Predicting Estimated Time of Arrival Using Boosting Models

Say-Hong Kam
School of Computer Sciences
Universiti Sains Malaysia
Penang, Malaysia
kamsayhong@student.usm.my

Yung-Wey Chong
National Advanced IPv6 Centre
Universiti Sains Malaysia
Penang, Malaysia
chong@usm.my

Noor Farizah Ibrahim
School of Computer Sciences
Universiti Sains Malaysia
Penang, Malaysia
nfarizah@usm.my

Sye-Loong Keoh
School of Computing Science
University of Glasgow
Glasgow, United Kingdom
SyeLoong.Keoh@glasgow.ac.uk

Somnuk Phon-Amnuaisuk
School of Computing and Informatics
Universiti Teknologi Brunei
Brunei, Brunei
span.amnuaisuk@gmail.com

Sharul Kamal Abdul Rahim
Faculty of Electrical Engineering
Universiti Teknologi Malaysia
Johor, Malaysia
sharulkamal@utm.my

***Abstract*— Estimating the time of arrival (ETA) in public transportation can be challenging due to incomplete data and the complex nature of the urban environment. This study aims to address persistent criticism of the poor punctuality problem in Malaysian buses through the modeling of bus arrival time predictions. The study uses geographical and time data to predict bus arrival times through several boosting models. The data cleaning method enhanced data quality by eliminating invariable entries, segmenting the bus route for a more granular analysis, and encoding the data for improved structure and reliability. Through the implementation of Boruta for feature selection, relevant variables crucial for prediction were identified, contributing to the model's precision. The results highlighted LightGBM's superiority over AdaBoost and XGBoost, exhibiting the highest accuracy and a balanced level of complexity. This integrated methodology not only presents a robust prediction model but also showcases a potential practical implementation.**

Keywords—bus arrival prediction, Boosting, Boruta, LightGBM, XGBoost, AdaBoost

I. INTRODUCTION

Public transportation plays a crucial role in the daily lives of urban residents, providing an efficient mode of commuting. Buses remain the most affordable public transport option in Johor Bahru, aside from taxis, meeting the diverse travel needs of the population. However, one of the enduring challenges faced by bus passengers is the uncertainty surrounding bus arrival times, leading to frustration and inconvenience. A study revealed that Malaysians have grappled with similar problems for decades, including the punctuality of bus drivers, a lack of schedule information, and a lack of continuous supervision by authorities [1].

The necessity for a prediction model to forecast bus travel times has become increasingly evident in recent years. However, the high installation and maintenance costs of On-Board Diagnosis (OBD) sensors [2], make it progressively challenging to collect and predict bus arrival time data. Moreover, sensor data is known for its temporal and spatial correlation characteristics. Temporal correlation implies that current moment data has a quantitative relationship with next moment data, potentially leading to invariable data. Spatial correlation suggests that data generated by nodes in a specific space have similar quantitative relationships. For example, sensor data collected from a bus route may exhibit similar error ranges [3].

Besides, computational resources are also one of the major issues. Most of the proposed models require a significant amount of resources, such as deep learning [4, 5] and support vector machine [6]. Although these models produce relatively accurate predictions, they demand substantial resources, making them impractical in our case.

This paper addresses these issues by implementing:

1. A data cleaning method to improve data quality by removing duplicate data from Bluetooth Low Energy (BLE) based system [7], breaking down bus routes into segments, and transforming data for training.
2. Feature selection method to select the most relevant features through the use of Boruta.
3. Boosting models to minimize computational power while preserving accuracy.

The rest of this paper is organized as follows. Section II presents the related works on dealing with bus ETA. Section III describes the data and methods used in these experiments. The results and discussions are presented in Section IV. Finally, the conclusion and future works are provided in Section V.

