CHALMERS

Destination Prediction with Decision Tree

An Intelligent Driver Assistant for Volvo Cars: Destination Prediction, Traffic Notifying and Route Advising

WEINAN FENG

Department of Computer Science & Engineering
Computer Science - Algorithm, Language and Logic
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2014
Master's Thesis 2014:8

Destination Prediction with Decision Tree
An Intelligent Driver Assistant for Volvo Cars: Destination Prediction, Traffic Notifying and Route Advising

WEINAN FENG

Examiner: CHRISTOS DIMITRAKAKIS

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0) 31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden August 2014

Abstract

In this thesis, a destination prediction method based on decision tree model will be presented. The method will be used to support the Volvo Car Intelligent Driver Assistant System. Accurate destination prediction method is useful in many cases, for instances, it helps to free driver's hand of manually inputting destination address, provide traffic jam report before running into traffic jam and recommend the optimal route with the supporting of third parties' services. The mean accuracy of the destination prediction introduced in this thesis can achieve about 75%. By collaborating with the INRIX Traffic API, WirelessCar API and Volvo On Call app, the Intelligent Driver Assistant System achieves the goal to notifying Volvo Car drivers with traffic information and providing possible routes to the predicted destination.

Acknowledgements

Firstly, thanks to Volvo Car Group Connectivity Department that supported me to study this topic. Secondly, thanks to Anders Adolfsson and Carl Johan Ström who spent much time to offer me useful guidance and many kinds of helps during the study. Thirdly, I must to thank my academic supervisor Christos Dimitrakakis who offered helps to improve the thesis quality.

Furthermore, thanks to Christian Hedegaard and Ingmar Olsson who were kindly offering their ideas to help me come up with the topic. To Timo Minderiein and Ian Whelan who helped me to solve the INRIX API related problems.

At last, thanks to all of my colleagues in Volvo Car Group who offered attentions and helps for the thesis, especially to five participants who provided their driving journal logs for studying and testing.

Weinan Feng, Gothenburg, Sweden 08/21/2014

Contents

1

Introduction

Connected car is a car concept that is getting more and more popular. In December 2012 [1], Volvo Car Group announced that they would make Volvo cars become smarter with help from Ericsson. With the support of Connected Vehicle Cloud by Ericsson, the connected Volvo Cars could access the Internet and be used as like a smart phone (e.g. Install apps specifically for cars). The concept of Connected Car opens the door to the Internet for cars. In 2005, Chris Evans-Pughe has given a vision of future traffic with connected cars in his paper [2]. He envisioned that the connected car-to-car network could make cars communicate with each other in order to improve the road safety. In another paper [3], researchers described a work done under the Traffimatics project, which was sponsored by Next Wave Technology and Markets (NWTM) DTI, to build a real world wireless communications testbed. It could help vehicles communicate with traffic information, road information, weather information and etc. The Connected Car market is growing. More and more vehicle manufacturers have joined the connected vehicle world and tried to make the Connected Car for the future. BMW has shown off its connected car [4] that puts local services and individual driver needs together. The prototype system can provide information on coupons, parking spaces and even the best place to drink coffee. In the Consumer Electronics Show, CES 2014 [5], Volvo launched its new cloud-based infotainment system that changed the way car drivers commute and perceive their time in the car [6]. It allowed drivers to find the nearby parking area and automatically pay for parking, to pre-heat or pre-cool their car remotely and to quickly access Wikipedia.

Even though the current Connected Car provides drivers with many Internet based and useful functionalities, it does not necessarily mean that the Connected Car is Intelligent. The current Connected Car is another platform that is simillar to smartphone, tablet and computers. It only reads the information from Internet and displays them to drivers. It is like an information board. The connection to the Internet creates a window to see the world, but it does not intent to help drivers like an agent who can give suggestion. In this thesis, we are going to introduce an Intelligent Driver Assistant System for Volvo Cars that can predict the destination of the driver's trip depending on the driver's driving journal logs. The system is a kind of driver's assistant or agent that can intuitively guess the destination the driver may go without the driver having to input any information to the system before setting off. To know the destination without the driver's input can make it possible to pre-notify the driver with traffic jam or road incidents information. It can also help the driver to choose an optimal route to the destination and give useful suggestions. The introduced Intelligent Driver Assistant System in this thesis is very similar to well-known Google Now and Siri, which can predict destinations and give useful information such as route to a destination, traffic incidents and Estimated Time of Arrival (ETA).

In the later sections, we will illustrate the overview of the architecture of the Intelligent Driver Assistant System for Volvo Cars. We will mainly introduce the idea of the whole project and the approach we used to predict the driver's destination by making study of the driver's driving journal logs. Using the predicted destination, INRIX Traffic API and WirelessCar API, the system can pre-notify the driver with traffic jam information and alternative routes to the destination. In order to know the performance of the algorithm that will be used to predict destination, we will also test the algorithm with different ratios between the amount of training data and the amount of testing data. The testing result and performance will be discussed at the end of the thesis report. In the section of future work, we will mainly introduce the possible usage depending on the predicted destination and also discuss about how to improve the system in the future.

2

Related Work

INTELLIGENT CAR is generally discussed about automatic control systems in cars [7]. The automatic control system has made a lot of contributions in improving the road safety. It has the advantage of taking action and reacting faster than human. It is more precise that have helped avoid a lot of accidents that could be caused by human mistakes. Volvo has introduced an Intelligent Car system called Pedestrian Safety. The radar and camera on the car can detect the surrounding environment and all moving objects. The system can analyse the receieved data and automatically take action and stop the car if driver does not take any actions in order to avoid hitting a pedestrian. On the other hand, Intelligent Car also relates to driver support or assistance that help to reduce driver's mental and physical load. Car manufacturers are trying to set up a communicating net among cars that can make cars communicate with each other. The traffic net can help individual vehicle know the traffic situation ahead, thus drivers can take actions based on the traffic situation. However, most of car manufacturers are not eager to make fully automatic driving car. They believe people enjoy the fun of driving, thus most of them are seeking for "autopilot" [8] solution, which is a semi-automatic driving system, as like Volvo Cars's self-driving autopilot system [9].

Most of the previous studies about Connected Car and Intelligent Car are concentrated on vehicle. Researchers are trying to enhance the functionality of the vehicles in order to provide a safer and more useful car to driver. The benefits to drivers are cars can automatically help drivers deal with some dangerous situation. However, there is rare study beyond car itself that can provide suggestions and guidance to drivers before they get into trouble. In this thesis, we are going to make an Intelligent Driver Assistant system for Volvo Car drivers that can predict the destination of the trip and send notification to driver before they set off when there is a slow traffic ahead to the destination.

Researchers who published paper [10] thought that the traditional relationship between the car, driver and city was just waypoint navigation with additional traffic information. The car could receive information of waypoint, for examples, points of interest, traffic information, shortest route and etc. They proposed an Affective Intelligent Driving Agent that was multi-goal-centric rather than traditional waypoint-centric system. They envisioned that the system mimicked the expertise of a driving assistant who was familiar with both the driver and the city. Not only focus on determining the routes to the specific waypoint, the system made analysis of driver's behaviour in order to identify the goals for the driver. The agent system included three parts: 1) Destination and trajectory prediction, 2) Semantics analysis of the City and 3) Profiling of the City using GPS data. The system was mainly focusing on trajectory and stop prediction and activity prediction and detection. It used data mining algorithm to predict driver's trajectory and activity in respect to commercial navigation systems. The purpose of the agent system was to give suggestions to drivers such as giving suggestion of the next possible destination to an art museum or a petrol station and etc. For the agent system, the first task that needs be complete was destination prediction. The destination prediction was the most important part of the system because it was the base for tasks 2) and 3).

In the previous studies, researchers had made efforts on the area of destination prediction and behaviour study. These studies about destination prediction became popular since GPS device equipped smartphone had occurred. People's activities got more and more related to geographic location and some of them like to expose the location to public social media (E.g. Facebook, Twitter and etc.). In the paper [11], researchers from Japan tried to predict the smartphone users' destination by studying their non-periodic position logs. Studying the historical logs data and setting up a spatiotemporal cluster model, their solution had achieved higher than 65% accuracy of destination prediction.

