Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 109C (2017) 474–481

Procedia Computer Science

www.elsevier.com/locate/procedia

The 8th International Conference on Ambient Systems, Networks and Technologies (ANT 2017)

Analyzing Gap Acceptance Behavior at Unsignalized Intersections Using Support Vector Machines, Decision Tree and Random Forests

Rakshita Nagalla[a], Prasanna Pothuganti[b], Digvijay S. Pawar[c*]

[a]*B.Tech, Department of Electrical Engineering, IIT Hyderabad, Hyderabad 502285, India*
[b]*B.Tech, Department of Civil Engineering, IIT Hyderabad, Hyderabad 502285, India*
[c*]*Assistant Professor, Department of Civil Engineering, IIT Hyderabad, Hyderabad 502285, India*

Abstract

Driver's gap acceptance behaviour highly influences the performance and safety of unsignalized intersections. At unsignalized intersections, crossing drivers have to accept or reject the available gap. The gap acceptance decision is influenced by different dynamics such as traffic, geometric, environmental and human factors. The decision to cross the major road largely depends on the speed, distance and the heaviness of the potentially conflicting vehicle on the major road. The driver's gap acceptance data was collected at 4-legged unsignalized intersections. The information such as the type of crossing vehicle, conflicting vehicle, speed and the spatial gap was extracted from the video data. This paper deals with the application of three widely used non-parametric data mining techniques, namely, Support Vector Machines (SVMs), Random Forests (RF) and Decision Trees (DT) to predict the gap acceptance behaviour of the driver. While SVMs are insensitive to class imbalance, decision tree generated by CART algorithm provides critical insights into decision making process employed by the driver. Random forests and decision tree implicitly establish the relative importance of different factors affecting the driver's decision. Further, skill scores used to validate the models revealed that SVM and DT models performed almost similarly whereas, RF model outperformed SVM and DT.

Keywords: Classification and Regression Trees; Random forests; Support vector machines; gap acceptance; uncontrolled intersections.

***Corresponding Author**
Dr. Digvijay S. Pawar, Email: dspawar@iith.ac.in, Ph: 040 2301 6167

10.1016/j.procs.2017.05.312

1. Introduction

Gap acceptance is one of the most important traffic characteristics that has been widely used in the analysis of unsignalized intersections. At unsignalized intersections, drivers have to either accept or reject the available gaps. The "yes/no" nature of this decision gives gap acceptance a distinctive set of conditions that can be used in the analysis. Various studies have analyzed minimum gap (i.e., critical gap) required for the drivers for safe maneuvering. There are a variety of factors which influence the decision of minor stream driver during gap acceptance. For example, Polus et al. [1] analyzed the gap acceptance at roundabouts and stated that increased waiting time reduced the critical gap and vice versa following an 'S' curve. The cumulative number of accepted and rejected gaps were used by Raff [2] to understand gap acceptance behavior. Hewitt [3] used maximum likelihood method to estimate average value and variance of critical gap based on the assumption that accepted gaps and rejected gaps follow the normal distribution. Hamed et al.[4] used multiple regression model to analyze gap acceptance as a function of the velocity of the major stream, the size of turn-left lane, lane number of minor road and conflicting flow volume. Pant et al. [5] applied neural networks and binary logit models to analyze gap acceptance behavior of vehicles at stop-controlled intersections. Pawar et al. [6] applied Support Vector Machines (SVM) to predict gap acceptance/rejection for uncontrolled intersections and midblock crossing and found that it either outperforms the Binary Logit Models (BLM) in terms of the prediction for some data sets or predicts at least as accurately as those using BLM.

