Freight transport prediction using electronic waybills and machine learning

Shoaib Bakhtyar
Department of Computer Science & Engineering
Blekinge Institute of Technology
Karlskrona, Sweden
shoaib.bakhtyar@bth.se

Lawrence Henesey
Department of Computer Science & Engineering
Blekinge Institute of Technology
Karlskrona, Sweden
larry.henesey@bth.se

***Abstract*— A waybill is a document that accompanies the freight during transportation. The document contains essential information such as, origin and destination of the freight, involved actors, and the type of freight being transported. We believe, the information from a waybill, when presented in an electronic format, can be utilized for building knowledge about the freight movement. The knowledge may be helpful for decision makers, e.g., freight transport companies and public authorities. In this paper, the results from a study of a Swedish transport company are presented using order data from a customer ordering database, which is, to a larger extent, similar to the information present in paper waybills. We have used the order data for predicting the type of freight moving between a particular origin and destination. Additionally, we have evaluated a number of different machine learning algorithms based on their prediction performances. The evaluation was based on their weighted average true-positive and false-positive rate, weighted average area under the curve, and weighted average recall values. We conclude, from the results, that the data from a waybill, when available in an electronic format, can be used to improve knowledge about freight transport. Additionally, we conclude that among the algorithms IBk, SMO, and LMT, IBk performed better by predicting the highest number of classes with higher weighted average values for true-positive and false-positive, and recall.**

Keywords—machine learning; Waybill; freight mobility; IBk; SMO; LMT

I. Introduction

In freight transport, different types of documents accompany a consignment. A waybill contains essential information about a consignment. The information includes origin and destination of the consignment, information about the involved actors, and information about the type of freight. We believe, the information present in a paper waybill, if available in an electronic format, can be utilized in several ways in order to achieve greater benefits. For example, an electronic waybill (e-Waybill) can utilize synergies (i.e., share information) with different Intelligent Transportation Systems (ITS) services [1], which can lead to reduced implementation cost of the services [2]. In this paper, we argue that the e-Waybill information may be utilized for building knowledge about the freight movement between a particular origin and destination city.

Predicting the type of freight to be transported (from an origin to a destination) may be helpful in deciding the type of vehicle to be used for transportation, e.g., today reefers (i.e., refrigerated trucks) are used in case of frozen freight. For transport companies, having advanced information about the freight type may be helpful in deciding on the type of transport to use (provided that the origin and destination cities of an oncoming order are known). The information may also help the transport companies in demand forecasting for a particular type of freight. For public authorities, predicting the freight type (moving between different cities) may be helpful in ensuring that the infrastructure (between the freight's origin and destination city) is capable of meeting the actual freight to be transported, e.g., the road choice in case of dangerous goods.

The main purpose of this paper is to predict the type of freight that is moving between a particular origin and destination. To achieve the purpose, we have used a dataset that contains data from a customer ordering database of a Swedish transportation company. It must be noted here that the collected data (i.e., from a database) is, to a greater extent, similar to the data present in a paper waybill. Therefore, we can assume that the data present in the customer ordering database is, to a greater extent, similar to the data that can be provided by an e-Waybill. For predicting the freight type between cities, we have used the open-source tool Waikato Environment for Knowledge Analysis (WEKA). WEKA provides a collection of machine learning algorithms (in a graphical user interface) that can be applied on a dataset. In addition to the main purpose, the secondary purpose of this paper is to evaluate the different machine learning algorithms, which are applied on our dataset for predicting the freight type.

The paper is structured as follows: in Section II, we describe the methodology for conducting the study, in Section III, we present and explain the data pre-processing step, which was needed in order to remove missing values from the dataset, in Section IV, we present the preliminary experiment step, where different classification algorithms were applied on the dataset and the best-performing algorithms were selected. In Section V, we describe the final experiment, which was conducted to evaluate the best three performing algorithms from the preliminary experiment step. In Section VI, we present a discussion of the results and analysis of the final experiment and finally in Section VII, we discuss the concluding remarks to the paper.