II. LITERATURE REVIEW

A. Data Preprocessing

One of the challenges in handling sensor data is dealing with noisy data. There is a high density of GPS points around a fixed location when a bus is approaching a bus stop. This density makes it difficult to precisely determine when the bus arrives, leaves, or stays at such stops—a phenomenon referred to as stay points issues. Additionally, missing data points are common along bus routes due to sensor failures. [8] addresses these challenges through trip segmentation and points interpolation methods. In trip segmentation, duplicate GPS points are removed. Furthermore, if the time difference between two consecutive GPS points is greater than 900 seconds, the point is treated as the last point of one trip and the first point of the next trip, resolving the missing data issue. For stop-based trip interpolation, kd-tree method is used to search for the nearest bus stop corresponding to the GPS data and fed this information into machine learning models. The application of these methods resulted in the best model demonstrating a low Root Mean Square Error (RMSE) value of 128 seconds per stop.

[9] proposed a spatial-based feature vector analysis to address the complexity and variability of city traffic. It suggests a similar method to that of [8], involving slicing the bus route and subsequently calculating the average travel speed to tackle the issue. Consequently, this approach yields satisfactory performance, with the prediction error for each station being below 1 minute.

B. Feature Selection

Feature selection is necessary when one is interested in discovering the mechanism related to the subject of interest rather than building a black-box model. In this case, identifying variables that are crucial to the prediction model is important for understanding how these variables affect the model.

One of the most common methods for finding correlations between variables is using the filter method. The filter method selects features based on a performance measure, regardless of the employed data modeling algorithm. It can be classified into information, distance, consistency, similarity, and statistical measures [10]. However, the filter method can only provide correlations between features and the target variable, not with other features. Therefore, one is restricted to the wrapper method, which is more computationally intensive than the filter method.

Boruta, as a wrapper method, has gained popularity recently. It offers a heuristic procedure designed to find all relevant attributes, including those weakly relevant attributes. The algorithm is capable of identifying attributes that are independently correlated with some other variables [11].

C. Modelling

Existing works focusing on building bus prediction models fall into these few categories:

1) Time Series Model: This model assumes that the temporal variations of the predicted value are periodic and forecasts the predicted value based on historical data. [9] used a hybrid of Long-Term Short-Term Memory (LSTM) and Artificial Neural Network (ANN) models to predict bus arrival time in China. It successfully predicts the bus arrival time with outstanding performance, an RMSE value of 22.82 seconds. One of the most important parameters for these models is the window length [12]. Without constant time lags, time series forecasting is hardly relevant.

2) Support Vector Machine (SVM): [6] used a hybrid of genetic algorithms and SVM to predict bus arrival time. It summarizes that the implementation of a genetic algorithm into SVM offered a globally optimal solution, helping the algorithm perform better than neural networks and traditional SVM methods. However, the SVM model consumes a lot of computational power when dealing with big data [13].

3) Ensembled Method: [14] introduces ensemble learning methods, including bagging, boosting, and stacking methods to predict bus travel time. The prediction results show that boosting and stacking methods outperformed other algorithms in terms of prediction accuracy. Another study on the Gradient Boosting algorithm further justifies that Adaboost [15], LightGBM and XGBoost, representative form boosting family [16] are able to model nonlinear data, offering high accuracy and a scalable solution.

III. METHODOLOGY

This research utilized data from a BLE-based fleet tracking and analytics [7] system that had been installed in Johor Bahru, Malaysia. It consists of two service routes: P211, from Terminal Taman U to Larkin Sentral and vice versa, and P411, from Terminal Kulai to Larkin Sentral and vice versa as illustrated in Fig. 1 and Fig. 2.

