Density Analysis Based Flight Delay Prediction with Genetic Algorithm Hyperparameter Tuning

Peerawat Nakornsri[1], Pruttipong Apivatanagul[2] and Phat Pisitkasem[3]

Master of Science Program in Management of Logistics, Graduate School, Rangsit University

---

ABSTRACT

Flight delays are a major problem in the current aviation system. Once there is a flight delay, this causes chain delays at multiple airports, which results in tremendous economic loss. This paper presented a machine learning (ML) approach for the prediction of flight delays by using a decision tree based algorithm called gradient boosting (XGBoost). The genetic algorithm was used as a tuning parameter for optimization to improve the prediction performance. Artificial intelligence (AI) analysis determined the likelihood of the effect of the flight delays by intervals. The departure delays and late arrival delays were the most important parameters for predicting the delays. The result (prediction accuracy) of the density-based method model was compared to the long short-term memory (LSTM) method. The experimental result showed that the top three causes of delays were the late arrival of aircraft by more than 45 minutes, current delay status of the airport, and the amount of planned departure flights.

Keywords: Flight Delay Prediction, Gradient Boosting Algorithm, Genetics Algorithm, Mass Density Features, Machine Learning

1. Introduction

Flight delays cause approximately 600 million USD per year in economic loss, which has been calculated from human time and fuel consumption. However, this issue could be resolved through the initiation of a smart schedule plan. One important key factor of flight schedule optimization is the prediction of airport delays (Sternberg et al., 2017). Consequently, the anticipated delay prediction through the adjustment of the arrival and departure times would reduce delays. In order to predict airport delays, the data used for the calculation comprised the current delay information, arrival flights information, departure flights information, weather information, etc. These data were utilized as a random variable in a machine learning (ML) model, which was implemented separately for every airport. The results showed the distribution of the next 15-minute airport flight delays.

In this research, flight delays and their root causes were the main considered targets to be predicted by using ML algorithms. Flight delays cause a huge loss in the air transportation industry; therefore, in predicting the delay in advance could reduce some economic loss (Du et al., 2018). Since the delay time was a continuous property, multiple classification problems were constructed to act as a regression problem, as this would reduce the

complexity of the problem. A density analysis was used to create the features, then they were fed into a gradient boosting classification algorithm. Moreover, a genetics algorithm was introduced to optimize the parameters related to the classification algorithm.

Chen and Li (2019) showed that delays had an effect on the next flights, which were called propagation delays. A random forest classifier and a delay propagation model were used for predicting the delays. The departure delays and late arrival delays were shown to be the most important features for predicting the delays. Yu et al. (2019) showed that aviation had a lot of wasted costs when flight delays occurred. They used the deep belief network and support vector regression (DBN-SVR) and detection of the key influential factors for deep learning, which they concluded that air traffic control (ATC) was the most important factor for the delays.

Gui et al. (2019) used the automatic dependent surveillance–broadcast (ADS-B) message based aviation big data platform and flight prediction formulation. They used data from the ground stations and data from the Internet for ML. Several kind of deep learning models such as RRN, NLP, and long short-term memory (LSTM) were also used. Random forest-based and LSTM-based architecture was implemented to predict individual flight delays.

2. Objectives of the Study

To develop effective machine learning (ML) models to predict the probability of the flight delays for each individual airport.

3. Materials and Methods

This section will provide the details to demonstrate the full replication of the study by suitably skilled investigators. The protocols for new methods would be included, but well-established protocols may simply be referenced.

The development of the machine learning (ML) algorithms consisted of four steps (Figure 3.1). First, the problem had to be transformed from a business problem into a ML problem. Second, a data visualization tool explored the appropriateness of the data. Third, feature extraction, model, and feature selection were implemented. Fourth, feature selection and hyperparameter tuning optimization were added, as it was expected that the performance for the algorithm would be increased.

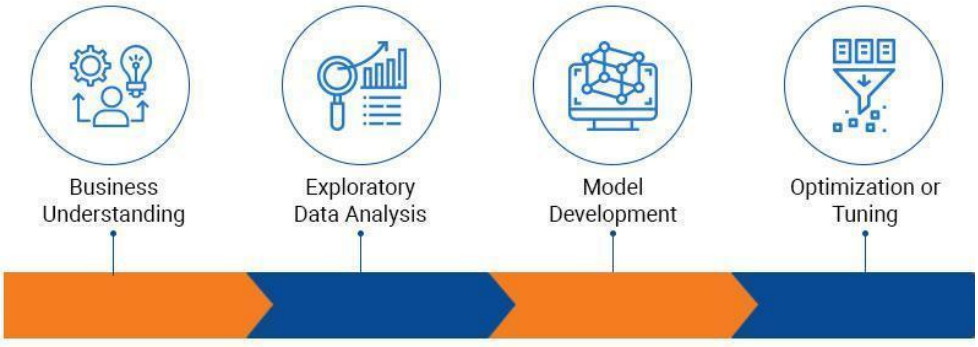

Figure 1 Machine learning (ML) development life cycle.