In machine learning, support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. The original SVM algorithm was introduced by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. SVMs constructs a hyperplane or set of hyperplanes in a high or multi-dimensional space, which can be used for regression and classification. The term Decision Trees (DTs) encompasses a set of non-parametric supervised learning methods used for classification and regression. They predict the value of a target variable by learning simple decision rules inferred from the data features. Their ability to model complex or high-order interactions among independent variables and the ease of interpretation has made DTs desirable for the task of classification. Classification and Regression Tree (CART), introduced by Breiman et al.[7] is one of the most commonly used binary decision tree algorithms and has been applied in business administration, medicine, industry, and engineering fields. Moreover, CART decision tree algorithm has been widely used to analyze crash severity in accidents. Pakgohar et al.[8] applied CART and Multinomial Logistic Regression(MLR) to study the role played by human factors in crash severity and found that results from CART were more accurate than MLR. They also stated that results obtained from CART are simpler to understand and interpret. CART was also used by Kashani et al.[9] to identify the most important features that contribute to crash severity of accidents in Iran rural roads. However, decision trees grown very deep tend to overfit to the training data i.e. they have low bias and high variance. Random Forests, introduced by Breiman et al.[10], are an ensemble learning method for classification and regression that address this issue by averaging multiple deep decision trees, trained on different parts of the same training set. This algorithm also has been widely used in analyzing traffic accidents. Rami Harb et al.[11] used random forests to understand the various attributes of human, vehicle and environment that result in crash evasive actions.

The objective of this study is to evaluate the application of SVM, DT and RF techniques for classifying and predicting gap acceptance decisions. We collected drives' gap acceptance data at four un-signalized intersections for analysis. SVM, DT and RF models were developed to predict gap acceptance behavior of the driver.

2. Background

2.1. Support Vector Machines

Support vector machine is a supervised machine learning algorithm that is widely used for classification. SVM tackles the task of classification by finding an optimal hyperplane that separates the two classes. While multiple solutions may exist that can completely classify the training data, SVM chooses the decision boundary that minimizes the generalization error by selecting the hyperplane that provides maximum separation or margin between the classes. Margin is defined as the perpendicular distance between the decision boundary and the closest of the data points which are called support vectors. The parameters of the linear classification model of the form, $f(x) = w.x+b$, can thus be obtained by minimizing the following objective function.

$$\underset{\mathrm{w,b}}{\arg max}\left\{\frac{1}{||W||}\underset{\mathrm{n}}{min}[t_n(W^T.X_n+b)]\right\} \tag{1}$$

Where t_n is either +1 or -1 depending on the class to which the data point belongs and X_n is the corresponding feature vector. In order to account for noisy data, a slack variable (ζ_n) is introduced for each training data point whose value is proportional to the distance of the data point from the margin. The tradeoff between the slack variable penalty and the margin can be controlled through the hyper parameter *C > 0*. By scaling W and b appropriately, the optimization can be expressed as follows:

$$\underset{\mathrm{w,b}}{\arg min}\left\{C\sum_{n=1}^{N}\zeta_n+\frac{1}{2}\,||W||^2\right\} \tag{2}$$

Subject to the constraints: $t_n(W^T.X_n+b) \geq 1$ n = 1, 2,.., N. and;

$\zeta_n > 0$

In the cases where the data is not linearly separable, the kernel trick is applied to transform the feature space to a higher dimension space where the data is linearly separable. The decision boundary can be expressed in terms of Lagrange multiplier $\{a_n\}$ and the kernel function $K(x,x')$ as follows:

$$y(x)=\sum_{n=1}^{N}a_nt_nK(x,x')+b \tag{3}$$

One common example of kernel function is the Gaussian radial basis function given by:

$$yK(x_i,x_j)=\exp\left(-\frac{||x_i-x_j||^2}{2\delta^2}\right) \tag{4}$$

Where, δ^2 is the bandwidth of the kernel. In the present study, since the classes were overlapping, we used a soft-margin SVM with RBF kernel.

2.2. Decision Tree

Decision tree (DT) is a predictive model that can be used for both classification and regression. They can be represented graphically as hierarchical structures making them easy to interpret. In a DT, each node represents an attribute variable and each branch represents the state of this variable. The terminal node or the leaf node specifies the expected class of the data point. In order to make the decision about a new data point, it is made to transverse the tree starting from the root node until it falls into one of the leaf nodes. The class denoted by the leaf node denotes the predicted class of the data point.