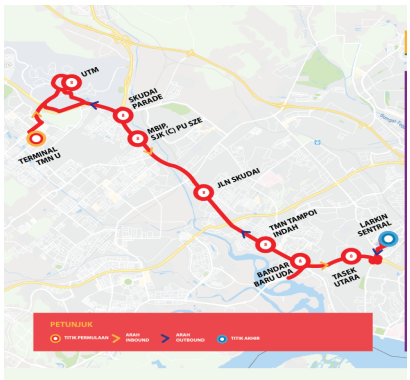

Fig. 1: P211 bus route from Larkin Sentral to Terminal Taman Universiti

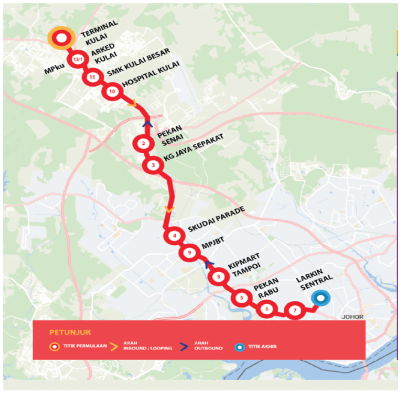

Fig. 2: P411 bus route from Larkin Sentral to Terminal Kulai

BLE-based sensors are installed at selected bus stops. When the bus approaches the bus stop, it will be detected by the sensors and uploaded to the cloud storage. The data consist of 47k instances and 11 features. Table 1 lists the features and their corresponding descriptions.

TABLE 1 FEATURES AND ITS DESCRIPTIONS

Features	Description
Bus_id	The unique id of the bus
Route_id	The unique id of the route
Latitude	Latitude of the bus
Longitude	Longitude of the bus
TimeStamp	Timestamp when the instance is recorded
Bus stop*	Bus stop no of the bus
Route order*	Current order of the route
Dest_route_order*	Destination of the bus
Name*	Nearest Bus stop name
Route*	Name of the route
dist_from_bus_stop*	Current distance of bus from the nearest bus stop
Sin_half_hour*	Hour encoded in sine
Cos_half_hour*	Hour encoded in cosine
Sin_month*	Month encoded in sine
Cos_month*	Month encoded in cosine
Sin_day*	Day encoded in sine
Cos_day*	Day encoded in cosine

Time diff*	The difference of neighbouring timestamp data

(*) indicates derived attribute

A. Data Cleaning and Preparation

The data cleaning and preparation steps are shown in Fig. 3.

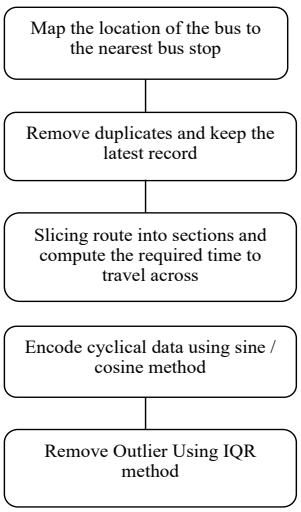

Fig. 3: Data cleaning and preparation.

1) Map the location of bus to the nearest bus stop: Since the bus is captured by the sensor when approaching the bus stop, it is possible to identify the location of the bus and the time it reaches the bus stop. The Ckd Tree is used to match the bus location to the nearest bus stop. Different from the normal kd tree, it uses the midpoint splitting rule to choose the axis and splitting point, making the search more efficient [17].

2) Remove duplicates and keep only the latest records : As the bus approaches the sensor, the sensor tends to record multiple values, affecting data quality. In this project, only the latest record is kept, and others are removed. This is achieved through searching for points where the bus stop ID changes, and only the previous record remains.

3) Slicing route into sections and compute time required to travel across: Sensors were not installed at all bus stop and sometimes the sensors were down. It resulted in missing data. For example, for the bus route travel from Larkin Sentral to Terminal Taman Universiti, the sensor only installed at 12 bus stops out of 28 bus stops, so there are no .data captured at the remaining 16 bus stops. To resolve this issue, the data were further split into sections based on the bus route. Then, the required time to travel from one bus stop to the next consecutive bus stop is calculated. Incomplete sections are excluded from this study.

