EPiC Series in Computing

Volume 76, 2021, Pages 95–104

SEDE 2020. 29th International Conference on
Software Engineering and Data Engineering

Ameliorating Accuracy of a Map Navigation When Dealing with Different Altitude Traffics that Share Exact Geolocation

Thitivatr PatanasakPinyo

Faculty of Information and Communication Technology, Mahidol University
Salaya, Nakhon Pathom, 73170, Thailand
thitivatr.pat@mahidol.edu

Abstract

Many users use a location-based application on a portable device to be a navigator when driving. However, there exists an incident that two roads are located on the same geolocation, i.e., same values of latitude and longitude but different altitude, for very long distance where one road is located on the ground level and another one is elevated. This incident mostly confuses a location-based application to precisely retrieve the actual road that a vehicle is currently on and, consequently, causes the application to either navigate incorrectly or suggest a route that is a detour. Calling an altitude from a GPS sensor might be a possible solution but it came with problems of accuracy, especially for mid-grade GPS sensors that equipped with most smartphone in today's market. We proposed a concept of implementing a classification model that can classify whether a vehicle is on a ground road or an elevated road regardless of geolocation data. We trained and validated two models using a dataset that we had collected from actual driving on two roads in Thailand that fell under this condition. A data instance that we collected contained measurements related to driving or driving environment such as a real-time speed at any certain interval of time. We reported validation results of both models as well as other important statistics.

1 Introduction

Location-based application is an application that has at least one module related to the location information. We can see a plenty of location-based applications in software marketplace. Good instances of location-based application are Google Maps, Bing Map, Yelp, The Weather Channel, etc. Nowadays, an application like Google Maps is very popular to users around the world because we can use it for navigation. With respect to the fact that a smart device (e.g., smartphone, phablet, or tablet) is cheap and it is affordable for everyone to own it, most people prefer to use a location-based application on a smart device to navigate when they travel rather than a traditional GPS device or even a traditional paper map.

Using location-based application on a GPS-enabled device in multiple activities, e.g., census or navigation, might yield a lot of benefits such as allowing a user to achieve his/her objective faster and more comfortable, however, it is a double-edge sword. Existing articles reported couple cases of failure of relying on location-based software [3]. Google Maps is a major player in the battle field of location-based applications that supports navigation function. Users trust it to serve as a navigator when they travel using road traffic, particularly, when they drive their private automobile. Google Maps retrieves a geolocation of a vehicle, i.e., a pair of latitude and longitude, from satellites and then represents it on the map [4, 5, 16] as well as gives a driver what should be the next action (e.g., take an exit, take a ramp, turn right at the intersection, enter the roundabout and take the 3rd exit, etc.) so the driver realizes where he actually is and what he should do next such that he can successfully arrive his destination.

However, there are multiple road conditions that cause a user of Google Maps to have a hard time. One condition that we found is an incident that there are two overlapped roads that located and share the same set of geolocations, i.e., those roads share the same values of both latitude and longitude. Even though this kind of road condition is rare in a country that occupies very large area like United States, Canada, or Australia, we found that it is dissimilar in a small country like Thailand, Singapore, etc. There is nothing to be surprised if we reside in Thailand and see a turnpike parallely overlapping with a ground-level road for more than twenty kilometers, which look like two parallel lines on a plane. A good instance is Borommaratchachonnani (pronounced as "*Bor-Rom-Ma-Raj-Cha-Chon-Na-Ni*") Elevated Road (Figure 1) and Borommaratchachonnani Frontage Road (Figure 2). Borommaratchachonnani Elevated Road was 15-meter elevated from the ground level. Those two roads, which located in Bangkok, Thailand, are 16-kilometer long (Figure 3). When Google map encounters this such condition, it is not able to correctly distinguish whether a vehicle is on the elevated road or the frontage road (ground level) because every corresponding location on those two roads shares exactly the same geolocation, i.e., same values of a latitude and a longitude. This resulted in an incorrect map representation on the application UI (Figure 4). In Figure 4 (a), the car was on the rightmost lane of Borommaratchachonnani Frontage Road (eastbound) on January 17, 2019, but Google map incorrectly represented as it was on the elevated road. Similar mistake can be seen in Figure 4 (b), which was taken on January 25, 2019 at 11:12am, when the driver was driving on the rightmost lane of Borommaratchachonnani Frontage Road (eastbound) heading to Central Plaza Westgate Shopping Mall. In both situations (Figure 4), Google Map incorrectly understood that both vehicles were on the elevated road, therefore, it suggested the drivers to take an exit to Sirindhorn Road (instead of making just a simple left turn) and take a left to Liap Thang Rotfai Taling Chan Road, which was a major detour. Since this type of road/traffic condition is very uncommon in USA, Google Maps developers might not be aware of this difficulty.