3.1 Business Understanding

The ML problem could either be supervised or unsupervised, and discrete or continuous (Figure 3.2). Hence, the problem of the flight delays was supervised in a continuous domain.

	Supervised Learning	*Unsupervised Learning*
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Figure 2 Machine learning (ML) problem.

In order to simplify this continuous problem, the regression problem was divided into five classifications:

1. Would the incoming flight in the next 10-minute be delayed by 10-minute?
2. Would the incoming flight in the next 10-minute be delayed by between 10-20 minutes?
3. Would the incoming flight in the next 10-minute be delayed by between 20-30 minutes?
4. Would the incoming flight in the next 10-minute be delayed by between 30-40 minutes?
5. Would the incoming flight in the next 10-minute be delayed by between 40-50 minutes?
6. Would the incoming flight in the next 10-minute be delayed by between 50-60 minutes?
7. Would the incoming flight in the next 10-minute be delayed by more than 60 minutes?

The expected value of the delay time in a 10-minute time frame (of a selected airport) would be equal to 10xP_10Mdelay + 20xP_20Mdelay + 30xP_30Mdelay + 40xP_40Mdelay + 50xP_50Mdelay + 60xP_60Mdelay

3.2 Exploratory Data Analysis

In this research, the flight delay information in the USA during January to December 2015 was used for developing the ML algorithm. The size of the data was approximately 600MB. The data comprised three tables: 1. the airlines had two columns and 14 rows, 2. the airports had seven columns and 323 rows, and 3. the flights had 31 columns and 5.82 million rows (from https://www.kaggle.com/.usdot/flight-delays/data). A data visualization tool, Microsoft Power BI, was used to explore these data.

3.2.1. Quantity of the data

The amount of records is shown in Figure 3.3.

Figure 3 Amount of records.

3.2.2. Distribution of the delays

The level of the delays, frequency and duration are described in Figure 3.4.

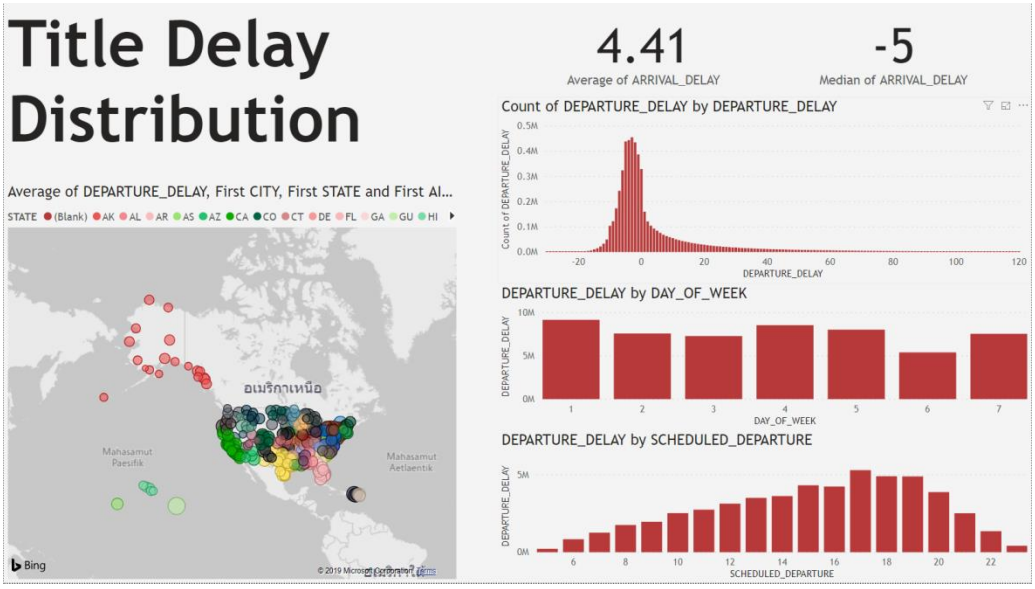

Figure 4 Distribution of the delays.