For the construction of decision tree, a greedy approach is adopted in which trees are built in a top-down recursive divide-and-conquer manner. The training set is recursively partitioned into smaller subsets as the tree is being built. Splitting into smaller subsets is continued until the purity of the subset can no longer be increased. CART uses GINI index of diversity as the measure of impurity of the node or diversity of classes in the node. For all input variables, splitting is performed by searching over all possible threshold values of splitting points so as to find the maximum threshold value, which changes the impurity of the resultant nodes. For a node $\boldsymbol{t}$, GINI index is defined as follows:

$$GINI(t)=1-\sum_{i=0}^{c-1}[p(i/t)]^2 \tag{5}$$

where $p(i/t)$ denotes the fraction of observations in node t that belong to class *i,* and *c* denotes the total number of classes.

Since the maximal binary tree thus grown is prone to overfit, pruning is performed. In CART, cost-complexity pruning algorithm is used for post-pruning. In this approach, the cost complexity of the tree is considered as a function of number of leaves in the tree and its error rate which is computed as the percentage of tuples misclassified by the tree. Starting from the bottom, for each internal node N, cost complexity of the subtree at N, and the cost complexity of the subtree at N if it were replaced by a leaf node are calculated. If there is a decrease in cost-complexity after pruning, then the subtree is pruned. A pruning set of class-labelled tuples, which is independent of the training set used to build the unpruned tree, is used to estimate cost complexity. From the set of progressively pruned trees generated by the algorithm, the smallest decision tree that minimizes the cost complexity is chosen.

2.3. Random Forests

Random forests are an ensemble learning method widely used for classification and regression tasks. This technique combines Brieman's bagging idea and Ho's "random subspace method" to construct a collection of decision trees with controlled variations. By building a multitude of weak decision tree classifiers in parallel and then combining them to form a single, strong learner by averaging their individual predictions, Random forests correct for the decision trees' habit of overfitting to their training sets. The pseudocode of the algorithm is shown below.

The algorithm works as follows: For each tree in the forest, a bootstrap sample is selected from the original data. The bootstrapped sample is obtained by randomly selecting instances from the original data with replacement and is of the same size as the original data set. A decision tree is then grown to the maximum extent possible without pruning on the bootstrapped sample using a modified decision-tree learning algorithm. The tree-learning algorithm is modified as follows: At each node, best split is selected by examining a random subset of features rather than the complete feature set. Since deciding the best-split is the most computationally expensive aspect of the learning process, choosing a subset of features will drastically speed up the learning of the tree. Once all the trees are constructed this way, final predictions are obtained by averaging individual predictions of the trees.

Algorithm: Random Forest

Precondition: A training set ***S*** : = $(x_1, x_2), \ldots, (x_1, x_2),$ features ***F***, and number of trees in forest ***B.***

```
function RANDOMFOREST (S, F)
  H ← θ
  for i ϵ , …, B do
      $S^{(i)}$ ← A bootstrap sample from S
      $h_i$ ← RANDOMTREELEARN ($S^{(i)}$, F)
      H ← H ∪ {$h_i$}
  end for
  return H
end function
function RANDOMIZEDTREELEARN
  At each node:
      f ← very small subset of F
      Split on best feature in f
  return The learned tree
end function
```

By using the bagging technique to build an ensemble of decision trees, random forests achieve lower variance. However, in traditional bagging, constituent trees may end up being highly correlated as same features tend to be used repeatedly to split the bootstrap samples. By modifying the tree learning algorithm to choose features from random subsets, the correlation between the trees comprising the ensemble is reduced. This way, random forests tend to achieve superior performance**.**

3. Data Description

Data collected from three four-legged un-signalized intersections was used for the analysis. Data collection focused on recording the behavior of drivers intending to cross the major road and drivers on the major road approaching the intersection. The selected 4-legged intersections were located on inner arterial roads with posted speed of 40 km/hr. The data set consisted of a total of 1234 observations. The feature set consists of two numerical variables describing the speed and distance of the approaching vehicle and a categorical variable describing the nature of the subject vehicle and the approaching vehicle.

Figure 1 shows the plot of spatial gaps and the associated approach speed of the respective vehicles on the main line stream for a typical 4 legged intersection. The accepted gaps and rejected gaps are distinguished in the Figure.