II. Methodology

The methodology followed in this paper includes: data collection, data pre-processing, and computational experiments that were conducted in WEKA. For conducting the experiments, we used a dataset that included data from a customer ordering database of a transport company was used. We employed the case study method in evaluating the collected customer order data from the transport company located in Karlshamn, Sweden.

The order data consisted of 243 orders, which were completed during a period of one month. The data included freight origin, destination, weight, and type. In order to ensure confidentiality of the company and the order data, the cities names and the freight types have been replaced with unique identifiers in this paper. For each completed order, there exists a paper waybill. Therefore, to ensure that the collected data is similar to the data present in a paper waybill, the order data was compared with the corresponding data from the waybills.

To overcome the problem of missing values, we performed a data pre-processing step. Data imputation was used for replacing the missing values with realistic values. After conducting the pre-processing step, our dataset was complete and ready for experiments. We used the tool WEKA for applying different machine learning algorithms on the dataset. The main purpose was to predict the freight type based on the origin, destination, and weight of the freight. Therefore, we used classification algorithms in the experiments. The experiments were conducted in two phases. In phase 1 (preliminary experiment), we applied different classification algorithms on the dataset in order to identify the best-performing algorithms. Based on the results from the preliminary experiment, a final experiment was conducted in phase 2. In the final experiment, we selected the top three best-performing algorithms for a comparison and evaluation.

III. Data pre-processing

The dataset had four attributes, i.e., origin city, destination city, weight of the freight, and the freight type. Of the four attributes, the freight type is the class attribute. In summary, the dataset had the following characteristics:

- Total No. of attributes = 4
- All attributes = {Origin City, Destination City, Weight, Freight Type}
- Class attribute = Freight Type
- Total No of class attributes = 17
- Class attributes = {G1, G2, G3, G4, G5, G6, G7, G8, G9, G10, G11, G12, G13, G14, G15, G16, G17}
- Total no. of instances = 243

The 243 instances had 68 instances with missing values in the class attribute. In a dataset with only 243 instances, we considered this number of missing values to be high, and therefore, it was necessary to impute the missing values. To impute the missing values in a dataset, there exist different methods. In the literature, there exist different data imputation techniques such as, the following: ignoring instances with unknown feature values, most common feature value in class, mean substitution, regression or classification method, hot-deck or matching imputing, and treating missing feature values as special values [5]. In our dataset, we could observe patterns between the instances with missing values. Therefore, we used the hot-deck imputation method. In this method, missing values are imputed through a process in which for an instance "x" with a missing value "y" an instance with similar values of "x" is observed and the data present in place of "y" is selected and imputed against the missing value.

In our dataset, all the missing values belonged to the class attribute (i.e., the freight type). Additionally, most of the missing values belonged to instances where the origin and destination cities were similar. The imputation was done using three steps approach: first, we observed an instance with a missing value at the origin and destination cities. Second, the entire dataset was searched for origin and destination cities that matched with the origin and destination of the missing value instance. Thirdly, if a match was found, the freight type value was taken from there and imputed against the missing value. There were cases where multiple matches were found for the same origin and destination cities with different freight type values. In such cases, the most frequently occurring freight type value was considered and imputed against the missing value. After all the missing values were replaced in the dataset, it was now ready for the experiments.

IV. Preliminary experiment

In our preliminary experiment, different classification algorithms were used on the dataset for predicting the type of freight moving between different cities. Before applying the algorithms in WEKA, we selected 10 folds cross-validation in the testing options since the dataset had 243 instances only and 10 folds cross-validation is considered to be useful on a dataset that is very small to be partitioned into separate training and testing [6]. In the 10 folds cross-validation, a dataset is partitioned into 10 sets of equal sizes. The algorithm then trains on 9 datasets and tests on 1 of the dataset. This step of training and testing is repeated 10 times, and in the end mean accuracy of the tests is calculated [6].

Once the experiment was conducted, we selected the best-performing algorithms based on their accuracy, i.e., the number of instances correctly classified by a particular algorithm. The algorithms that performed with higher accuracy were Averaged one-dependence estimators (AODE), Sequential minimal optimization (SMO), k-Nearest Neighbors (IBk), LogitBoost, JRIP, Logistic Model Trees (LMT), and HyperPipes.