Besides the study above, most of the destination prediction studies are driver oriented and based on the driver's driving history such as in paper [12]. To know the destination of a driver's destination is necessary for the purpose of delivering useful information. For instance, drivers want to know the traffic situation in front of them and also want to know services (e.g. petrol station, parking lot, restaurant and etc.) near their destination. By knowing the destination, there is no need to distract driver's attention away from the road on inputting destination address. We can still provide this kind of useful information for drivers without driver's manually inputting if the car can predict the destination. Relevant studies have been made in papers [13, 14]. In both papers, they studied about the real time destination prediction based on historical GPS logs. The GPS logs that were used in paper [13] were from each individual driver. The researchers built a polygon strip for each trip logs, and then they predicted the trip by matching the driver's current GPS location with those strips. In paper [14], Johan Krumm did not use the GPS logs from a single driver. Instead he used logs from 118 driving volunteers. He proposed an intuitive predictor to predict the destination based the most efficient route. He believed that drivers would prefer the most efficient route to the destination rather than the one cost more time. The paper [15] made the prediction similar to what had been done in paper [13] but also added map matching to increase the accuracy.

By observing drivers' driving behaviour, it shows that car navigation system is useful but drivers rarely use it. There are several reasons, for examples, inputting a destination to the system takes too much time and it will distract the driver's attention during driving. Another group researchers in paper [16] built a new navigation system that could automatically predict the destination by using probability model. This new navigation system could provide useful traffic information for drivers in real-time without distracting driver on inputting.

Similar to the predicting method that had been used in paper [16], papers [17, 18] also used Bayesian inference to predict the destination. The differences were that they split trips into small pieces, which were called sub-trajectories. The purpose of their studies was to push advertisements to the driver if there was a possibility that the driver would drive into specific areas.

Besides the methods that had been used in the papers above, two main approaches also been studied: Hidden Markov Model that was used in papers [19, 20] and Decision Tree based classification in papers [17, 18, 21, 22, 23]. Both of those two approaches had made achieved high accuracy of prediction. By using Hidden Markov Model, researches in paper [19] achieved approximately 98% accuracy in prediction. On the other side, Christian and Raja showed the result of prediction with 96% mean accuracy by using Decision Tree model in the paper [24], which was published in recent year 2013.

In this thesis, we will mainly use the Decision Tree model to predict the destination based on the historical driving journal logs from 5 Volvo Car drivers.

3

Design and Implementation

3.1 Problem Description

It often happens that a driver drives into unpredictable traffic jam. Getting stuck in traffic jam is a annoyance to drivers especially in big cities such as London, New York, Beijing, Shanghai and etc. It wastes time and also makes more air pollution. We imagine: if we could predict the destination of the driver, the car system could automatically check the current traffic situation for drivers and send traffic flow speed of route to drivers, then drivers could make actions such as delay setting off time or escape choosing the slow route in order to save the time on the road. The aim of this research is to predict the destination of trip based on the Volvo Car driver's driving journal. Using the predicted destination together with INRIX Traffic API, WirelessCar API and Volvo On Call app, a Volvo Car Driver Agent (VCDA), which is running in the Volvo Car Cloud Server, can check the current traffic information of the route between current car position and predicted destination in order to provide useful traffic information to drivers. The VCDA will push notification to Volvo Car driver's Volvo On Call app with current traffic information when the traffic flow of the routes to the predicted destination is slow.

3.2 Resources

3.2.1 Volvo On Call

Volvo On Call [25] is a telematics system that assists Volvo Car customers with immediate access to 24-hour emergency and roadside assistance. Volvo On Call is designed to provide Volvo Car owners with services of safety, security and convenience. By using Volvo On Call, a Volvo Car driver can easily get in touch with a Volvo On Call operator who knows the precise location and can dispatch the help quickly.

Volvo Cars already has a mobile application Volvo On Call (VOC) [26] in the market. Volvo Car drivers can access their car by using VOC app. By using VOC app, Volvo Car drivers can access and control their Volvo cars as follows [27]:

1. Check on car
   - Check information on temperature at car location.
   - Check status of door and window locks.
   - Check remaining fuel, battery, washer fluid and etc.
   - Access dashboard
   - Access climate control system
   - Get alerts if something needs to be fixed
2. Find car
   - Find location of the car
   - Show the car on the map
   - Provide directions to the car
3. Lock car
   - Remotely lock and unlock the car
   - Remotely open and close the windows
4. Driving Journal
   - The journal includes data of start and end position of the car of each trip
   - Distance of each trip
   - Duration and fuel consumption during 40-day period
   - The journal can be exported as spreadsheet
5. Start parking heater
   - Remotely turn on the parking heater if the car has the feature
6. Start the engine to cool or heat car
   - Remotely start the engine in order to get comfortable cabin climate when the driver gets into the car

3.2.2 WirelessCar Service

WirelessCar [28], which is owned by the Volvo Group, is an automotive telematics service provider (TSP) provides customized telematics services to various car manufacturers and commercial vehicles. It provides VOC app with backend supports such as Emergency call, Breakdown call, Stolen vehicle tracking, Send to Car and Remote Services. By using WirelessCar Restful API, VOC app can retrieve Volvo car's location and driver's driving journal, turn on/off the Volvo Car engine, turn on/off Volvo Car heater, lock/unlock Volvo Car door, check status information of Volvo Car such as fuel status, battery status, dashboard, and etc.

VOC app has to send a URL constructed by function description to WirelessCar server in order to control Volvo Car or check status of Volvo Car. The WirelessCar server will also send a response with JSON format that may include requested data or error message.

3.2.3 INRIX Traffic Service

INRIX [29] is a traffic service provider that is dedicated to helping people by offering best-in-class traffic information. It offers unparalleled knowledge of what's happening on the road by collecting and analyzing the high quality traffic information from sources around the world. As a result, INRIX users can get accurate real-time, historical and predictive traffic services on various kinds of roads and streets.

INRIX Traffic API offers a large number of functions in order to provide users with useful traffic information. By sending a URL request to INRIX server with specified function information, the server will send back a response in XML format. Based on the function that specified in the URL, the INRIX server can return traffic information such as alternative routes, average speed of each route, real time traffic speed map and etc.

3.3 Approach

In previous studies about destination prediction, several different approaches have been introduced and we have discussed in the related work section. For examples, papers [17, 18, 21, 24] achieved the destination prediction by using decision tree based classifier and papers [19, 20] achieved it by using Hidden Markov Model.

In this thesis, Decision Tree will be used to predict the destination of Volvo Car drivers' trip. One reason to choose Decision Tree is depending on features of the Volvo Car driver's driving journal log. The driving journal logs contains the start and end positions of trips, it is easy to use decision tree classify them. Another reason is that the system we will make should make pre-prediction, which means the system should predict the destination before the driver starts a trip. Using Markov Chain similar methods are good for real-time prediction when the car can send a sequence of positions. Thus we choose the decision tree as the main approach for this thesis.

3.3.1 Overview

The architecture of the Intelligent Drivers Assistant System in this thesis is shown in Figure 3.1. The whole system consists with four main components: Volvo Car AI Cloud (supported by WirelessCar API), Volvo On Call app, Volvo Car and INRIX Traffic Server. The AI Cloud makes prediction and sends notification to Volvo On Call user.