4) Encode cyclical data using sine cosine method : Next, cyclical data, including month, day, and hours, are encoded using the sine-cosine method. The formula is listed as follows:

$$t_{half\ hour} = \frac{minutes\ of\ day}{30} \quad (1)$$

$$sin_{half\ hour} = \sin(2 \times \pi \times \frac{t_{half\ hour}}{48}) \quad (2)$$

$$cos_{half\ hour} = \cos(2 \times \pi \times \frac{t_{half\ hour}}{48}) \quad (3)$$

$$sin_{month} = \sin(2 \times \pi \times month \div 12) \quad (4)$$

$$cos_{month} = \cos(2 \times \pi \times month \div 12) \quad (5)$$

$$sin_{day} = \sin(2 \times \pi \times day \div 7) \quad (6)$$

$$cos_{day} = \cos(2 \times \pi \times day \div 7) \quad (7)$$

Equation (1) to (7) shows the encoding of hour, month, and day respectively. The division of daily cycle of 24 hours into half an hour enabled a granular analysis, providing more insight [18]. Whereas for the month and day it is divided into 12 (12 months in a year) and 7 (7 days in a week) respectively.

5) Remove Outlier using IQR method : The presence of outliers imposes difficulties for the model to understand the data behavior. In this context, outliers are defined as data points that significantly deviate from the upper and lower bounds, representing instances where bus travel durations are exceptionally longer than the typical values. Interquartile Range Method (IQR) was used to remove outliers and removed approximately 4% of the data. The result of month August is shown in Fig. 4.

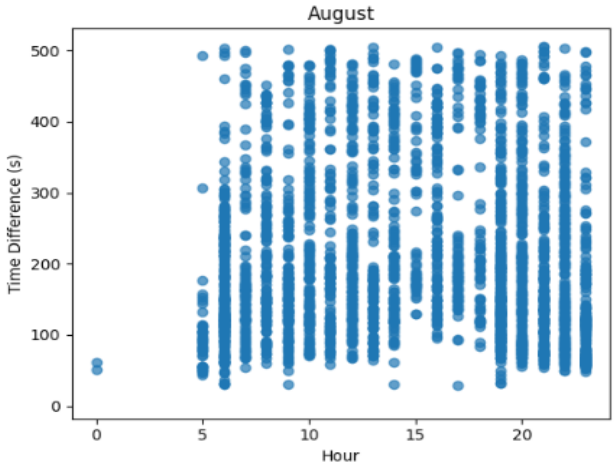

Fig. 4: Time difference vs hours remove outlier.

B. Features selection

Feature selection is carried out using the Boruta package. Boruta is a wrapper method designed around the Random Forest algorithm to iteratively remove features that are statistically irrelevant to the target value [11]. The selected features are listed in Table 2.

TABLE 2 SELECTED FEATURES

Features	Descriptions
Route id	Route Id of the trip
Route order	Origin route order of the trip
Destination route order	Destination route order of the trip
Sin half hour	Hour encoded in sin
Cosine half hour	Hour encoded in cosine
Time difference	Time taken to travel from origin route order to destination, target variable

C. Modelling

Boosting algorithms are commonly used for modeling nonlinear datasets, combining multiple weak learners into a robust learner to enhance prediction accuracy. In this paper, three (3) models namely AdaBoost, XGBoost, and LightGBM are used for regression and feature importance analysis.

1) LightGBM : LightGBM represents an enhancement of the Gradient Boosting algorithm, incorporating improvements in efficiency and scalability. This is achieved through the implementation of Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) using a histogram algorithm data structure. GOSS selectively retains instances with large gradients while randomly sampling instances with small gradients. To maintain data distribution integrity, GOSS introduces a constant multiplier for instances with small gradients during information gain computation [19]. EFB is a method designed to reduce the number of features used in training by grouping together features with similar values and treating them as a single feature [19].