3.2.3. Potential causes of the delays

The potential causes of the delays is shown in Figure 3.5.

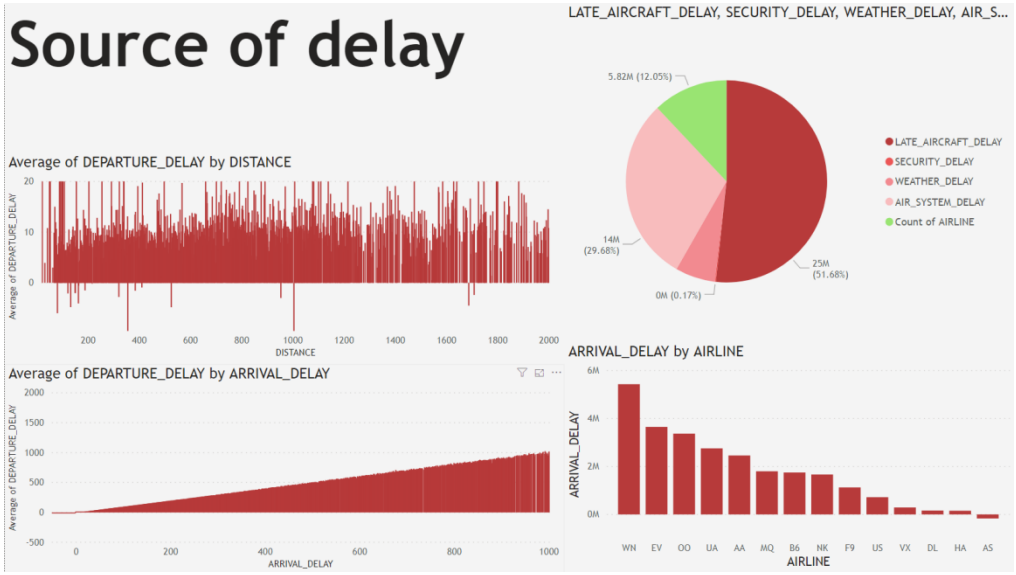

Figure 5 Source of the delays.

3.3 Model Development

3.3.1 Density analysis based feature extraction

In order to utilize the performance of the ML algorithms, the input data needed to be extracted from the raw data. Therefore, the delay density distribution or histogram and its normalization were calculated for most of the features related to each airport. The features were divided into the following categories:

- Features that described the seasonal effect
- Day of the week
- Month
- Hour
- Week of the month
- Features that described the current and historical state of the airport. These features informed the algorithm about the current delay problem, and how prone the airport would be to delays in the future. Nevertheless, a chain of effects from other airports was included because the information was captured from the data of the flight arrivals. All of the features used in this algorithm were based on a histogram of a 10-minute interval (this was limited by the computational power).
- Count of the flight departures by interval (the histogram of the departed flights in the past 10-minute).
- Count of the flight arrivals by interval (the histogram of arrived flights in the past 10-minute).
- Count of the delayed departures by interval (the histogram of the delays in the past 10-minute).
- Ratio of the delayed departures by interval (the probability density of the delayed departures in the past 10-minute).
- Count of the departure taxi time by interval (the histogram of the departure taxi time in the past 10-minute).
- Ratio of the departure taxi time by interval (the probability density of the departure taxi time in the past 10-minute).
- Count of the arrival delay by interval (the histogram of the arrival delay in the past 10 minutes).
- Ratio of the arrival delay by interval (the probability density of the arrival delay in the past 10-minute).
- Count of the arrival taxi time by interval (the histogram of the arrival taxi time in the past 10-minute).
- Ratio of the arrival taxi time by interval (the probability density of the arrival taxi time in the past 10-minute).
- Features that described the schedule of the airport (plan). The features informed the current situation of the delays, which could trigger the delays to happen if the current state of the airport was prone to delays. All of the features used in this algorithm were based on a histogram of a 10-minute interval (this was limited by the computational power).
- Count of the departure flights (the histogram of the flights that would depart in the next 10-minute).
- Count of the arrival flights (the histogram of the flights that would arrive in the next 10-minute).
- Count of the departure distance by 0, 500, 1000, 2000, 4000, and 8000 inf km interval (the histogram of the departure distance in the next 10-minute).
- Ratio of the departure distance by 0, 500, 1000, 2000, 4000, and 8000 inf km interval (the probability density of the departure distance in the next 10-minute).