- AODE algorithm is a probabilistic classification learning algorithm. It is considered to be the most effective Naive Bayes algorithm due to its focus on addressing the attribute-independence problem of the popular Naive Bayes algorithm [7]. AODE has been developed for the purpose of improving the accuracy of Naive Bayes [8].
- SMO was proposed for training support vector machines with a high speed. Support vector machines consist of a set of supervised learning models for classification and regression problems [9]. Kernal

functions, such as polynomial or Gaussian are used by SMO in order to implement the sequential minimal-optimization algorithm for training support vector machines [10][11].

- IBk is an implementation of the k-nearest-neighbour classifier. A variety of different search algorithms can be used to speed up the task of finding the nearest neighbors. In IBk, predictions from more than one neighbor can be weighted according to their distance from the test instance, and two different formulas are implemented for converting the distance into a weight [12][6].
- LogitBoost is a boosting algorithm. In a boosting algorithm, a number of iterations are run over the data in order to find the simple regression function with the smallest error. The algorithm can be iterated until convergence. However, for optimal performance it is unnecessary to wait for convergence. This can be achieved by determining the appropriate number of boosting iterations through the expected performance measures until the performance stops to increase. The performance for a given number of iterations can be calculated using cross-validation [6][13].
- JRip is an implementation of the popular RIPPER (repeated incremental pruning to produce error reduction) algorithm [6]. It is a rules-based learner that determines propositional rules, which can be used to classify elements [14].
- LMT is a supervised learning algorithm that combines decision trees and logistic regression. The algorithm uses LogitBoost algorithm in order to induce trees with linear-logistic regression models at the leaves [6].
- HyperPipes is considered to be a very simple algorithm, and it is used in discrete classification problems [15]. For each attribute in a training data, the algorithm records the range of values and calculates the ranges containing the attribute values of a test instance. The algorithm then chooses the category with the largest number of correct ranges [6].

A. *Preliminary Experiment Results*

The results of the preliminary experiment suggested that the best accuracy was by the algorithm IBk, and the algorithm with lowest accuracy was HyperPipes. To categorize the algorithms based on the results, the top three algorithms (based on high accuracy) were IBk, SMO, and LMT with an accuracy of more than 80%. The algorithms with accuracy between 70-80% were AODE, LogitBoost, and JRIP. The algorithm HiperPipes had the lowest accuracy of 65.02%. A summary of the results is presented in Table I.

TABLE I. ALGORITHMS WITH HIGH ACCURACY

Algorithm	Accuracy
AODE	79.83%
SMO	81.07%
IBk	82.72%
LogitBoost	79.83%
JRIP	75.30%
LMT	80.24%
HyperPipes	65.02%

We selected the algorithms SMO, IBk, and LMT in phase 2 for the final experiment. The remaining algorithms, i.e., AODE, LogitBoost, JRIP, and HyperPipes, were analyzed based on their confusion matrices in order to identify the correctly classified classes.

All the three algorithms (except HyperPipes) were able to correctly classify the classes; G1, G3, G6, and G16. The HyperPipes algorithm was able to correctly classify the class G8 in addition to G1, G3, G6, and G16. The AODE algorithm was able to correctly classify 129 out of 138 instances for G1, 43 out of 53 instances for G3, 19 out of 21 instances for G6, and 3 out of 4 instances for6 G16. The LogitBoost and AODE algorithms produced similar results concerning G6 and G16 by correctly classifying the same number of instances. LogitBoost was able to correctly classify 132 out of 138 instances for G1 and 40 out of 53 instances for G3. Whereas, HyperPipes was able to correctly classify 15 out of 53 instances for G3, 6 out of 21 instances for G6, and 2 out of 3 instances for G8. For G1, the HyperPipes algorithm predicted the same number of instances as was predicted by LogitBoost. For G16, HyperPipes, LogitBoost, and AODE were able to correctly classify the same number of instances, i.e., 3 out of 4 instances. The algorithm JRip was able to correctly classify 124 out of 138 instances for G1, 38 out of 53 instances for G3, 17 out of 21 instances for G6, and 4 instances out of G16. Hence, the confusion matrices of the algorithms indicate that AODE, and LogitBoost performed equivalently by correctly classifying 194 out of 243 instances. The algorithm JRip was able to correctly classify 183 instances, while HyperPipes was able to correctly classify 158 instances out of the total 243. Thus, the HyperPipes algorithm correctly classified the lowest number of instances. However, HyperPipes was able to correctly classify more classes as compared to AODE, LogitBoost, and JRip.