The mentioned issue might simply be solved by comparing altitudes of both roads. An altitude of a location is a measurement representing a vertical difference between the location and a sea level. Even though there mostly exists a built-in method to retrieve a real-time value of an altitude, most location-based application developers decided not to use it as we can see an obvious example from Figure 4. One reason that might be related is about an accuracy of an altitude retrieved by a device's sensor. Generally, we measure an altitude of any location by using an altimeter. However, most mobile devices are not equipped with an altimeter sensor for some reason. Therefore, a method that a mobile device uses to retrieve an altitude is to combine the use of both a GPS sensor and a barometer sensor to evaluate an altitude, which is not a direct way. Normally, a horizontal error that we can expect from a dedicated mid-grade GPS device is around ±15 meter 95% of the time [6, 7]. It is worse with a vertical error since we can expect around ±23 meter or 1.5 times greater than a horizontal error. This vertical

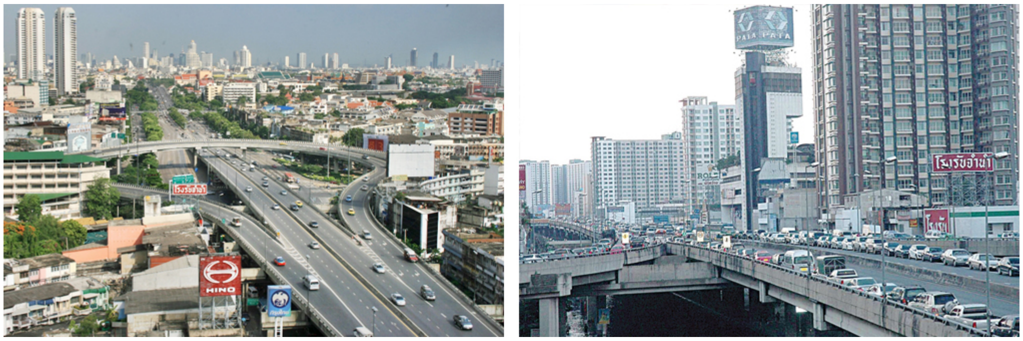


Figure 1: Borommaractchachonnani Elevated Road (image source: *www.dailynews.co.th*).

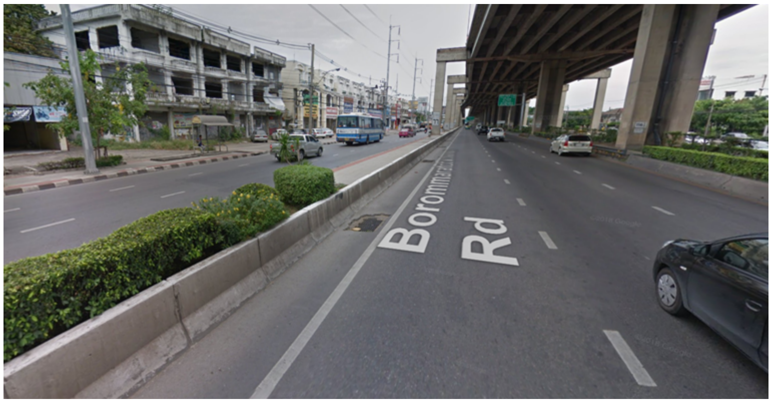


Figure 2: Borommaractchachonnani Frontage Road (image source: *Google Maps Street View*).