- Count of the arrival distance by 0, 500, 1000, 2000, 4000, and 8000 inf km interval (the histogram of the arrival distance in the next 10-minute).
- Ratio of the arrival distance by 0, 500, 1000, 2000, 4000, and 8000 inf km interval (the probability density of the arrival distance in the next 10-minute).
- Count of the departure delay by interval (the histogram of the departure delay in the next 10-minute).
- Ratio of the departure delay by interval (the probability density of the departure delay in the next 10-minute).
- Count of the arrival delay by interval (the histogram of the arrival delay in the next 10-minute).
- Ratio of the arrival delay by interval (the probability density of the arrival delay in the next 10-minute).
- Count of the arrival delay of the departing flight (the histogram of the departure delay and related arrival delay in the next 10-minute).
- Ratio of the arrival delay of the departing flight (the probability density of the departure delay and related arrival delay in the next 10-minute).
- Count of the flights that have a departure after arriving less than 12 hours (the histogram of non-layover flights in the next 10-minute).
- Ratio of the continuous flights that have a departure after arriving less than 12 hours (the probability density of non-layover flights in the next 10-minute).

3.3.2 XGBoost classification

The XGBoost algorithm is a flexible and portable gradient boosting library (Chen and Guestrin, 2016). The process of this algorithm is described in Figure 3.6. This algorithm proved to be the most appropriate algorithm in almost every competition in Kaggle.com.

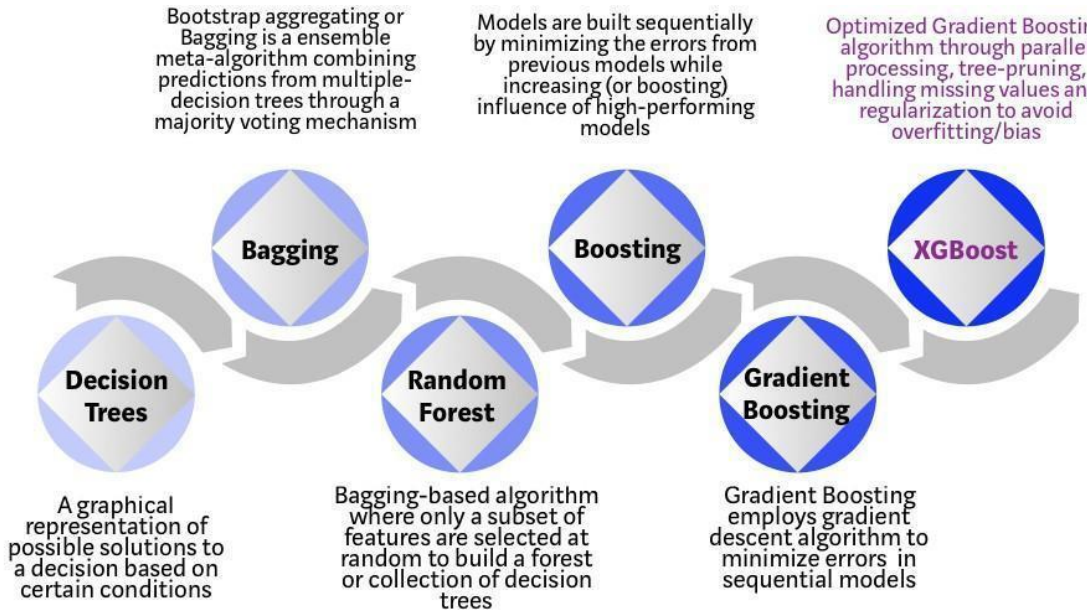

Figure 6 Process of the XGBoost algorithm (Goon, 2018).

3.4 Hyperparameters Optimization

The common parameters that highly affected the algorithm performance were the number of estimators, maximum tree depth, minimum number of splits, and learning rate (Chen and Guestrin, 2016). These parameters were fed into the genetics algorithm optimization where the cost function was the classification loss (Di Francescomarino et al., 2018).

3.5 Tools

In this research, Microsoft Power BI was used for the exploratory data analysis. The predictive model was developed on a Python ML environment called Anaconda 3. The specification of the computing machine was CPU: i7-7600 with 32 GB of ram.

The Python library used in this project comprise the following:

3.5.1 Random forest classifier

A random forest is a meta estimator that fits a number of decision tree classifiers on various subsamples of the data set and uses averaging to improve the predictive accuracy and control the overfitting. The size of the subsample is always the same as the original input sample, but the samples are drawn with the replacement if bootstrap=True (default).

class sklearn.ensemble.RandomForestClassifier(n_estimators=100; criterion='gini'; max_depth=None; min_samples_split=2; min_samples_leaf=1; min_weight_fraction_leaf=0.0; max_features='auto';

max_leaf_nodes=None; min_impurity_decrease=0.0; min_impurity_split=None; bootstrap=True; oob_score=False; n_jobs=None; random_state=None; verbose=0; warm_start=False; class_weight=None; ccp_alpha=0.0; max_samples=None).