V. FINAL EXPERIMENT

For the final experiment, we used the Experimenter application in WEKA. The same dataset (from the preliminary experiment) was used in our final experiment. In the experiment type, 10 folds cross-validation was selected, which is the same testing option from the preliminary experiment. We present the results from the preliminary experiment in Table I. We observe that the algorithms IBk, SMO, and FT have higher accuracy (i.e., more than 80%) as compared with the rest of the algorithms. Therefore, in the final experiment, we compared and evaluated the three algorithms based on their weighted average values for *true-positive* (TP) and *false- positive* (FP) rate, *area under the curve* (AUC), and *recall.*

TP rate is the number of correct classifications made by an algorithm, whereas FP rate is the number of incorrectly classified instances, i.e., predicting negative instances as positive [6]. In the final experiment, we selected weighted average TP and FP rate in order to calculate the correct and incorrect classification rates of all the classes by the three algorithms.

AUC is the probability of randomly choosing and ranking a positive instance above a randomly chosen negative instance in a test data. The probability is based on the rankings by the classifier. In the best case, AUC is 1 and all the positive instances are ranked above all the negative instances. In the worst case, AUC is 0 and all the positive instances are ranked below the negative instances. In the case of random rankings, AUC is 0.5. Anti-learning is expected to have been performed by the classifier if the AUC is significantly less than 0.5 [6]. We selected AUC as a performance metric since the literature (see, e.g., [3] and [4]) suggests that AUC measure is better (as compared to accuracy) when comparing the performance different classification algorithms.

Recall can be defined as a ratio of the total number of instances classified correctly by the algorithm to the total number of actual instances [6, 16]. In the final experiment, we selected recall for evaluation of the algorithms since it can help in calculating the correctly classified instances as compared to the actual instances in the dataset.

VI. Results and Analysis

We present results from the final experiment in Table II.

TABLE II. ALGORITHMS EVALUATION

Evaluation Criteria	Algorithms		
	SMO	IBk	LMT
Weighted average TP rate	0.80	0.82	0.79
Weighted average FP Rate	0.12	0.11	0.14
Weighted average AUC	0.87	0.89	0.91
Weighted average recall	0.80	0.82	0.79

We can observe from the results that IBk has a higher weighted average TP rate than SMO and LMT, i.e., IBk could classify 82% of the instance correctly as compared to 80% and 79% by SMO and LMT respectively. Regarding incorrectly classified instances (i.e., weighted average FP rate), IBk (with the FP rate of 11%) outperformed SMO and LMT, which had weighted average FP rate of 12%, and 14% respectively. By analyzing the confusion matrices of the three algorithms, we observed that IBk performed better as compared to SMO and LMT by correctly classifying 6 out of the 17 classes. However, SMO and LMT correctly classified 5 out of the total 17 classes. The classified classes by each of the algorithms are presented in Table III.

TABLE III. CLASSIFICATION OF CLASSES

Algorithms	Correctly classified	Incorrectly classified
SMO	G1, G3, G6, G16	G2, G4, G5, G7, G8, G9, G10, G11, G12, G13, G14, G15, G17
IBk	G1, G3, G4, G5, G6, G16	G2, G7, G8, G9, G10, G11, G12, G13, G14, G15, G17
LMT	G1, G3, G5, G6, G16	G2, G4, G7, G8, G9, G10, G11, G12, G13, G14, G15, G17

From Table II, we can observe that the AUC of LMT is closer to 1 as compared to the AUC of SMO and IBk. This indicates that in most of the cases, the positive instances were ranked above the negative instances by LMT. Therefore, based on AUC, the algorithms LMT performed better than SMO and IBk. The weighted average recall value of IBk is higher than SMO and LMT. IBk has a weighted average recall value of 82%, which means 199 out of 243 instances were correctly classified by IBk. SMO, with 80% of weighted average recall value, was able to correctly classify 194 out of 243 instances. LMT, with a lowest weighted average recall value of 79%, was able to correctly classify 192 instances out of the total 243 instances.