2) XGBoost : Proposing an extension on the Gradient Boosting Decision Tree, XGBoost expands the objective function to the second-order Taylor expansion. This extension provides a more accurate local approximation of the loss function, thereby improving decision tree construction and enhancing predictive accuracy. Additionally, XGBoost introduces regularization techniques to address overfitting issues [20].

3) AdaBoost: AdaBoost combines multiple weak regression models into a strong regression model. Initially assigning equal weight to all data points, it sequentially fits weak regression models to the data. Subsequently, these weak learners are aggregated, and their predictions are weighted based on their performance, resulting in a final ensemble regression model. This process emphasizes hard data points, contributing to the robustness and accuracy of the regression model [21].

D. Evaluation Method

The dataset is sequentially split, with data from September serving as the testing set (approximately 20%), and the remaining data as the training set (about 80%). The training set undergoes validation through 5-fold cross-validation, following the methodology proposed by [22] which has shown that 5-fold cross-validation yields superior outcomes. This approach ensures that the algorithm is tested on an unseen time range, providing a more accurate estimate of the model's quality.

The three (3) models are evaluated using the metrics below:

- Mean Squared Error (MSE): This metric calculates the average squared difference between the predicted and actual values of the target variable.
- Root Mean Squared Error (RMSE): This metric is the square root of the MSE and is also commonly used to measure the average difference between predicted and actual values. RMSE is a more interpretable metric since it is expressed in the same units as the target variable.
- Mean Absolute Error (MAE): This metric measures the average absolute difference between the predicted and actual values of the target variable. MAE is less sensitive to outliers than MSE, making it a useful metric when dealing with skewed data.
- R-squared (R2): This metric measures the proportion of the variance in the target variable that is explained by the model. A higher R-squared value indicates a better fit of the model.

IV. RESULT AND DISCUSSION

TABLE 3 PERFORMANCE OF LIGHTGBM, ADABOOST AND XGBOOST

Algorithms	Training				Testing				Training Time (s)
	R2	***MAE***	***MSE***	***RMSE***	***R2***	***MAE***	***MSE***	***RMSE***	
With cyclical encoding features									
LightGBM	0.8705	26.9656	1529.5820	39.1009	0.8468	30.0727	2079.6053	45.6027	3.75
AdaBoost	0.5409	53.6239	5424.0889	73.6484	0.4899	64.2462	6923.4502	83.2073	0.2031
XGBoost	0.8361	30.4841	1936.6205	44.0071	0.7481	37.8071	3419.7583	58.4787	9.1875
Without cyclical encoding features									
LightGBM	0.8219	31.9949	2104.1626	45.8712	0.8145	32.7940	2517.8129	50.1778	2.4219
AdaBoost	0.5409	53.6239	5424.0889	73.6484	0.4899	64.2462	6923.4502	83.2073	0.1719
XGBoost	0.7858	34.8125	2530.1913	50.3010	0.7184	41.4807	3822.7701	61.8286	6.5156

Table 3 displays the performance of LightGBM, AdaBoost, and XGBoost. Overall, LightGBM achieves the highest accuracy, with the lowest MAE, MSE, and RMSE values, which are 30.0727, 2079.6053, and 45.6027, respectively, across all the algorithms. Additionally, it has the highest R-squared value, reaching 0.8468, indicating its ability to understand the pattern of the dataset. Following LightGBM is XGBoost, which achieves an R-squared value of 0.7481, along with comparable MAE, MSE, and RMSE values of 37.8071, 3419.7583, and 58.4787.

Although AdaBoost exhibits the lowest accuracy, with the lowest R-squared, MAE, MSE, and RMSE values at 0.4899, 64.2462, 6923.4502, and 83.2073, respectively, its training time is the shortest, completing training in only 0.2031 seconds. In contrast, LightGBM's training time is 18 times that of AdaBoost, taking 3.75 seconds, while XGBoost is approximately 100 times longer, requiring 9.1875 seconds.