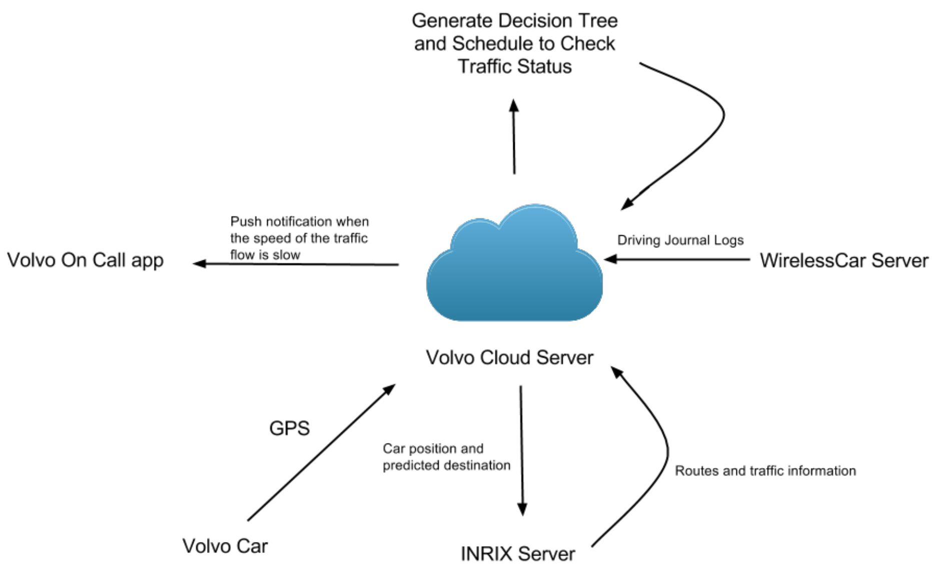


Figure 3.1: Architecture of Intelligent Driver Assistant System

Volvo Cloud Server is the brain of the whole system. It manages the communications between each component and makes prediction. The WirelessCar API helps Volvo Cloud Server get Volvo Car driver's driving trip logs and it makes use of the data and generates a decision tree and a schedule to check traffic status for each driver. It asks the location of the driver's Volvo Car based on the generated schedule. When the location is received, the Volvo Cloud Server will predict the destination depending on the received location data and generated decision tree. Then the car's location and the predicted destination will be sent to INRIX server. The INRIX server calculates possible routes between the car's location and the predicted destination and then sends the routes with average traffic speed back to Volvo Cloud Server. At last, the Volvo Cloud Server will compare the route speed with the threshold that has already been set by the driver. Only at the case that the route speed is lower than the threshold, the Volvo Cloud Server pushes a notification to driver (via Volvo On Call app). Otherwise, the Volvo Cloud Server will just drop this message and wait until the next time coming in the schedule. The

INRIX server can return at most 3 alternative routes. If there exists one route that the average speed is higher than the threshold, while other routes' speed are lower than the threshold, the route that the average speed is higher than the threshold will be decided to be pushed to the driver as a suggestion by the Volvo Cloud Server, in case of the driver would choose other slow routes and waste time on them.

3.3.2 Decision Tree

Decision Tree [30, 31] is a classification algorithm popularly used in data mining and machine learning. It is a tree-like graph or model that is used to make decisions and induct possible consequences [32]. A decision tree consists of two different elements, internal decision nodes and terminal goal leaves. At each decision node, there is a function to generate the child branches with outcome labels. The process begins at the root and recurs until a leaf node is found. The leaf node is the goal of each decision path. Figure 3.2 visualizes an example of a decision tree. The nodes with white text have feature information about day of week and time. The leaves with black text are the goals for each path. For example, people go home at 16:00 o'clck on Friday.

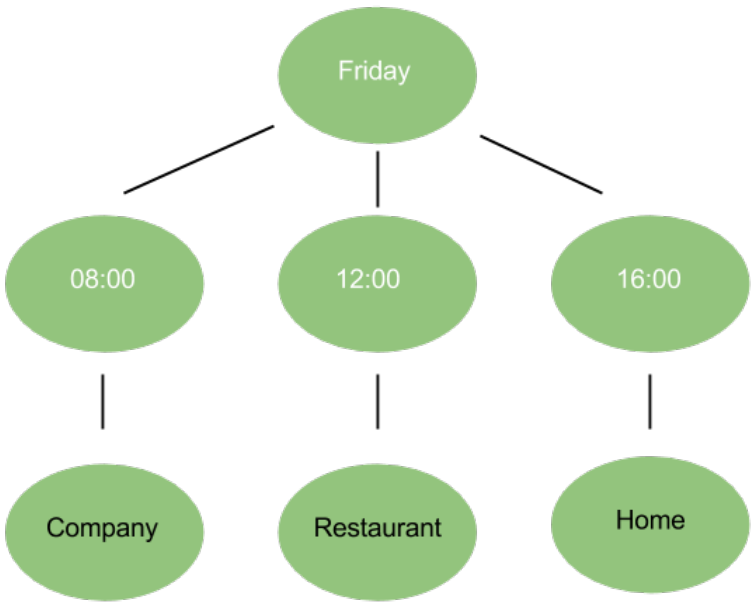


Figure 3.2: A decision tree example

An accurate approach of destination prediction has been introduced in paper [24] by using Decision Tree. Christian and Raja [24] asked 10 subjects to record GPS trace data of their travelling by using smartphone in the San Francisco Bay Area for 2-9 weeks. They used the current position of a driver, the position 5 minutes ago, the time of day and the day of week as input of the Decision Tree model to output the predicted destination. Decision Tree has several advantages [33], for example, it is easy to understand and to interpret. A Decision Tree can be easily visualized by drawing a tree-like graph. The cost of using the tree is logarithmic in the number of training data. It is possible to validate the model by using statistic tests.

3.3.3 ID3

ID3 algorithm [34] is designed for the case that there are many attributes, which can be *Start Time*, *Start Address* and *Day of Week* in this thesis, as input and the training set contains many objects. It can generate a reasonable good decision tree, which is a well-constructed simple tree, without much computation.

ID3 algorithm has an iterative data structure. A subset of training data will be chosen to form a sub-decision tree at each iterative process. The subset is named window in ID3. Data in the window will be appropriately classified by using this tree. This tree is also used for classifying all other training data. If the entire training set is classified, then the process terminates. If not, an iteration of new selection of subset data will be executed. The tree will be expanded and the classifying continues. For arbitrary collection of objects, there are several cases we have to consider. For instance, training data collection is empty, all training data belong to one same class. If the collection is empty, the tree should terminate by returning a default class. If all training data belong to one same class, the tree returns this class. There is also another case that the training set has objects that atrributes' values are all the same as each other but the objects belong to different classes. In this case, we need an evaluating function to evaluate each object and give an evaluation for each object in order to decide which class should be chosen.

Selection of attribute for the root is important in order to get a simple decision tree at last. ID3 usually uses the entropy theory, which measures the amount of disorder in a data set, to select the most suitable attribute for the next node of the decision tree. The hidden idea to use entropy is to find the attribute that has the lowest entropy value for the data set, thus it reduces the amount of information that needs to describe the whole data set.

The paper [34] introduced the information-based method that relays on two assumptions. For all training data set T that contains ci objects of class Ci, it follows the rules (assume that we have two classes C1 = P and C2 = N, there are p objects of class P and n objects of class N):

1. In the decision tree for training set T, an object will be classified to Ci with probability is:

$$Prob(P) = \frac{p}{p+n}$$

$$Prob(N) = \frac{n}{p+n}$$

2. The decision tree will classify an object to a class. The information-based method uses logarithm function to represent the information for classes. The expected information for all classes Ci needed to generate is:

$$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

In order to find the best attribute for the training set T, we have to calculate the expected information required for the tree with each attribute (A). For instance, if the root node attribute A which has values $\{a_1, a_2, ..., a_k\}$, sub-branches will be generated by values $\{a_1, a_2, ..., a_k\}$. Each sub-branch patition the dataset T into $\{t_1, t_2, ..., t_k\}$ that t_k will contains all objects having value a_k.

$$E(A) = \sum_{i=1}^{k}\frac{p_i+n_i}{p+n}I(p_i,n_i)$$

The E (A) is the proportion of the objects in training set T that belong to subset Ti (Ti is the subset of the T that involves all objects have attribute Ai).

Finally the gain information of attribute A can be calculated by

$$gain(A) = I(p,n) - E(A)$$

ID3 will choose the attribute that has maximum gain (A).