In Figure 1, 2, 3, and 4, we present a comparison of the TP and FP rate, Recall, and AUC by IBk, SMO, and LMT for each of the class attributes. The TP rate comparison, in Figure 1, of the algorithms indicates that IBk performed better than SMO and LMT for the classes G1, G3, G4, and G5. The three algorithms had similar performance concerning the class G16, whereas SMO and LMT performed better than IBk.

In Figure 2, a comparison of the FP rate of the three algorithms indicates that the algorithms performed equally well for the classes G6, G8, G10, and G13. However, for the classes G1 and G3, IBk performed better than both SMO and LMT. The performance of SMO and IBK was equally better than LMT for the class G3. For the class G1, SMO and LMT had equally lower performance than IBk.

In Figure 3, we present a comparison of the three algorithms concerning recall values. It can be observed that IBk performed better than SMO and LMT for the class G1, G4, and G5. For the class G3, IBk has better performance than LMT but has an equivalent performance as SMO. For G6, IBk has lower performance than SMO and LMT, while the performance of all the three algorithms is the same for G16.

AUC comparison of the three algorithms is shown in Figure 4. The figure indicates that IBk performed better than SMO and LMT for the classes G6, G11, and G14. Whereas, for the classes G4 and G17, IBk lower performance than SMO and LMT. For the classes G1, G3, G7, G9, G11, G12, and G15, IBk performed higher than SMO but lower than LMT. For the classes G8, G10, and G13, IBk and SMO both performed equally higher than LMT, while IBk and LMT performed equally higher than SMO for the class G16. For the class G2, SMO performed better than IBk. However, the algorithm IBk may have performed anti-learning as the AUC is lesser than 5%.

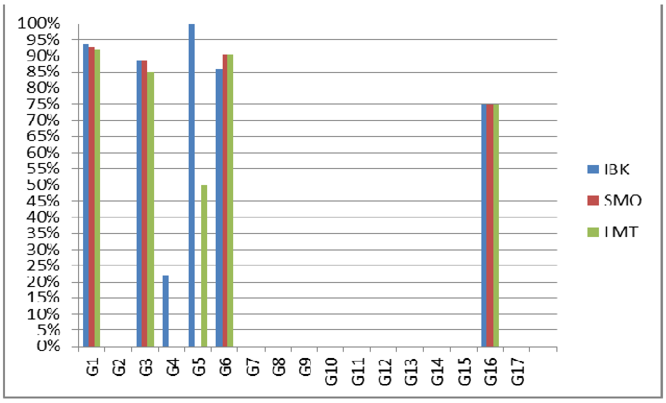

Fig. 1. True Positive rate by IBK, SMO, and LMT.

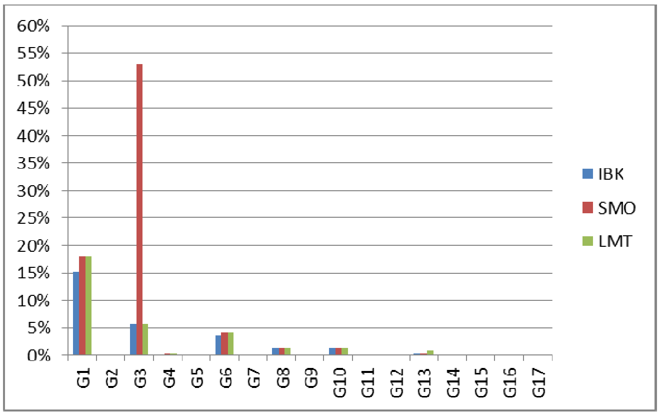

Fig. 2. False Positive rate by IBK, SMO, and LMT.