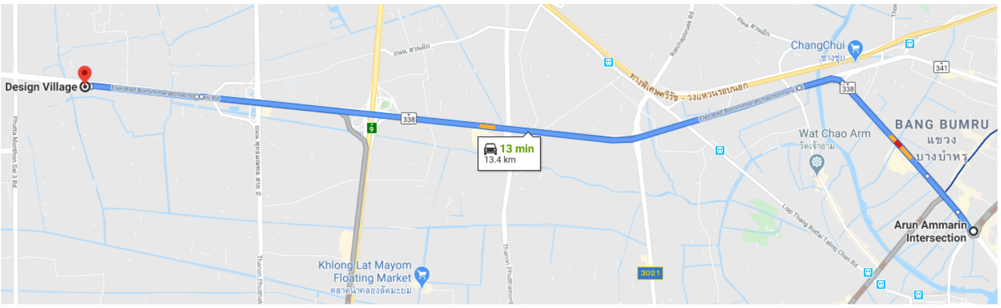


Figure 3: A Google Map shows the location of Borommaratchachonnani Road (both elevated and frontage).

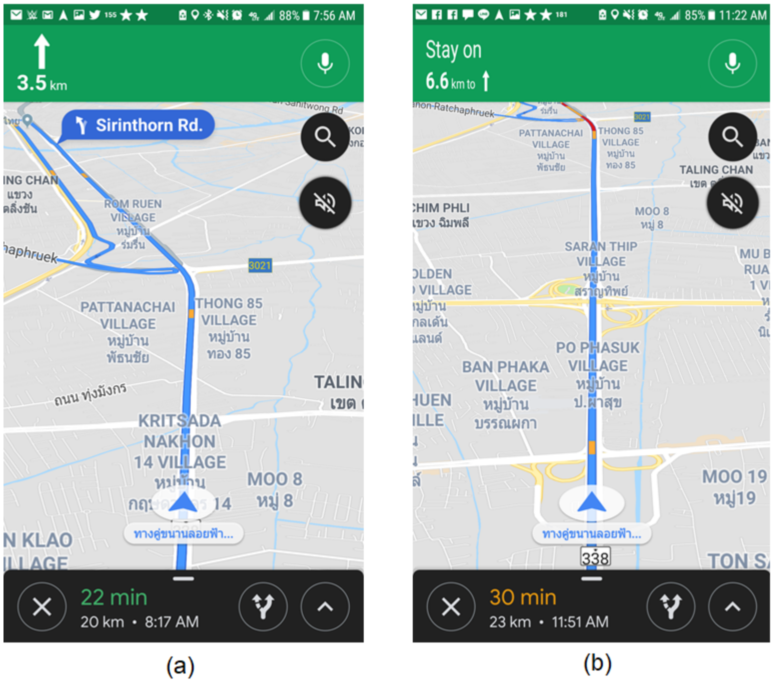


Figure 4: Google Map showed incorrect real-time location detected on: (a) January 17, 2019 (b) January 25, 2019.

error might be increased when we switch from a dedicated GPS device to just a GPS sensor in a mobile phone. Based on the case that we selected, Borommaratchachonnani Elevated Road is 15-meter elevated from the ground, the average error of an altitude (±23 meter) is big enough for a mobile phone to misunderstand the actual location whether or not a vehicle is currently on an elevated road and, consequently, incorrectly navigate the driver that can lead to a detour. This is still a big challenge even we change to an expensive GPS device rather than the average one because it can reduce a vertical error to only ±10 meter. Therefore, relying on an altitude might be impractical for this issue.

We then proposed to implement a computer classification technique as a geolocation-free solution to solve this matter. We will create a classification model that retrieves as input a set of real-time traffic data and combines with essential aggregated data, if needed, then outputs a result whether a vehicle is on an elevated road or a ground-level road. The output from the classification model can be utilized by being sent to a location-based application that is used as a navigator. The application then shows a precise map representation of a vehicle on the application UI.