For instance, we have data set that is shown in Figure 3.3. The attributes of the data set are ***Start Address***, ***Start Hour*** and ***Day of Week***. The data set also has two classes ***Address P*** and ***Address N***. Each row stands for one individual trip log. There are 9 logs in total in the data set. The goal is to create a decision tree that can classify trips into Address P and Address N depending on attributes using information theory.

The first step is calculating the information of the data set that includes classes Address P and Address N. We have 6 objects of class Address P and 3 objects of class Address N. Then we calculate the information by formula:

$$I(p,n) = -\frac{6}{6+3}\log_2\frac{6}{6+3} - \frac{3}{6+3}\log_2\frac{3}{6+3} = 0.92$$

Start Address	Start Hour	Day of Week	Destination
Address 1	8	Mon	Address P
Address 2	9	Mon	Address P
Address 3	11	Wed	Address N
Address 1	13	Thu	Address P
Address 2	8	Tue	Address N
Address 3	8	Fri	Address P
Address 1	18	Fri	Address P
Address 2	17	Wed	Address P
Address 3	18	Tue	Address N

Figure 3.3: Sample data set

The result means that it needs 0.92 bits in order to represent the information of the data set that includes classes Address P and Address N.

Next step, we have to calculate the information or entropy for each attribute. We start from the attribute Start Address. Attribute Start Address has 3 different values {Address 1, Address 2, Address 3}. For value Address1, there are 3 objects in data set having value Address 1, Three of them from class Address P and zero from class Address N. So

$$p_{address\ 1} = 3 \qquad n_{address\ 1} = 0 \qquad I(p_{address\ 1}, n_{address\ 1}) = 0$$

and similarly

$$p_{address\ 2} = 2 \qquad n_{address\ 2} = 1 \qquad I(p_{address\ 2}, n_{address\ 2}) = 0.92$$

$$p_{address\ 3} = 1 \qquad n_{address\ 3} = 2 \qquad I(p_{address\ 3}, n_{address\ 3}) = 0.92$$

Then the expected information for attribute Start Address is

$$\begin{aligned} E(Start\ Address) = {} & \frac{3}{9} I(p_{address\ 1}, n_{address\ 1}) \\ & + \frac{3}{9} I(p_{address\ 2}, n_{address\ 2}) + \frac{3}{9} I(p_{address\ 3}, n_{address\ 3}) = 0.61 \end{aligned}$$

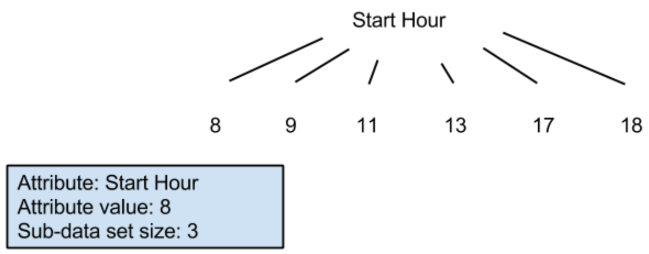
The gain information of attribute Start Address is

$$gain(Start\ Address) = 0.92 - 0.61 = 0.31$$

which means that representing attribute Start Address needs 0.31 bits.
We can implement the same process to other attributes. Then we have

$$gain(Start\ Hour) = 0.61$$
$$gain(Day\ of\ Week) = -0.08$$

so the the attribute Start Hour will be chosen as the root node as it has the maximum gain information. A part of the tree is visualized in Figure 3.4. The attribute Start Hour has been selected as the root node. The tree classifies the training data set into six sub-branches and finishes the first round of classifcation. In the next round, the tree will classify the sub-data set for each value. For example, the sub-data set for value 8 has three objects as shown in Figure 3.4. The algorithm will calculate information of attributes Day of Week and Start Address based on those 3 objects. The calculation process is as same as the process we used to find Start Hour. The algorithm will apply the same process to other values. The classification will not stop until all objects are classified.



Address 1	8	Mon	Address P
Address 2	8	Tue	Address N
Address 3	8	Fri	Address P

Figure 3.4: Tree with root node of Start Hour

3.3.4 Driving Journal Logs

The data that are used for testing are gained from 5 participants. They are colleagues who are working at Volvo Cars. All of them drive Volvo car and use Volvo On Call app. Each time when they put the key in and out, the car will send the location (GPS coordinates) to Volvo Cloud Server and generate one trip data.

The data is shown in the table of Figure 3.5. Five participants offered 3063 individual trips in total and 2381 of them are valid that can be used to create decision tree.

Some reasons may generate invalid trip: If the car were parked in the garage, communication between satellites and the car would not be set up, then the trip would not have a valid GPS location data. Then we treat this kind of trips as invalid. Similar situations like cloudy and rainy days could also cause the same problem. The invalid trips will not be used for creating the decision tree and later testing. There are 3063 trips in total and 2381 of them are valid. Start Positions and End Positions show the amount of different start positions and end positions for each driver. The differences between the numbers of Start Position and the number of End Positions were caused by the invalid trips.

	Number of Trips	Start Positions	End Positions	Valid Trips
Driver 1	777	61	59	565
Driver 2	658	67	66	518
Driver 3	1000	68	64	771
Driver 4	324	44	45	277
Driver 5	304	35	35	250
Total	3063	275	269	2381

Figure 3.5: Trips from driving journal logs

Figure 3.6, 3.7, 3.8, 3.9 and 3.10 show destination distribution for those 5 drivers. It is obvious that there are several mainly visited destinations for each driver. The destination having the highest frequency is most likely the home address of the driver. Even though each driver has travelled several hundreds trips (Driver 3 even has travelled 1000 trips), the mainly visited addresses are only several.

3.3.5 Destination Prediction

In this thesis, a Decision Tree based approach will be presented to predict the destination for Volvo Car drivers. The data that will be used for training the decision tree model is Volvo Car driver's driving journal logs, which can be fetched via WirelessCar API.

WirelessCar Restful API supports the Volvo On Call app and Volvo Cloud Server at the backend. By sending HTTP request that includes the driver's Volvo Car username and password, we can fetch Volvo Car driver's driving journal logs from WirelessCar Server. The driving journal will be used to generate the Decision Tree in order to

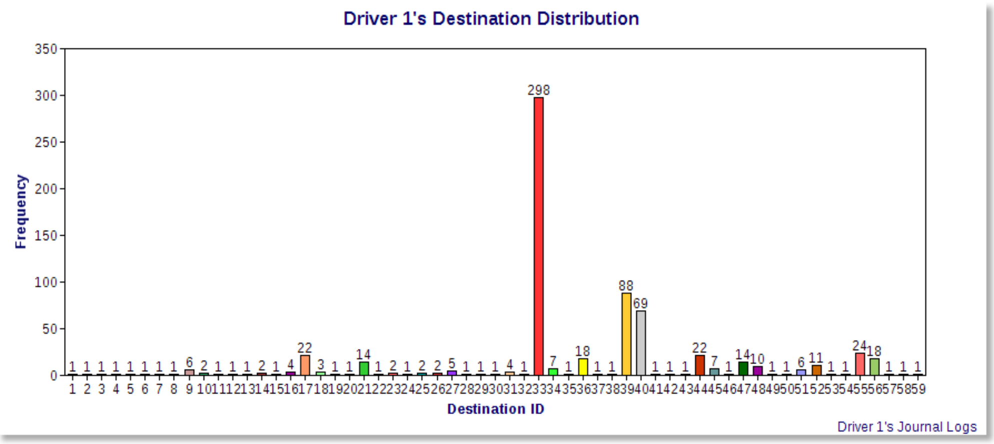


Figure 3.6: Destination distribution for Driver 1

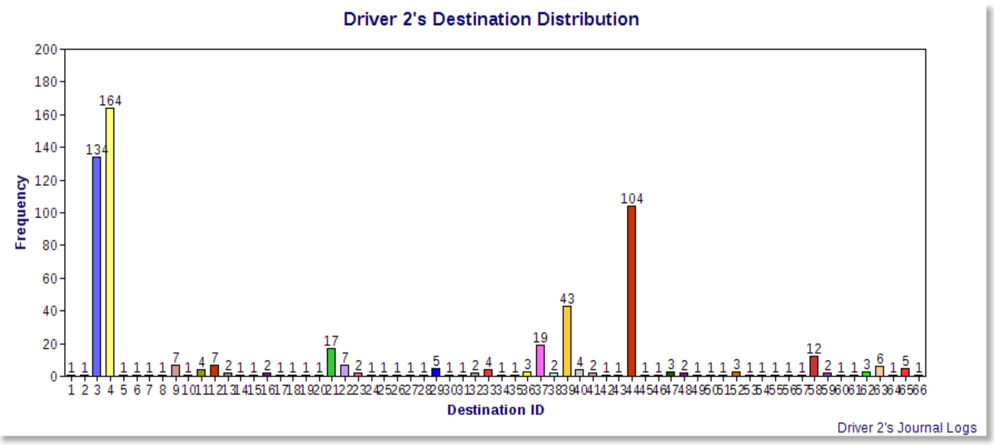


Figure 3.7: Destination distribution for Driver 2

make a private decision tree model for each driver. Figure 3.11 is an example of Volvo Car driver's trip log that is returned by WirelessCar Restful API. The returned trip data includes information of category, userNotes, id, name and the most useful data tripDetails.