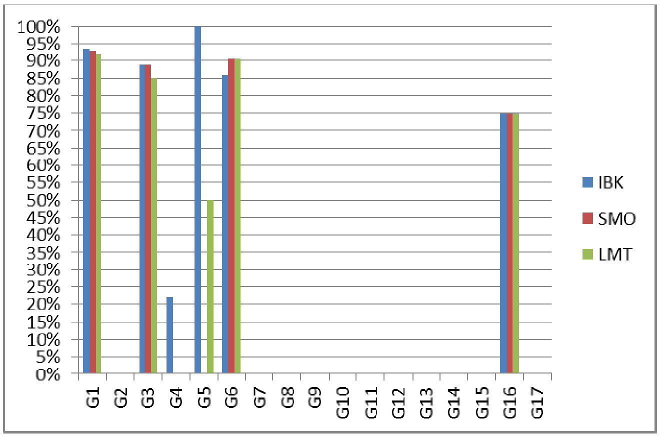

Fig. 3. Recall values by IBK, SMO, and LMT.

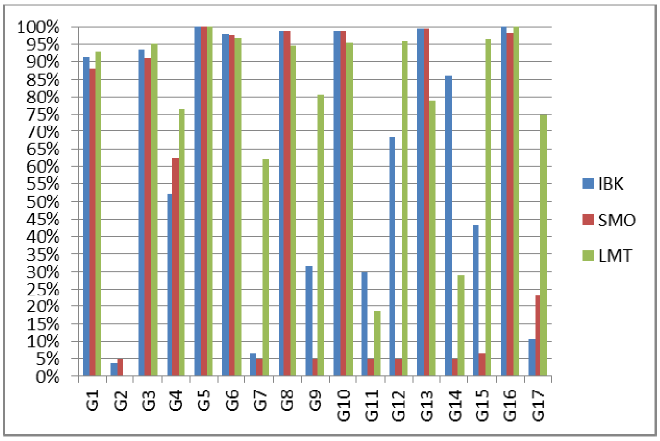

Fig. 4. AUC by IBK, SMO, and LMT.

VII. CONCLUSION

The main purpose of this paper was to use machine learning algorithms for predicting the type of freight transported between different origin and destination cities. Based upon a dataset obtained from a Swedish transport company, we applied different machine learning algorithms on the dataset (in order to predict the type of freight transported). In addition, we selected the top three best-performing algorithms for evaluation based upon their values for weighted average true-positive, false-positive, AUC, and recall.

We conclude that overall the algorithm IBk performed better than the algorithms SMO and LMT. Based on weighted averages of their TP and FP rate, and Recall value, the algorithm IBk can be considered better than SMO and LMT. Under the same criterion, SMO is at the second place, i.e.; its performance was lower than IBk but better than LMT. However, based on the weighted average AUC value, LMT performed better than IBk and SMO. IBk performed better, based on the weighted average AUC value, than SMO. Of the total 17 classes, the algorithm IBk was able to correctly classify 6 classes, while SMO and LMT both were able to correctly classify 4 and 5 classes each respectively. Thus, IBk performed better than SMO and LMT by correctly classifying more classes. All of the correctly classified classes by LMT and SMO were correctly classified by the IBk. However, a comparison of SMO and LMT indicates that LMT was able to correctly classify one more class, i.e., G5. The classes G2, G7, G8, G9, G10, G11, G12, G13, G14, G15, and G17 were not correctly classified by any of the three algorithms. A common feature among the incorrectly classified classes is that all of them have less than or equal to three instances. However, having less than or equal to 3 instances, may not be the reason for incorrect classification since the class G5, which has 2 instances, was correctly classified both by IBk and LMT. A lesson learned from the incorrectly classified classes is to group the different classes based on some common features. The grouping of classes, we believe, will not only reduce the number of classes, but it may also increase the classification accuracy.