We organized this paper as follows: Section 2 walks through related literature. Section 3 describes the details of the study, which include both designs and procedures. Section 4 shows the results of validation of our models. Finally, Section 5 summarizes the contribution of models described in this paper.

2 Related Work

Nobody can deny a statement that we are in the era of ubiquitous computing at this moment. From a very high performance server machine to a handheld device like a smartphone, a computer application becomes something that people cannot live without. Most smartphone applications are location-based which have a map or geolocation data involved. An integration with a map can serve a user more objectives than ever before. An experiment of Batinov et al. [1] implemented a location-based application for a purpose of census. Several studies also focused on the use of location-based application for an address verification task [10, 11, 13, 14, 15]. PatanasakPinyo et al. [12] proposed a method to add an indexing ability to a location-based application using a raster map. Sulaiman [17] and Whitney [18] introduced a way to increase both accuracy and usability of using a location-based application. Modsching et al. [8] examined and reported an accuracy on a location-based application for pedestrian navigation in the city of Gorlitz. Park and Kim [9] analyzed driver acceptance of a car navigation system.

3 Methodology

Since we cannot rely on an altitude retrieved from a GPS sensor in a mobile device to be an indicator to distinguish whether a car is on an elevated road or a ground-level road, we ended up with an idea of implementing a classification model without using geolocation data as a model's feature, particularly, a latitude, a longitude, and an altitude were excluded from the model. This study was divided into three parts, i.e., study design, data gathering, and model implementation.

3.1 Study Design

First, we selected both Borommaratchachonnani Elevated Road and Borommaratchachonnani Frontage Road that we have already mentioned to be our study site. Next, we defined a set of features of a model with one constraint that geolocation variables (latitude, longitude, altitude) cannot be included. We came up with all possible nine variables that involved in a driving context or a driving environment. Those nine variables were:

1. **Distance:** A distance that a vehicle moves within a certain interval of time. A value of this variable can be obtained by evaluating a difference between two points (start and end) at a certain interval of time.

2. **Speed:** A speed of a vehicle at a certain point of time. A value of this variable can be obtained by calculating a rate of change of a distance by a certain time interval.

3. **Direction:** A direction that a vehicle heading to. A value of this variable can be obtained (and represented) by evaluating an angle θ between a line $\langle p_s, p_e \rangle$ and a line $y = 0$ where p_s and p_e are a start point and an end point of a certain time interval, respectively.

4. **Light Intensity:** A light intensity that can be retrieved by a light sensor which always comes with most smartphone devices.

5. **Time:** A timestamp.

6. **Day-of-Month:** A day of month.

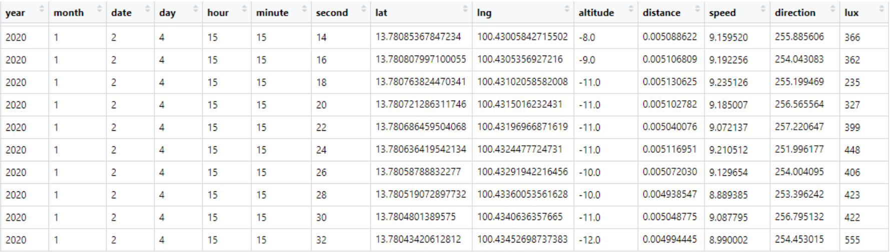
7. **Day-of-Week:** A day of week.

8. **Bound:** A road condition, which can be {northbound, southbound, eastbound, westbound}
9. **Road Type:** A class variable indicating whether a vehicle is on an elevated road or a frontage road.

The next step was to develop an application on a mobile device that able to automatically retrieve and log values of those variables. When the set of variables and the application were stable and ready, we started a process of data gathering by driving a car on those two roads.