The tripDetails data provides start position and end position of the trip. Inside startPosition and endPosition structure, data of GPS location, street address and other details about the position can be found. The tripDetails data also provides the start and end timestamps of the trip. From the observation of general driver's driving behavior, there are regularities between destination and some of data in journal logs. For instance, driving to company mostly happened about 9 am while driving home mostly happened around 5 pm. Day of week from Monday to Friday also has regularity of commute. Destination usually is home when the current position is at company in the afternoon in weekdays. Thus the data within the tripDetails are useful inputting attributes for the

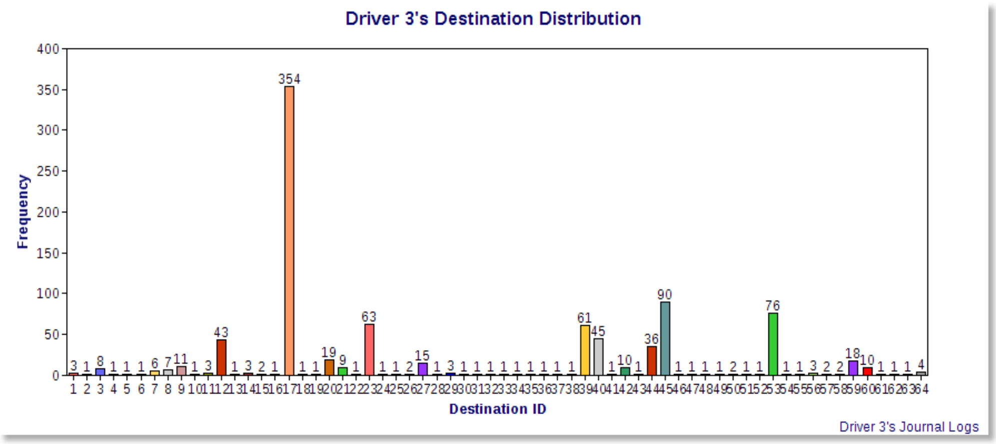


Figure 3.8: Destination distribution for Driver 3

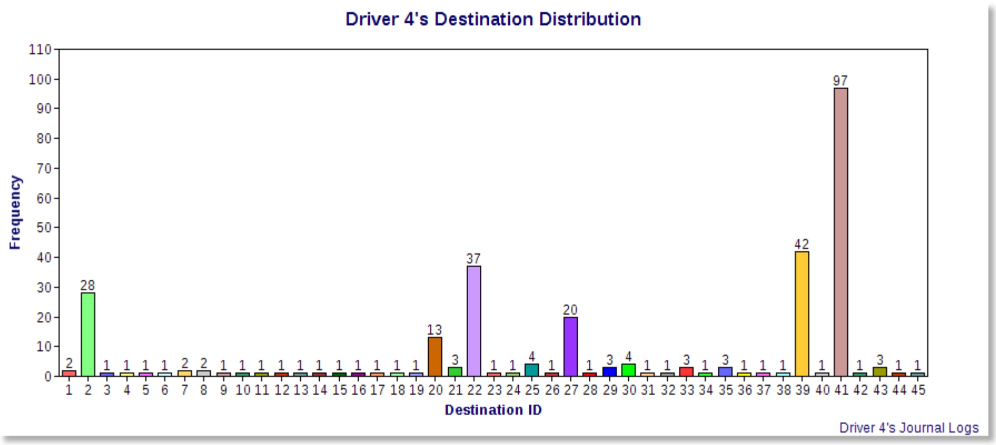


Figure 3.9: Destination distribution for Driver 4

Decision Tree model of destination prediction.

The Volvo Cloud Server takes the responsibility of generating the decision tree for each driver depending on each driver's journal logs.

The decision tree for destination prediction in this thesis has 3 attribute properties: tripStartPosition, tripStartHour and tripDayOfWeek. The potential predicted class is tripDestination. After the Volvo Cloud Server gets the trips of driver, the trips have to be verified before creating decision tree. Since the communication between satellites and Volvo Car can be interrupted by cloud, garage roof or other factors, the GPS coordinates of the car sometimes cannot be uploaded. Thus some invalid trips do not have data of either startPosition or endPosition. The invalid trips have to be eliminated before generating decision tree. Since most of the data in trip structure is useless, we need to extract useful data from original trip data. The data that the decision tree needs are startTime stamp, latitude and longitude of startPosition, streetAddress of startPosition,

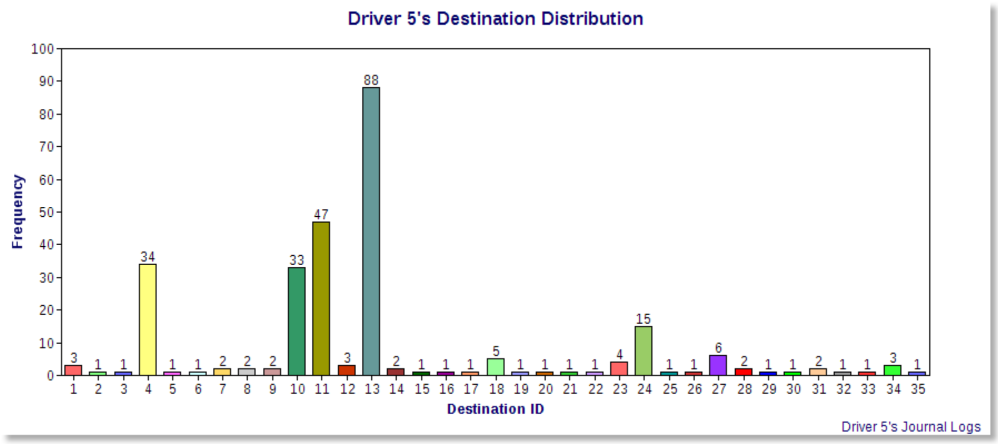


Figure 3.10: Destination distribution for Driver 5

latitude and longitude of endPosition and streetAddress of endPosition. The startTime is used to get start hour and day of week of trip. The data structure of the refined trip includes latitude and longitude of startPosition, streetAddress of startPosition, latitude and longitude of endPosition and streetAddress of endPosition, day of week and start hour.

After the refined trips are ready, a recursive function will be used to create the decision tree by making use of the valid trips data. The function that creates the decision tree takes 4 inputs: training data set (refined trips), attributes (streetAddress of startPosition, startHour and dayOfWeek), class of destination (streetAddress of endPosition) and gain function. The gain function uses ID3 algorithm to get entropy information of each attribute. The classes are the street addresses of endPostion. The output of the function is a well-classified decision tree. The pseudo code of the function to create decision tree is as Pseudo code in Listing 1.

The algorithm is simple. It is just a recursive function recursively generate decision tree branches. The gain_func follows the information theory in ID3. It helps to choose the best attribute for creating node.