Additionally, we conclude that the data from a waybill, if available in an electronic format, i.e., in the form of an e-Waybill, can help in improving knowledge about the freight movement by predicting the freight type (transported between a particular origin and destination city). In addition, we conclude that different machine algorithms perform differently and produce different results when used for predicting the freight type (based on the freight's origin, destination, and weight). Predicting the freight type transported from a particular origin to a particular destination may help transport companies in improved decision making about the type of transport required for a particular origin and destination city of a future order. Additionally, predicting the freight type may lead to improved decision making such as, investment decisions and policy making concerning the infrastructure (between freight's origin and destination), e.g., expanding road capacity or route choice in case of dangerous goods.

A possible limitation of this study is that multiple orders may be transported using a single vehicle and hence a vehicle may contain more than one type of freight. However, we believe that our results from this study can still be valid for the back-office level, i.e., by considering the order data in a database only and not focusing on the actual fulfillment of the orders by the vehicle. Potential future work to this paper would be to extend the features of the dataset in order to further strengthen the results achieved in this paper. The dataset can be extended with more attributes and features, such as the freight volume under transport and the routes travelled by a vehicle, which is used for transporting the freight, between a particular origin and destination.

REFERENCES

[1] S. Bakhtyar, J. Holmgren, and J. A. Persson, “Analysis of information synergy between e–Waybill solutions and intelligent transport system services,” World Review of Intermodal Transportation Research, vol. 4, no. 2–3, 2013.

[2] G. Mbiydzenyuy, J. A. Persson, and P. Davidsson, “Toward cost-efficient integration of telematic systems using K-spanning tree and clustering algorithms,” in 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2011, pp. 803–808.

[3] C. X. Ling, J. Huang, and H. Zhang, “AUC: a Better Measure than Accuracy In Comparing Learning Algorithms,” in IN PROC. OF IJCAI’03, 2003, pp. 329–341.

[4] A. P. Bradley, “The use of the area under the ROC curve in the evaluation of machine learning algorithms,” Pattern Recognition, vol. 30, no. 7, pp. 1145–1159, 1997.

[5] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine Learning: A Review of Classification and Combining Techniques,” Artificial Intelligence Review, vol. 26, no. 3, pp. 159–190, 2006.

[6] I. H. Witten, E. Frank, M. A. Hall, and & 0 more, Data Mining: Practical Machine Learning Tools and Techniques, Third Edition, 3 edition. Burlington, MA: Morgan Kaufmann, 2011.

[7] J. Wu and C. Zhihua, “Learning Averaged One-dependence Estimators by Attribute Weighting,” Journal of Information and Computational Science, vol. 8, no. 7, pp. 1063–1073, 2011.

[8] L. Jiang and H. Zhang, “Lazy Averaged One-Dependence Estimators,” in Advances in Artificial Intelligence, L. Lamontagne and M. Marchand, Eds. Springer Berlin Heidelberg, 2006, pp. 515–525.

[9] A. Shmilovici, “Support Vector Machines,” in Data Mining and Knowledge Discovery Handbook, O. Maimon and L. Rokach, Eds. Springer US, 2005, pp. 257–276.

[10] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, “Improvements to Platt’s SMO Algorithm for SVM Classifier Design,” Neural Computation, vol. 13, no. 3, pp. 637–649, 2001.

[11] T. Hastie and R. Tibshirani, “Classification by pairwise coupling,” The Annals of Statistics, vol. 26, no. 2, pp. 451–471, 1998.

[12] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” Machine Learning, vol. 6, no. 1, pp. 37–66, 1991.

[13] J. Friedman, T. Hastie, and R. Tibshirani, “Additive Logistic Regression: a Statistical View of Boosting,” Annals of Statistics, vol. 28, p. 2000, 1998.

[14] A. Hindle, D. M. German, M. W. Godfrey, and R. C. Holt, “Automatic classication of large changes into maintenance categories,” in IEEE 17th International Conference on Program Comprehension, 2009. ICPC ’09, 2009, pp. 30–39.

[15] N. Holden and A. A. Freitas, “Improving the Performance of Hierarchical Classification with Swarm Intelligence,” in Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, E. Marchiori and J. H. Moore, Eds. Springer Berlin Heidelberg, 2008, pp. 48–60.

[16] D. Hull, “Using Statistical Testing in the Evaluation of Retrieval Experiments,” in Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 1993, pp. 329–338.