3.2 Data Gathering

We developed an Android application that retrieved a value directly from a device's sensor such as a light sensor and also be able to calculate real-time variables, e.g., speed. Even though we did not use any geolocation data (latitude, longitude, altitude) as model's features, the application still retrieved both a latitude and a longitude for the purpose of calculating a horizontal distance. A handler inside the application was set to trigger and call up all required values from either device's sensors or aggregation processes every two second (2-second interval). All variables were automatically logged and saved to the device's storage. We deployed this application on Samsung Galaxy Tab A (2019). After the application was successfully deployed, we started the process of data gathering by driving on both roads while having the application ran. We made totally fifteen trips for data gathering. Four trips were on the elevated road eastbound. Four trips were on the frontage road eastbound. Three trips were on the elevated road westbound. And last four trips were on the frontage road westbound. Figure 5 shows an example of data that were logged by the application.



year	month	date	day	hour	minute	second	lat	lng	altitude	distance	speed	direction	lux
2020	1	2	4	15	15	14	13.78085367847234	100.43005842715502	-8.0	0.005088622	9.159520	255.885606	366
2020	1	2	4	15	15	16	13.780807997100055	100.4305356927216	-9.0	0.005106809	9.192256	254.043083	362
2020	1	2	4	15	15	18	13.780763824470341	100.43102058582008	-11.0	0.005130625	9.235126	255.199469	235
2020	1	2	4	15	15	20	13.780721286311746	100.4315016232431	-11.0	0.005102782	9.185007	256.565564	327
2020	1	2	4	15	15	22	13.780686459504068	100.43196966871619	-11.0	0.005040076	9.072137	257.220647	399
2020	1	2	4	15	15	24	13.780636419542134	100.4324477724731	-11.0	0.005116951	9.210512	251.996177	448
2020	1	2	4	15	15	26	13.78058788832277	100.43291942216456	-10.0	0.005072030	9.129654	254.004095	406
2020	1	2	4	15	15	28	13.780519072897732	100.43360053561628	-10.0	0.004938547	8.889385	253.396242	423
2020	1	2	4	15	15	30	13.7804801389575	100.4340636357665	-11.0	0.005048775	9.087795	256.795132	422
2020	1	2	4	15	15	32	13.78043420612812	100.43452698737383	-12.0	0.004994445	8.990002	254.453015	555

Figure 5: Data logged by the application.

3.3 Model Implementation

For training a model, we decided to use two classifiers: **Random Forest** and **Bagging**. A random forest is a classifier that will generate n decision trees. For each tree, a classifier randomly selected a variable to make a decision which child node should be selected at each split. A bagging also implements the concept of n iterations like a random forest except that it allows us to configure a base classifier to be anything not only a decision tree.

We implemented (i.e., trained and validated) the two models corresponding to two classifiers via Weka [2]. For the first model implemented using a random forest as a classifier, most parameters for the classifier were set as default. However, we set a number of trees to 50. Furthermore, we limited a model's visibility to eliminate a bias by set a bag size to 5, i.e., each tree would see only 5% of all instances rather than all of them.

For the other model implemented using a bagging method, we set a bag size to 5 due to the bias reason. The number of bag also was 50. We chose a REPTree (default base classifier) as a base classifier of bagging.

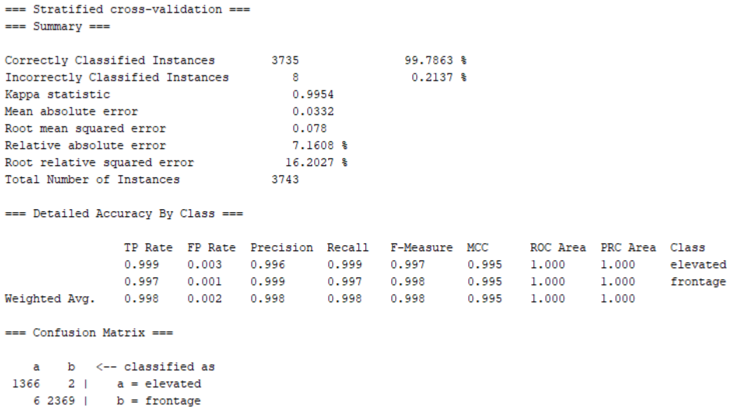
There were 3743 instances of data that we could gather from the data gathering process (This number was a result after we cleaned out all incomplete instances, i.e., an instance that contained a null value). All instances were used in the training process. From 3743 instances, 1368 instances were labeled "Elevated Road" and 2375 instances were labeled "Frontage Road". For both models, we selected a stratified cross-validation with ten folds to be a validation method.