Using streetAddress of startPosition and EndPosition faces a problem: drivers do not park their cars at the exact same location every day. This fact produces noise during the process of creating decision tree. In order to improve the accuracy of prediction, we have to remove the noise from training data. In paper [24], researchers used United States National Grid (USNG) [35] system to label GPS locations that close each other. Since the USNG only can be applied to the United States, it does not help for the case in this thesis. Instead, we will use Swedish National Grid (Swedish Rikets Nät, RT 90) [36] to remove the noise from training data. RT 90 is a 2D reference system that maps the region of Sweden by using the Transverse Mercator projection (More information of RT 90 can be found in [36]). Instead of using streeAddress of startPosition and streetAddress of endPosition as inputs, we use RT 90 labeled ids to represent areas that cover those addresses or in another word, the GPS coordinates. We specify the accuracy within 1000

```
{
    "category": "unassigned",
    "userNotes": null,
    "tripDetails": [
        {
            "distance": 34617,
            "endOdometer": 7265307,
            "fuelConsumption": 190,
            "startPosition": {
                "city": "Gothenburg",
                "ISO2CountryCode": "SE",
                "Region": "Västra Götaland County",
                "longitude": [redacted]59570312,
                "postalCode": "4[redacted]8",
                "streetAddress": "G[redacted] 22",
                "latitude": [redacted]66845703125
            },
            "electricalRegeneration": null,
            "startOdometer": 7230690,
            "electricalConsumption": null,
            "startTime": "2014-06-09T15:57:22+0000",
            "endTime": "2014-06-09T16:39:00+0000",
            "endPosition": {
                "city": "Lerum",
                "ISO2CountryCode": "SE",
                "Region": "Västra Götaland County",
                "longitude": [redacted]46582031,
                "postalCode": "4[redacted]8",
                "streetAddress": "T[redacted] 17",
                "latitude": [redacted]624902344
            }
        }
    ],
    "id": 91864990,
    "name": null
}
```


Figure 3.11: Volvo Car driver's trip log example

meters, which means the distance between two GPS locations within 1000 meters, will be labeled as one same RT 90 id.

One of the generated decision trees, which choose start location as root node, can be visualized like in Figure 3.12. The decision tree in Figure 4 starts with the start location that is RT90 ids for the trip start GPS coordinates. The second level is the attribute startHour (Time) and the third level is attribute dayOfWeek. After those three iterations, a destination (RT90 id) always can be specified.

In the first if-case of the create_decision_tree function, either dataset or attribute list is empty the function will return the most likely destination. In order to find the most likely destination of the rest attribute, we need a majority_book to record frequency of each destination for each attribute. This majority_book will be used when the objects in the dataset have equivalent gain information (For example, all trips have the same start location, time and day of week.). In this case, we have to select the destination with the highest frequency of the current attribute if the current attribute is Time and the value is 8 am, the destination with the highest frequency probably is the location of company.

Besides the decision tree and majority_book, we also need a timetable that will be used to schedule the time to send request to WirelessCar Server for asking the car's

```
def create_decision_tree (data, attributes, destination, gain_func):

        if either data or attributes is empty:

            return most_likely_destination

        elif all trips in data have the same destination:

            return the destination of trips

        else:

            best_attribute = select_best_attribute(gain_func)

            tree = {best_attribute:{}}

            # create branches for the best_attribute node
            for val in values_for_best_attribute:

                subtree = create_decision_tree(
                    data_set_for_best_attribute_with_value_val)

                tree[best_attribute][val] = subtree

        return tree
```

Listing 1: Pseudo code for the recursive function to generate decision tree

location. For example, if the driver only drives his car from 6 am to 10 pm, the timetable will collect the time from 6 to 20. The Volvo Cloud Server will request the car's location at every hour from 6 to 20.

The last step is destination prediction. After the location of the car is gained, it will be combined with the current hour and the day of the week. Based on the information of the location, the hour and the day, a destination will be found by using the pre-trained decision tree model. The destination prediction processing is complete.

Considering about the case of real life usage, we adjust the part of majority_book. Instead of choosing the destination having highest frequency, we put all possible destinations with the inputted attributes in the list. The reason to do is increasing the usability of the system. We want to check the traffic information to all those possible destinations for drivers.

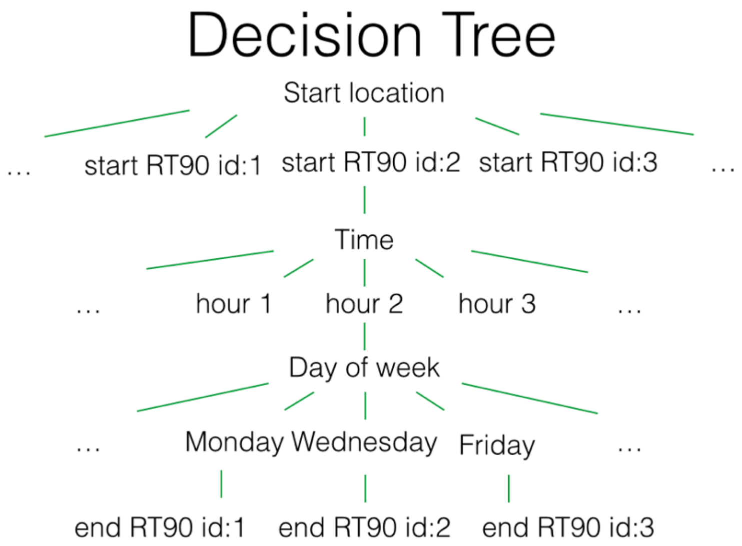


Figure 3.12: An example of decision tree structure

3.3.6 Fetch Traffic Information from INRIX

After a destination is predicted, the next task is to get the real-time traffic information. As described in the previous section, INRIX provides useful traffic information and we can get the traffic information by using its Traffic API. INRIX Traffic API provides a function called findRoute. This function needs inputs of GPS coordinates of the start position and GPS coordinates of the end position. Since our decision tree model used RT 90 ids, we need to translate the RT 90 ids to GPS coordinates. In the experimental testing of study, we selected one GPS coordinate from the RT 90 id from the journal logs. Thus the location is a place the driver had been and it has a high possibility that the location is close to the destination where the driver wants to go.

The function can return at most 3 alternative routes from the start position to the end position. In the data structure of each route, there are route id, travel time and average speed of the route. The most useful information for this paper is travel time and average speed. There are two approaches to decide if the current traffic flow is too slow. The first one is comparing the travel time to the average travel time in journal logs. The second one is comparing the average speed to the threshold that driver specified. In this thesis, we choose the second one.

This function can also find the route for different purpose. We can specify the route type when we send the request to INRIX server. There are two types of route, one is the fastest route and another is the shortest route. This feature is useful if we want to provide driver with optimal route depending on the preference that driver has set.

3.3.7 Notify Driver

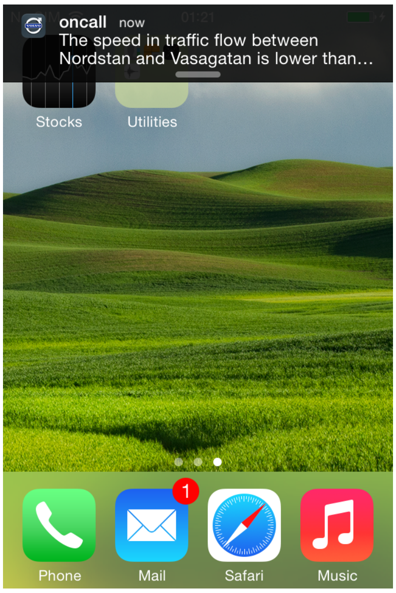
After we get the predicted destination and the traffic information of the route, the last step is to notify the driver with traffic information.

The most popular operating systems for mobile phone (e.g. iOS[37], Android [38] and Windows Phone [39]) support push notification feature. In this thesis, we only test it on the iOS platform. Push notification is a service, which is running in the background, reminding users to know there is information for them. The information could be a text message, calendar event or new data on a remote server.