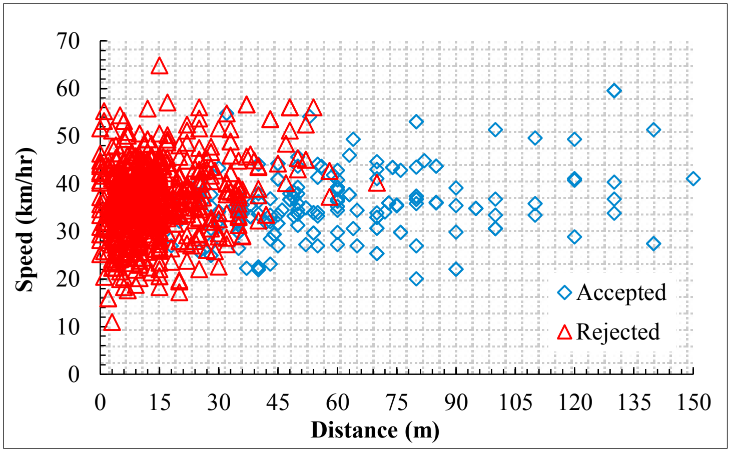

Fig. 1. Observed trajectories of main line stream vehicles while acceptance and rejection by minor road vehicle

4. Methodology

We modeled the gap acceptance behavior of the driver to predict whether the gap would be accepted or rejected. Three features, namely, the speed of the approaching vehicle, the distance between the approaching vehicle and subjective vehicle and a variable encoding information about the type of subject and through vehicle are used for building the models. A data classification task typically involves two steps, training the model on the available data and validating it by predicting the categorical labels of unseen data using the model. To replicate this process, 80% of the samples were used for training and the remaining 20% were used for validation. For the analysis of 4-legged intersections, we used 987 gap/lag values for training and 247 gap/lag values for validation. Based on the data collected, the performances of Support Vector Machine, Decision tree and Random forest models are evaluated and compared.

SVM and Random forest models were implemented using the python programming language whereas decision tree algorithm has been implemented using R-Project. The optimum value for the parameters required in the construction of random forest and SVM models were selected by performing a grid search over all possible combinations of parameter values. Ten-fold cross-validation technique was employed.

In order to evaluate the performance of the three gap acceptance prediction algorithms, we compute bias, probability of detection, probability of non-accepted detection and accuracy based on a confusion matrix (Table 1):

Table 1. Confusion Matrix.

	Observed Accept	**Observed Reject**
Predicted Accept	Hits(h)	False Alarms(f)
Predicted Reject	Misses(m)	Correct Negatives(z)

- Bias: It is the ratio of number of predicted accepted gaps to the total number of observed accepted gaps. It indicates whether the classifier underestimates or overestimates the number of accepted gaps.

$$Bias = \frac{f+h}{m+h} \tag{6}$$

- POD: It is the ratio of the number of correctly predicted accepted gaps to the total number of observed accepted gaps. It gives the fraction of observed rejected gaps that are correctly forecasted.

$$POD = \frac{h}{m+h} \tag{7}$$

- POND: It is the ratio of the number of correctly predicted rejected gaps to the total number of observed rejected gap values. It indicates the fraction of observed rejected gaps that are correctly detected.

$$POND = \frac{z}{z+f} \tag{8}$$

5. Results and Discussion

5.1. Decision Tree Model

A decision tree is constructed employing the CART algorithm discussed in Section 2.2 as shown in Figure 2. A rule can be obtained for each node by tracing its path from the root node. Each splitting criterion along a given path can be combined through the logical "AND" operation to form the rule antecedent ("IF" part) and the majority class forms the consequent ("THEN" part). Each of these rules can be assessed by their coverage and accuracy. While coverage is the percentage of tuples that are covered by the rule (i.e., their attribute values hold true for the rule's antecedent), accuracy considers the tuples that it covers and indicates what percentage of them the rule can correctly classify. In figure 2, for each node, the feature selected to split the node further, accuracy and coverage of the rule represented by the node are mentioned.