4 Results and Discussion

We divided this section into two parts according to the two models that we implemented.

4.1 Classification Model 1

Applying the configuration that we previously mentioned on a random forest, we trained the first model with all 3743 records. Even though both classes were imbalanced, particularly, instances of a frontage road outnumbered the other class by 1007 instances, we decided not to over-sample nor under-sample. We would like to observe the preliminary result that we can achieve from the original dataset. The result of 10-fold cross validation showed that there were eight instances that were incorrectly classified, particularly, six instances of frontage road were classified as elevated road and two instances of elevated road were classified as frontage road (Figure 6). The overall accuracy of the model was 99.7863%. The ROC curve and detailed accuracy statistics plus a confusion matrix were reported in Figures 7 and 6, respectively.



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        3735               99.7863 %
Incorrectly Classified Instances         8                0.2137 %
Kappa statistic                          0.9954
Mean absolute error                      0.0332
Root mean squared error                  0.078
Relative absolute error                  7.1608 %
Root relative squared error             16.2027 %
Total Number of Instances             3743

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.999    0.003    0.996      0.999    0.997      0.995    1.000     1.000     elevated
                 0.997    0.001    0.999      0.997    0.998      0.995    1.000     1.000     frontage
Weighted Avg.    0.998    0.002    0.998      0.998    0.998      0.995    1.000     1.000