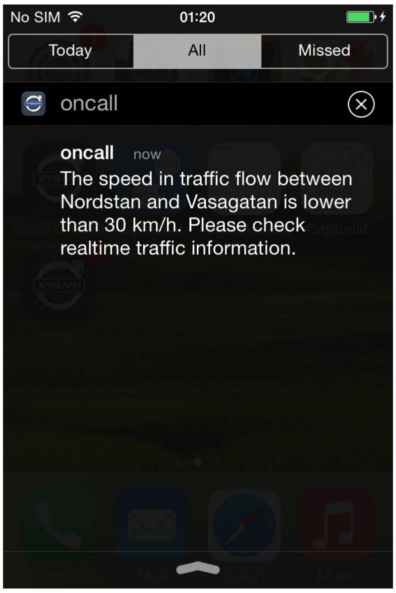
The timing to push a notification to the user is important, because we do not to push useless information to the driver (e.g. the traffic information at 3 am). Since the purpose is to help drivers escape from traffic jam, thus we must try to push the traffic information before the driver start a trip. To predict the exact time of starting a trip for drivers is almost impossible, because we humans are not machine that has a schedule that can be accurate to second. Human's activities are dynamic. A reasonable solution to push notification before starting a trip is pushing before the time in the history journal logs. For instance, if there are logs showing a driver has been starting trips at 8:30, 9:01 and 16:59, then the possible times to push notification will be 8:00, 9:00 and 16:00. This solution will bring a problem that the notification can be pushed too early in the case of 16:59. Thus the present Volvo On Call app has to add a new function that allows users manually checking the current traffic status and the future traffic status. The INRIX API has supported the function of checking predicted traffic information of the future, this is useful for the case in this thesis.

Volvo Cloud Server has to run at the backend and check the timetable at every hour. If the current hour is in the timetable, then request the location of the driver's car by using WirelessCar API. Based on the gained location and the generated decision tree model for destination prediction, predict the possible destination of the trip. Then find routes between the car's location and the predicted destination and extract the average speed value. Comparing the speed to the threshold driver set, only if the speed is lower than the threshold, Volvo Cloud Server pushes a traffic jam warning to the driver.

Figure 3.13(a) and Figure 3.13(b) show the notification on the iOS device. The slow traffic notification will be send to the Volvo Car driver's Volvo On Call app. Based on the notification, the driver can check the real-time traffic information by using Volvo On Call app in order to escape from traffic jam.



(a) Slow traffic notification banner



(b) Notification in notification center

Figure 3.13: Notification on iOS device

4

Testing and Discussion

Destination prediction is the core of the system and it is also the main study of this thesis. Thus the testing is mainly focused on the performance of the destination prediction.

We have five participant drivers in total and we will test prediction for each of them individually. Firstly we choose 200 trips from the driver's history logs for training. Then we choose 20 trips out of those selected trips and 20 trips that differ from those 200 trips. Thus there are 20 trips overlapping among training trips and testing trips. It can be visualized by Figure 4.1. We choose 20 from training trips for testing to avoid the case that all test trips are new, which could cause the problem that all predictions became wrong. Thus the testing should at least make 10% correct prediction. We assume that there are $\boldsymbol{n}$ trips predicted correctly. The correctness ratio of prediction $\boldsymbol{H}$ can be calculated by the formula:

$$H = \frac{n}{200} \times 100\%$$

In order to evaluate our algorithm's performance, we compare the correctness ratio of the algorithm with a baseline. The baseline is also a correctness ratio but it is from a naive predicting algorithm. We use the most visited destination $\boldsymbol{d}$ as predicted destination of testing trips. The baseline can be calculated as the amount of trips (40 testing trips) that visited the $\boldsymbol{d}$ divided by the amount of testing trips. We assume that there are $\boldsymbol{k}$ testing trips visited $\boldsymbol{d}$, and the baseline $\boldsymbol{B}$ can be calculated by formula:

$$B = \frac{k}{40} \times 100\%$$

Finally we get the results for five drivers in Figure 4.2 that shows the results of the prediction for five drivers. In the table, the column *Correctness Ratio* is from algorithm we presented and *Baseline* is the correctness ratio from the naive algorithm. Compare the

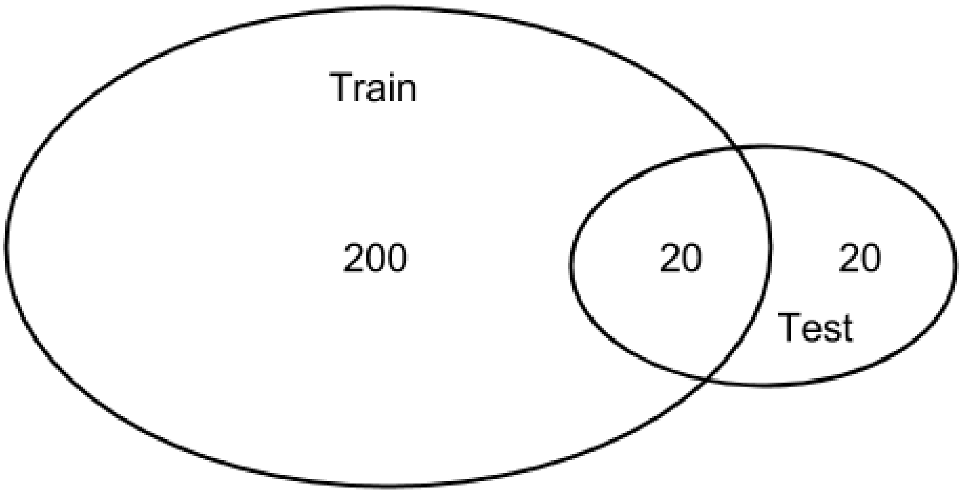


Figure 4.1: Training and testing trips

performance between the algorithm we presented and the naive algorithm, the algorithm we presented is more reliable than the naive one. Among the results of five drivers, the correctness ratios of prediction to four of them are higher than 70%. The mean of correctness ratios for all five driver is about 75%.

	Correctness Ratio: %	Baseline: %
Driver 1	68	38
Driver 2	79	25
Driver 3	81	38
Driver 4	74	40
Driver 5	71	40

Figure 4.2: Testing result for drivers

As a conclusion, the correctness ratio of destination prediction of the presented algorithm for each driver is higher than 60%. The mean correctness ratio of prediction using the algorithm shown in this paper can achieve 75%. The performance of prediction of the algorithm is much better than the naive algorithm.

5

Conclusion

In this thesis, an Intelligent Driver Assistant System was introduced to the current Connected Volvo Car. The decision tree algorithm in this thesis can enhance the current Volvo Cloud predicting the destination of the driver's trip, which opens a door to a more useful and intelligent Volvo Car for the future.

The introduced Intelligent Driver Assistant System can predict the destination of a trip by utilizing of the driving journal logs that are from driver's Connected Volvo Car. Based on the predicted destination result, the system can automatically check the real-time traffic information by using INRIX Traffic service. The Intelligent Driver Assistant System can also push notification to drivers when there is traffic jam or traffic incidents ahead of time, which can assist the driver to change the route or postpone setting off. Besides the traffic incidents information, the INRIX server can also provide alternative routes to the destination based on two different route types: shortest first and fastest first. Driver can choose either one based on their preference.

The decision tree algorithm to predict destination has been tested on five participants who drive Connected Volvo cars. There are 3063 trips in total from five different participant Volvo Car drivers. We select 200 trips from each driver's log for training. After that we select 40 trips from each driver's log for testing and 20 of them are overlapping with training trips. The lowest correctness ratio of destination prediction is 68% for driver 1 while the highest is 81% for driver 3. For all five drivers, the mean correctness ratio of prediction is about 75%.