All predictions were made using only five features: Route ID, Route Order, Destination Route Order, Sine Half Hour, and Cosine Half Hour. It is evident that significant time differences exist between routes and hours. For example, time required to travel through a route is lowest from Terminal Taman Universiti to Larkin Sentral, taking only about 86 seconds to travel to the next bus stop about 0.7 km , significantly lower than its return bus route which takes about 92 seconds. Similar pattern also observed in bus route from Kulai Bus Terminal to Larkin Terminal and its return bus route, in which bus travel time across 1 bus stop distance, about 1 km takes 115 seconds and 92 seconds respectively. Additionally, it is observed that buses take longer to travel at 7 am and 7 pm compared to other periods.

Comparing the three models, LightGBM outperforms the others. It not only achieves the highest accuracy, with an RMSE of approximately 46 seconds and 85% R-squared but also has the shortest training time. It performs 10% better than XGBoost in terms of accuracy and is about 10 times faster. On the other hand, although AdaBoost is 10 times faster than LightGBM, its accuracy is unacceptable, falling about 35% less than LightGBM.

One reason for AdaBoost's relatively poor performance in this experiment is its sensitivity to outliers. As outliers are more likely to be misclassified, they are likely to carry larger weights, diminishing their generalization power [23]. The gradient boosting algorithm, by introducing a gradient-based incremental search, enhances its generalization power and thus performs better than AdaBoost.

Besides, we also notice that the usage of cyclical features improves the R2 value about 3% and 3-5 seconds in terms of RMSE value. It shows that cyclical encoding effectively encodes cyclical information of time series data, improving model's prediction power. Although it performs great at XGBoost and LightGBM model, it does not bring any effect to the AdaBoost model. It aligns with the findings of [24] where not all model benefit from cyclical encoding.

V. CONCLUSION AND FUTURE WORK

This paper evaluated the performance of bus arrival prediction model using three (3) boosting models: AdaBoost, XGBoost, and LightGBM. The results indicated that LightGBM surpassed both XGBoost and AdaBoost, demonstrating superior accuracy and efficiency. Despite having the shortest runtime, AdaBoost performed the least accurately among the three algorithms.

To obtain an accurate estimated time of arrival, various parameters such as flood, nearby traffic, accidents have to be complete. Hence, the proposed study only fits to the existing situation.

In future work, our goal is to enhance the performance of ETA prediction by exploring the combination of multiple stacking model algorithms. This approach aims to build a more robust model capable of mitigating the impact of noise and further improving prediction accuracy.

ACKNOWLEGMENT

This publication is the output of ASEAN IVO (https://www.nict.go.jp/en/asean_ivo/Project_List_of_ASEAN_IVO.html) project, "An IoT-based public transport data collection and analytics framework using Bluetooth proximity beacons" and financially support by NICT (http://www.nict.go.jp/en/ index.html). The research was also supported by Keio University and APNIC Foundation under CBR grant number 304/PNAV/6501372/K164.

REFERENCES

[1] K. Soh, C. L. Chong, W. Wong, and Y. Hiew, "Proclivity of university students to use public bus transport service," *Comprehensive Research Journal of Education and General Studies (CRJEGS),* vol. 2, no. 2, pp. 24-34, 2014.

[2] M. A. B. SHAFIE, "rrnrrrrn," HOCHSCHULE KARLSRUHE TECHNIK UND WIRTSCHAFT, 2016.

[3] C.-H. Zhou, B. Chen, Y. Gao, C. Zhang, and Z.-J. Guo, "A technique of filtering dirty data based on temporal-spatial correlation in wireless sensor network," *Procedia Environmental Sciences,* vol. 10, pp. 511-516, 2011.

[4] N. C. Petersen, F. Rodrigues, and F. C. Pereira, "Multi-output bus travel time prediction with convolutional LSTM neural network," *Expert Systems with Applications,* vol. 120, pp. 426-435, 2019.