=== Confusion Matrix ===

    a    b   <-- classified as
 1366    2 |    a = elevated
    6 2369 |    b = frontage
```

Figure 6: Validation Result of Classification Model 1.

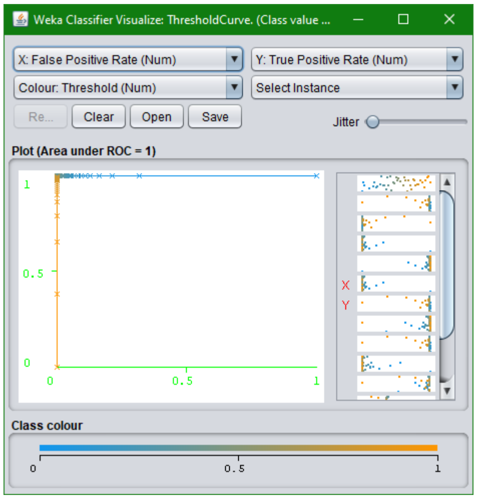
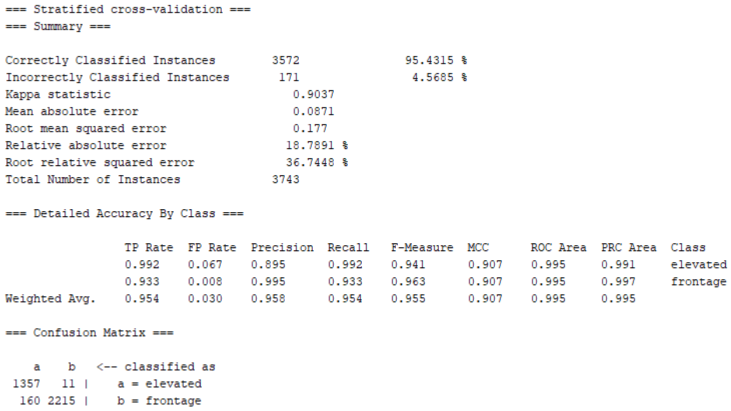

Weka Classifier Visualize: ThresholdCurve. (Class value ...
X: False Positive Rate (Num)
Y: True Positive Rate (Num)
Colour: Threshold (Num)
Select Instance
Re...
Clear
Open
Save
Jitter
Plot (Area under ROC = 1)
1
0.5
0
0
0.5
1
X
Y
Class colour
0
0.5
1

Figure 7: ROC of Classification Model 1.

4.2 Classification Model 2

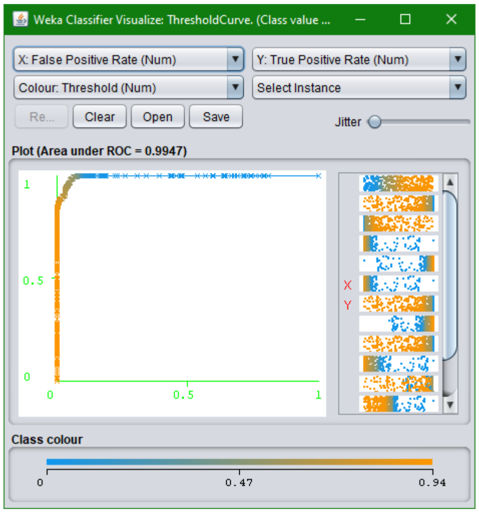
For the second model, we also applied the same validation strategy, which was a stratified 10-fold cross validation. 160 instances of frontage road were classified as elevated road. 11 instances of elevated road were classified as frontage road. The overall accuracy of the model was 95.4315%. The ROC curve and detailed accuracy statistics plus a confusion matrix were reported in Figures 9 and 8, respectively.



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        3572               95.4315 %
Incorrectly Classified Instances       171                4.5685 %
Kappa statistic                          0.9037
Mean absolute error                      0.0871
Root mean squared error                  0.177
Relative absolute error                 18.7891 %
Root relative squared error             36.7448 %
Total Number of Instances             3743

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.992    0.067    0.895      0.992    0.941      0.907    0.995     0.991     elevated
                 0.933    0.008    0.995      0.933    0.963      0.907    0.995     0.997     frontage
Weighted Avg.    0.954    0.030    0.958      0.954    0.955      0.907    0.995     0.995

=== Confusion Matrix ===

    a    b   <-- classified as
 1357   11 |    a = elevated
  160 2215 |    b = frontage
```

Figure 8: Validation Result of Classification Model 2.

From both validation results, we observed that the first model yielded better result of validation with higher in both accuracy and area under ROC.



Weka Classifier Visualize: ThresholdCurve. (Class value ...

X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Select Instance

Plot (Area under ROC = 0.9947)

Figure 9: ROC of Classification Model 2.

5 Conclusion

We trained and validated two classification models to classify a road type whether a vehicle was on an elevated road or a frontage road by using features that were not a geolocation value since both a latitude and a longitude were not capable of distinguishing any vertical difference and an altitude was involved with accuracy issues. The first model was implemented using a random forest as a classifier while the other one was implemented using a bagging approach. For both models, we set a bag size to be 5 that allowed each bag/tree to see only 5% of all data instances to eliminate a bias as much as possible. Each data instance consisted of distance, speed, direction, light intensity, timestamp, day-of-month, day-of-week, bound, and road type. After we did stratified 10-fold cross validations to both models, the first model (random forest) yielded an accuracy of 99.7863% where the second model (bagging) yielded an accuracy of 95.4315%. Our future work is to test both models on real-time data when driving on both the roads to observe their performances and we will then apply this concept of classification model to other pairs of roads in the country that have the same condition as Borommaratchachonnani Elevated Road and Frontage Road.

6 Acknowledgements

This research project is supported by Mahidol University. We would like to express our special thanks to Mahidol University, who gave us a funding, and the Faculty of Information and Communication Technology for greatly supporting us to pursue the study.

References

[1] Georgi Batinov, Michelle Rusch, Tianyu Meng, Kofi Whitney, Thitivatr Patanasakpinyo, Les Miller, and Sarah Nusser. Understanding map operations in location-based surveys. In *Eighth*

International Conference on Advances in Computer-Human Interactions (ACHI 2015), pages 144–149, Lisbon, Portugal, 2015.