6

Future Work

The introduced Intelligent Driver Assistant System can predict the destination before the driver starting a new trip. It does not work in real-time when the car is already on the road. The reason is that most of Volvo Cars in the market do not communicate with Volvo Cloud Server, which is supported by WirelessCar, in real-time. The car only sends the GPS location to Volvo Cloud Server when the key is in and out. All of five participants were driving this type of Volvo Cars. In the newest model Volvo Car V60, a new GPS device has been equipped that can send GPS location of the car to Volvo Cloud Server in real-time. One improvement that can be implemented is an Intelligent Driver Assistant system that can also predict the destination in real-time, not only predicting before driver setting off. Other improvements also can be done such as collaborating with Calendar and Mailbox, which can improve the accuracy of the prediction and the usability of the system. For example, based on the calendar event, the system can automatically send email to meeting participants to notify with the driver's current position and estimated time of arrival. Based on the prediction of the destination and the parking lot booking function on Volvo V60, we can also implement the function to give parking lot suggestions to a driver without the driver manually typing the destination. Suggesting the driver to rearrange the route to nearby petrol station when the system detects the fuel status is low.

A great number of useful functions can be implemented based on the destination prediction. This Volvo Car Intelligent Assistant System opens a door to the future for vehicle industry. Outside the door, too much new things with great useful functions that can benefit drivers and road safety can be explored. The new Intelligent Car System will make the car become an assistant of the driver. It is more like a human agent who can give drivers advises not only a machine with cold steel body.

Bibliography

[1] D. Landes, Ericsson to help volvo cars connect to the net @ONLINE (2012).
URL http://www.volvocars.com/intl/sales-services/sales/volvo-on-call/Pages/default.aspx

[2] C. Evans-Pughe, The connected car, IEE Review 51 (1) (2005) 42–46.

[3] B. Farr, U. Shadow Creek Consulting Ltd, E. Peytchev, The connected car–building a real world testbed for vehicle communication.

[4] C. Davies, Bmw connected car concept finds parking and deals as you drive @ONLINE (2014).
URL http://www.slashgear.com/bmw-connected-car-concept-finds-parking-and-deals-as-you-drive-23314283/

[5] W. Foundation, Consumer electronics show @ONLINE (2014).
URL http://en.wikipedia.org/wiki/Consumer_Electronics_Show

[6] E. N. Center, Volvo launches new cloud-based infotainment system for its connected car @ONLINE (2014).
URL http://www.ericsson.com/news/140108-volvo-launches-new-cloud-based-infotainment-system-for-its-connected-car_244099438_c

[7] G. Richards, Intelligent cars [control forecasts], Engineering Technology 5 (1) (2010) 40–41.

[8] V. C. G. G. M. Newsroom, Volvo car group's first self-driving autopilot cars test on public roads around gothenburg @ONLINE (2014).
URL https://www.media.volvocars.com/global/en-gb/media/pressreleases/145619/volvo-car-groups-first-self-driving-autopilot-cars-test-on-public-roads-around-gothenburg

[9] T. Smith, Volvo demos automobile auto-pilot tech @ONLINE (2011).
URL http://www.theregister.co.uk/2011/01/17/project_sartre_vehicle_platooning_demo/

[10] G. Di Lorenzo, F. Pinelli, F. C. Pereira, A. Biderman, C. Ratti, C. Lee, An affective intelligent driving agent: Driver's trajectory and activities prediction, in: Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th, IEEE, 2009, pp. 1–4.

[11] T. M. Fumitaka Nakahara, A destination prediction method based on behavioral pattern analysis of nonperiodic position logs, Information Processing Society of Japan.

[12] V. Kostov, J. Ozawa, M. Yoshioka, T. Kudoh, Travel destination prediction using frequent crossing pattern from driving history, in: Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE, IEEE, 2005, pp. 343–350.

[13] D. Ashbrook, T. Starner, Using gps to learn significant locations and predict movement across multiple users, Personal and Ubiquitous Computing 7 (5) (2003) 275–286.

[14] J. Krumm, Real time destination prediction based on efficient routes, Tech. rep., SAE Technical Paper (2006).

[15] V. S. Tiwari, A. Arya, S. Chaturvedi, Route prediction using trip observations and map matching, in: Advance Computing Conference (IACC), 2013 IEEE 3rd International, IEEE, 2013, pp. 583–587.

[16] T. Terada, M. Miyamae, Y. Kishino, K. Tanaka, S. Nishio, T. Nakagawa, Y. Yamaguchi, Design of a car navigation system that predicts user destination, in: Mobile Data Management, 2006. MDM 2006. 7th International Conference on, IEEE, 2006, pp. 145–145.

[17] A. Y. Xue, R. Zhang, Y. Zheng, X. Xie, J. Huang, Z. Xu, Destination prediction by sub-trajectory synthesis and privacy protection against such prediction, in: Data Engineering (ICDE), 2013 IEEE 29th International Conference on, IEEE, 2013, pp. 254–265.

[18] A. Y. Xue, R. Zhang, Y. Zheng, X. Xie, J. Yu, Y. Tang, Desteller: A system for destination prediction based on trajectories with privacy protection, Proceedings of the VLDB Endowment 6 (12) (2013) 1198–1201.

[19] R. Simmons, B. Browning, Y. Zhang, V. Sadekar, Learning to predict driver route and destination intent, in: Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE, IEEE, 2006, pp. 127–132.

[20] J. A. Alvarez-Garcia, J. A. Ortega, L. Gonzalez-Abril, F. Velasco, Trip destination prediction based on past gps log using a hidden markov model, Expert Systems with Applications 37 (12) (2010) 8166–8171.

[21] L. Nguyen, H.-T. Cheng, P. Wu, S. Buthpitiya, Y. Zhang, Pnlum: System for prediction of next location for users with mobility, in: Nokia Mobile Data Challenge 2012 Workshop. p. Dedicated challenge, Vol. 2, 2012.

[22] M. Morzy, Mining frequent trajectories of moving objects for location prediction, in: Machine Learning and Data Mining in Pattern Recognition, Springer, 2007, pp. 667–680.

[23] A. Monreale, F. Pinelli, R. Trasarti, F. Giannotti, Wherenext: a location predictor on trajectory pattern mining, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2009, pp. 637–646.

[24] C. Manasseh, R. Sengupta, Predicting driver destination using machine learning techniques, in: Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on, 2013, pp. 142–147.

[25] WirelessCar, Five key facts about wirelesscar @ONLINE (2014).
URL `http://www.wirelesscar.com/wordpress/?page_id=26`

[26] V. C. Group, Volvo on call @ONLINE (2014).
URL `http://www.volvocars.com/intl/sales-services/sales/volvo-on-call/Pages/default.aspx`

[27] V. C. Group, Volvo on call mobile app @ONLINE (2014).
URL `http://www.volvocars.com/intl/sales-services/sales/volvo-on-call/pages/mobile-app.aspx`

[28] WirelessCar, Volvo on call, volvo cars @ONLINE (2014).
URL `http://www.wirelesscar.com/wordpress/?page_id=502`

[29] INRIX, Who we are @ONLINE (2014).
URL `http://www.inrix.com/companyoverview.asp`

[30] A. Abdelhalim, I. Traore, A new method for learning decision trees from rules, in: Machine Learning and Applications, 2009. ICMLA'09. International Conference on, IEEE, 2009, pp. 693–698.

[31] J. Mingers, An empirical comparison of pruning methods for decision tree induction, Machine learning 4 (2) (1989) 227–243.

[32] W. Foundation, Decision tree @ONLINE (2014).
URL `http://en.wikipedia.org/wiki/Decision_tree`

[33] Scikit-learn, Decision trees @ONLINE (2014).
URL `http://scikit-learn.org/stable/modules/tree.html`

[34] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106.

[35] F. G. D. Committee, United states national grid — federal geographic data committee @ONLINE (2014).
URL `https://www.fgdc.gov/usng`

[36] Lantmäteriet, Rt 90 @ONLINE (2014).
URL http://www.lantmateriet.se/en/Maps-and-geographic-information/GPS-and-geodetic-surveys/Reference-systems/Two-dimensional-systems/RT-90/

[37] A. Inc., Local and push notification programming guide @ONLINE (2014).
URL https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Introduction.html

[38] A. Developers, User notifications @ONLINE (2014).
URL http://developer.android.com/google/gcm/notifications.html

[39] W. D. Center, Push notifications for windows phone 8 @ONLINE (2014).
URL http://msdn.microsoft.com/library/windows/apps/ff402558(v=vs.105).aspx