The splitting procedure is as follows: The initial splitting of node 1 is based on the spatial gap. Instances, where spatial gap is less than 34 m go to the right, forming node 3 while the others form node 2 in the left. Though the majority decision in node 3 and node 2 is to reject and accept the gap respectively, the majority is not sufficient to take the decision. Hence the nodes 2 and 3 are further split on the spatial gap criteria to form terminal nodes 4 and 7, and intermediate nodes 5 and 6. The terminal nodes 4 and 7 respectively indicate that the driver makes the decision to accept the gap when the distance is greater than 58m with an accuracy of 97% and decides to reject the gap when the distance is less than 24 m with an accuracy of 98%. The terminal nodes 4 and 7 cover 9% and 67% of the data points respectively. For cases where the distance of approaching vehicle is in the ambiguous range of 24 m to 58 m, the algorithm selects speed of the through vehicle as the criteria to take the decision. With speed of 45 km/hr. and 26 km/hr as the chosen split for nodes 4 and 5 respectively, terminal nodes 8, 9, 10 and 11 are formed. For vehicles at a distance of 34 m to 58 m, the gap is accepted if their speed is less than 45 km/hr. or else, the gap is rejected. On the other hand, for vehicles at a distance of 24 m – 34 m, the gap is not accepted unless the speed is less than 26 km/hr.

Ease of interpretability of decision trees helps us understand the reasoning behind the driver's decision to accept or reject the gap. Further, splitting the data using the most informative feature first, provides an insight into the relative importance of the features. Spatial Gap being chosen as the basis for the first split indicates that it may have the most significant effect on the driver's decision to accept or reject the gap. This finding also makes intuitive sense as the distance of approaching vehicle can be more easily perceived than the speed. Further, the feature obtained by joining the type of subject vehicle and through vehicle did not even appear in tree post pruning indicating that nature of the vehicle played little role in influencing the driver's' decision.

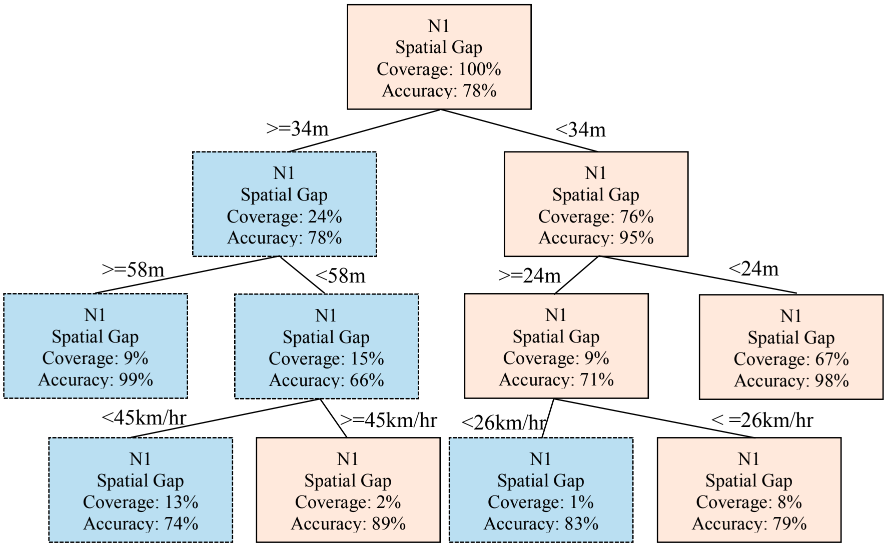

Fig. 2. Decision tree for gap acceptance

5.2. Comparison of SVM, Decision tree and Random Forest

From the skill scores for SVM, Decision Tree and Random Forest (see Table 2), it is evident that all the models perform reasonably well. The overall accuracy is highest for Random Forest followed by SVM and Decision Tree indicating that Random Forest model is the most precise. The difference in accuracy score between SVM and DT is not very significant. Probability of detection (POD) and Probability of non-detection (POND) also follow a similar trend indicating that Random forests perform also perform the task of accurately predicting accepted gaps and rejected gaps better than SVM and DT. However, it must be noted that the bias score for SVM and Decision Tree models is closer to one than for Random Forest model.

Table 2. Skill Scores for SVM, DT and Random Forest.

	SVM	**Decision Tree**	**Random Forest**
Bias	0.97	0.97	0.94
POND	0.97	0.95	0.98
POD	0.86	0.81	0.90
Accuracy	0.94	0.93	0.97

6. Summary and Conclusions

This study evaluates and compares the performance of SVM, Random forests and Decision tree algorithm to classify and predict driver's gap acceptance/rejection decision at uncontrolled intersections. Data at three 4-legged intersections was collected and used to develop the models. Since all three are non-parametric techniques, no assumption is needed to be made on the underlying data distribution. Skill scores such as Bias, POD, POND and Accuracy were used for performance evaluation. Study results were found to be promising.