[5] P. P. F. Balbin, J. C. Barker, C. K. Leung, M. Tran, R. P. Wall, and A. Cuzzocrea, "Predictive analytics on open big data for supporting smart transportation services," *Procedia Computer Science,* vol. 176, pp. 3009-3018, 2020.

[6] M. Yang, C. Chen, L. Wang, X. Yan, and L. Zhou, "Bus arrival time prediction using support vector machine with genetic algorithm," *Neural Network World,* vol. 26, no. 3, p. 205, 2016.

[7] S. Gunady and S. L. Keoh, "A non-gps based location tracking of public buses using bluetooth proximity beacons," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019: IEEE, pp. 606-611.

[8] D. Liu, J. Sun, and S. Wang, "Bustime: Which is the right prediction model for my bus arrival time?," in *2020 5th IEEE International Conference on Big Data Analytics (ICBDA)*, 2020: IEEE, pp. 180-185.

[9] H. Liu, H. Xu, Y. Yan, Z. Cai, T. Sun, and W. Li, "Bus arrival time prediction based on LSTM and spatial-temporal feature vector," *IEEE Access,* vol. 8, pp. 11917-11929, 2020.

[10] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th international convention on information and communication technology,*

electronics and microelectronics (MIPRO), 2015: Ieee, pp. 1200-1205.

[11] M. B. Kursa and W. R. Rudnicki, "Feature selection with the Boruta package," *Journal of statistical software,* vol. 36, pp. 1-13, 2010.

[12] A. Azlan, Y. Yusof, and M. F. M. Mohsin, "Determining the impact of window length on time series forecasting using deep learning," *International Journal of Advanced Computer Research,* vol. 9, no. 44, pp. 260-267, 2019.

[13] Y. Bin, Y. Zhongzhen, and Y. Baozhen, "Bus arrival time prediction using support vector machines," *Journal of Intelligent Transportation Systems,* vol. 10, no. 4, pp. 151-158, 2006.

[14] G. Zhong, T. Yin, L. Li, J. Zhang, H. Zhang, and B. Ran, "Bus travel time prediction based on ensemble learning methods," *IEEE Intelligent Transportation Systems Magazine,* vol. 14, no. 2, pp. 174-189, 2020.

[15] W. Fan, S. J. Stolfo, and J. Zhang, "The application of AdaBoost for distributed, scalable and on-line learning," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1999, pp. 362-366.

[16] V. A. Dev and M. R. Eden, "Formation lithology classification using scalable gradient boosted decision trees," *Computers & chemical engineering,* vol. 128, pp. 392-404, 2019.

[17] S. Maneewongvatana and D. M. Mount, "It's okay to be skinny, if your friends are fat," in *Center for geometric computing 4th annual workshop on computational geometry*, 1999, vol. 2: Citeseer, pp. 1-8.

[18] Y. T. Lim, "A Deep Learning based Prediction of Bus Arrival Timings," Final Year Project Report Singapore Institute of Technology-University of Glasgow, 2023

[19] G. Ke *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems,* vol. 30, 2017.

[20] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785-794.

[21] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences,* vol. 55, no. 1, pp. 119-139, 1997.

[22] L. Breiman and P. Spector, "Submodel selection and evaluation in regression. The X-random case," *International statistical review/revue internationale de Statistique,* pp. 291-319, 1992.

[23] A. H. Li and J. Bradic, "Boosting in the presence of outliers: Adaptive classification with nonconvex loss functions," *Journal of the American Statistical Association,* vol. 113, no. 522, pp. 660-674, 2018.

[24] T. Mahajan, G. Singh, G. Bruns, G. Bruns, T. Mahajan, and G. Singh, "An experimental assessment of treatments for cyclical data," in *Proceedings of the 2021 Computer Science Conference for CSU Undergraduates, Virtual*, 2021, vol. 6, p. 22.