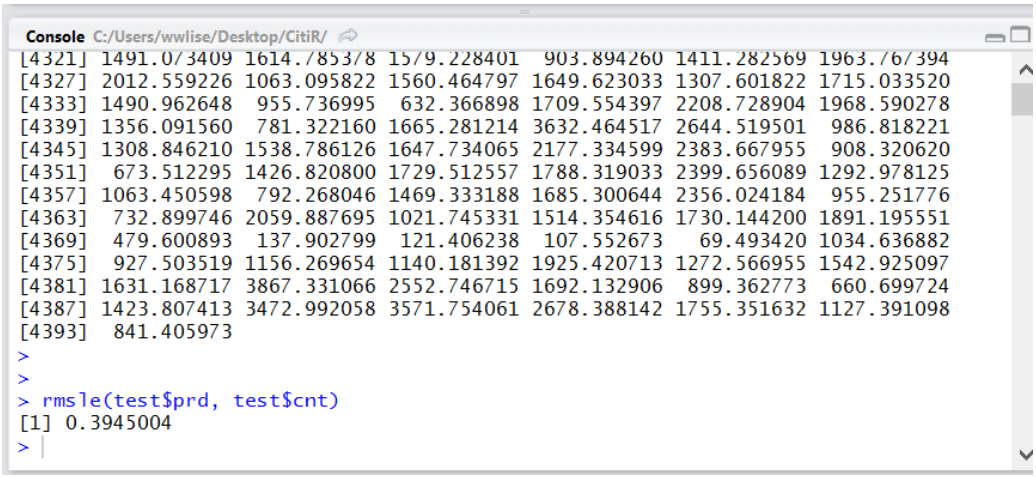


Figure 50 Result of improved RF model by adding new features (70/30 method)

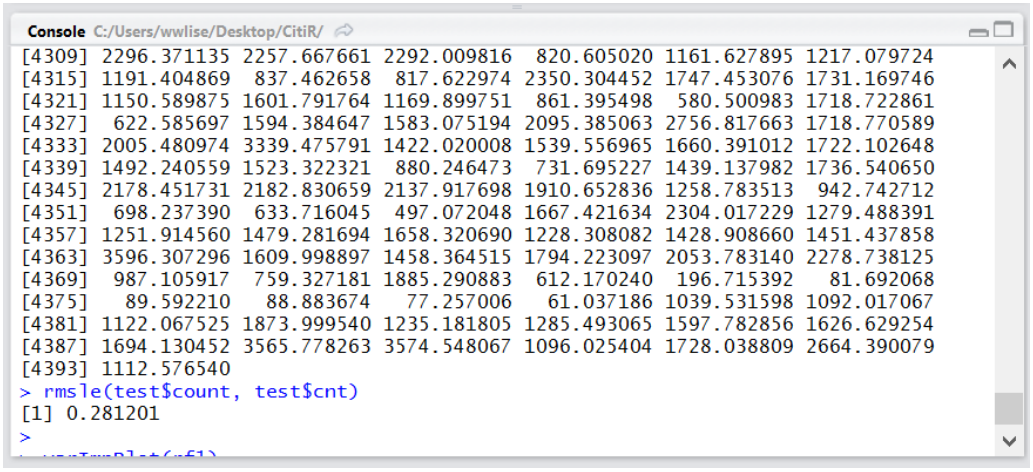


Figure 51 Result of improved RF model by log transformation (70/30 method)

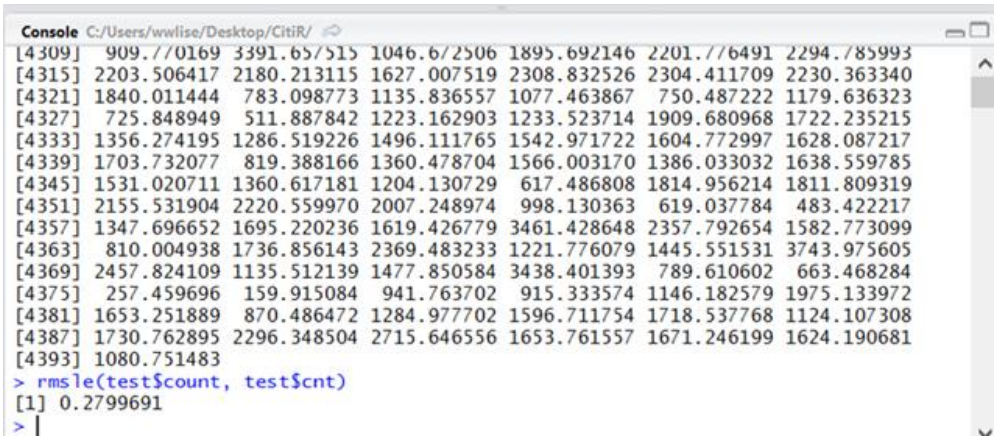


Figure 52 Result of improved RF model by adding heat index (70/30 method)

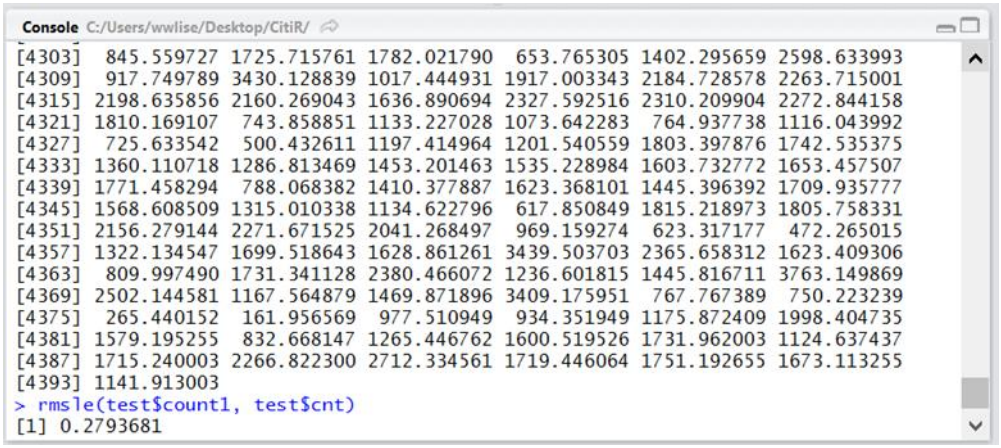


Figure 53 Result of improved RF model by feature selection (70/30 method)