While SVMs seem to perform better than Decision Trees the difference is not significant. Since SVMs are insensitive to class imbalance in the training data, they are expected to perform better than other models. However, SVMs also have some disadvantages over Decision trees. First, Decision tree and Random Forest models provide information about the relative importance of variables, SVM cannot give this information. The inclusion of insignificant variable may result in over-fitting. Second, for large datasets, solving the quadratic program during the training procedure becomes difficult with standard QP solvers[15]. Further, as seen in Section 5.1, decision tree models can be used to determine interactions between variables and useful decision rules can be inferred. This can prove to be useful, especially when dealing with a large number of features as they perform implicit feature selection.

While all the three algorithms perform reasonably well, Random Forest technique outperforms SVM and Decision tree model in terms of prediction. Further, Random Forests are faster to train, have fewer parameters to be tuned and can handle a large number of predictors without requiring any variable selection. These advantages of Random forests and better performance as observed from our results indicate that Random Forest algorithm has very good potential to be an alternative tool for the gap acceptance/rejection predictions.

Precise prediction of gap acceptance at uncontrolled road sections is of great importance in developing real-time applications such as Advanced Warning and Safety System (AWSS) and Advanced Traffic Management Systems (ATMS). Such systems help drivers choose an appropriate course of action at un-signalised intersections. Future studies should apply the Random forest technique to the gap acceptance data from different cities and check the applicability of the models developed.

References

1. Polus, A., Lazar, S. S., & Livneh, M. (2003). Critical gap as a function of waiting time in determining roundabout capacity. Journal of Transportation Engineering, 129(5), 504-509.
2. Raff, M. S. and Hart, J. W. (1950). “A volume warrant for urban stop signs.” Eno Foundation for highway traffic control, Saugatuck, Connecticut.
3. Hewitt, R. H. (1983). “Measuring critical gap.” Transportation Science, 17(1), 87–109.
4. Hamed, M. M., Easa, S. M., and Batayneh, R. R. (1997). “Disaggregate gap-acceptance model for unsignalized T-intersections.” Journal of Transportation Engineering, 123(1), 36-42.
5. Pant, P. D., and Balakrishnan, P. (1994). “Neural network for gap acceptance at stop-controlled intersections.” Journal of Transportation Engineering, 120(3), 432–446.
6. Pawar, D., Patil, G. R., Chandrasekharan, A., and Upadhyaya, S. (2015). Classification of gaps at uncontrolled intersections and midblock crossings using Support Vector Machines. Transportation Research Record: Journal of the Transportation Research Board. 2515, 26-33.
7. Breiman, Leo, et al. Classification and regression trees. CRC press, 1984.
8. Pakgohar, A., Tabrizi, R.S., Khalilli, M., Esmaeili, A. (2010). “The role of human factor in incidence and severity of road crashes based on the CART and LR regression: a data mining approach.” Procedia Computer Science 3, 764-769.
9. Kashani, A., Mohaymany, A., Ranjbari, A. (2011). “A Data Mining Approach to Identify Key Factors of Traffic Injury Severity.” Promet-Traffic & Transportation, 23 (1), 11-17.
10. Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.
11. Harb, R., Yan, X., Radwan, E., & Su, X. (2009). Exploring precrash maneuvers using classification trees and random forests. Accident Analysis & Prevention, 41(1), 98-107.
12. Vapnik, V. N. (1998). Statistical learning theory (Vol. 2). New York: Wiley, 2.
13. Burges, C. J. C., (1998). “A tutorial on support vector machines for pattern recognition.” Data Mining and Knowledge Discovery, 2 (2), 121–167.
14. Scholkopf, B., and Smola, A. J. (2001). “Learning with kernels: support vector machines, regularization, optimization, and beyond.” MIT press.
15. Zhang, Y., and Xie, Y. (2008). “Forecasting of short-term freeway volume with v-support vector machines.” Transportation Research Record: Journal of the Transportation Research Board, 2024(1), 92-99.