[2] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[3] Allen Yilun Lin, Kate Kuehl, Johannes Schöning, and Brent Hecht. Understanding "death by gps" a systematic study of catastrophic incidents associated with personal navigation technologies. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 1154–1166, 2017.

[4] Paul W McBurney and Arthur N Woo. Infrastructure-aiding for satellite navigation receiver and method, October 29 2002. US Patent 6,473,030.

[5] Paul W McBurney and Arthur N Woo. Satellite navigation receiver and method, August 20 2002. US Patent 6,437,734.

[6] Joe Mehaffey. Gps altitude readout how accurate? *Retrieved*, 12(12):2016, 2001.

[7] Joe Mehaffey. Error measures, 2011.

[8] Marko Modsching, Ronny Kramer, and Klaus ten Hagen. Field trial on gps accuracy in a medium size city: The influence of built-up. In *3rd workshop on positioning, navigation and communication*, volume 2006, pages 209–218, 2006.

[9] Eunil Park and Ki Joon Kim. Driver acceptance of car navigation systems: integration of locational accuracy, processing speed, and service and display quality with technology acceptance model. *Personal and ubiquitous computing*, 18(3):503–513, 2014.

[10] Thitivatr PatanasakPinyo. *Flattening methods for adaptive location-based software to user abilities*. Graduate Theses and Dissertations, Iowa State University, 2017.

[11] Thitivatr PatanasakPinyo, Georgi Batinov, Kofi Whitney, and Les Miller. Methods that flatten the user space for individual differences in location-based surveys on portable devices. In *31st International Conference on Computers and Their Applications (CATA 2016)*, pages 65–70, Las Vegas, Nevada, 2016. International Society for Computers and their Applications (ISCA).

[12] Thitivatr Patanasakpinyo, Georgi Batinov, Kofi Whitney, Adel Sulaiman, and Les Miller. Object-indexing: A solution to grant accessibility to a traditional raster map in location-based application to accomplish a location-based task. *International Journal of Computing, Communication and Instrumentation Engineering (IJCCIE)*, 5(1), 2018.

[13] Thitivatr Patanasakpinyo, Georgi Batinov, Kofi Whitney, Adel Sulaiman, and Les Miller. Enhanced prediction models for predicting spatial visualization (vz) in address verification task. In Gordon Lee and Ying Jin, editors, *Proceedings of 34th International Conference on Computers and Their Applications*, volume 58 of *EPiC Series in Computing*, pages 247–256. EasyChair, 2019.

[14] Thitivatr PatanasakPinyo, Georgi Batinov, Kofi Whitney, Adel Sulaiman, Les Miller, and Stephen Gilbert. Extracting useful features for users with different levels of spatial visualization. In *33rd International Conference on Computers and Their Applications (CATA 2018)*, pages 86–91, Las Vegas, Nevada, 2018. International Society for Computers and their Applications (ISCA).

[15] Thitivatr Patanasakpinyo and Les Miller. Ui error reduction for high spatial visualization users when using adaptive software to verify addresses. In Gordon Lee and Ying Jin, editors, *Proceedings of 35th International Conference on Computers and Their Applications*, volume 69 of *EPiC Series in Computing*, pages 22–31. EasyChair, 2020.

[16] Pierluigi Silvestrin, Peter Daly, David Walsh, and Eric Aardoom. Receiver for a navigation system, in particular a satellite navigation system, May 30 2000. US Patent 6,069,583.

[17] Adel Sulaiman. *Training and evaluation in a large-scale virtual environment for a location-based mobile application*, volume 17573. Graduate Theses and Dissertations, Iowa State University, 2019.

[18] Kofi Whitney. *Taking the lab on the road and bringing the road to the lab: On using mixed-methods and virtual reality to study a location-based task*, volume 17123. Graduate Theses and Dissertations, Iowa State University, 2019.