3.5.2 Gradient boosting (XGBoost)

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements ML algorithms under the Gradient Boosting framework.

class xgboost.DMatrix (data; label=None; weight=None; base_margin=None; missing=None; silent=False; feature_names=None; feature_types=None; nthread=None.)

xgboost.train (params; dtrain; num_boost_round=10; evals=(); obj=None; feval=None; maximize=False; early_stopping_rounds=None; evals_result=None; verbose_eval=True; xgb_model=None; callbacks=None.)

3.5.3 Genetics algorithm

Customized algorithms were written in Python with Numpy and Pandas. The optimization was conducted to find the appropriate general model that would be suitable for every airport; thus, the data sets from all the airports were combined before fitting the model. The main parameters of the XGBoost algorithm that affected the accuracy of the model were the learning rate, minimum, sample split, subsample, and maximum depth.

The population number was limited to 100 with only the best 10 chromosomes being selected from each iteration. The crossover rate was set to 10%, and the mutation rate was set to 10%.

4. Results & Discussion

Before training a model for each airport, first a dummy model was trained and optimized for finding the best parameter setup of the machine learning (ML) model. Figure 4.1 shows the decrease of the training error through each iteration of the optimization. The five main features that could have an important effect were the chain delays at each airport (Table1). First, arriving aircraft that were late by more than 45 minutes were the most effective for the chain delay, which had importance of 0.278. This was followed by the past delays of departures -5 to 5-minute delays, which had importance of 0.089 then the planned number of departures that had importance of 0.057. Other features and their importance are shown in Table 1. The results were shown as the normalization of the causes due to the delays. Table 2, shows the optimized parameter setup of the XGBoost algorithm.

The results were compared with the long short-term memory (LSTM) method (Gui et al., 2019). Both the area under the ROC curve (AUC) and the R-squared were the two metrics for comparing the performance. The Y axis represented a true positive and the X axis represented a false positive. The density based features extraction method showed an increase of both the AUC and R-squared.

The final models were fitted for each individual airport. Hartsfield-Jackson Atlanta International Airport is shown in Figure 4.4 as the LSTM method and Figure 4.5 as the density based method. Denver International Airport is shown in Figure 4.6 as the LSTM method and Figure 4.7 as the density based method. Dallas/Fort Worth International Airport is shown in Figure 4.8 as the LSTM method and Figure 4.9 as the density based method. George Bush Intercontinental Airport is shown in Figure 4.10 as the LSTM method and Figure 4.11 as the density based method. Chicago O'Hare International Airport is shown in Figure 4.12 as the LSTM method and Figure 4.13 as the density based method.

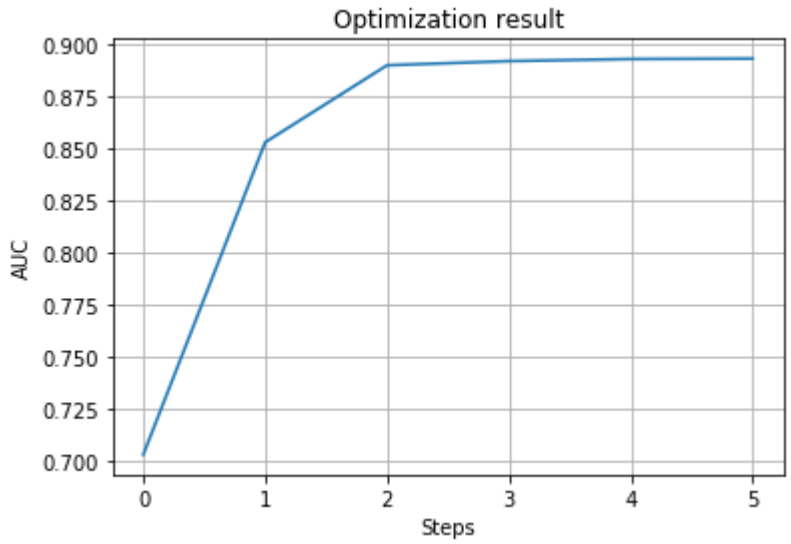

Figure 7 GA-hyperparameter tuning result.

Table 1 Features and importance.

Feature	Importance
Past delay of arrivals more than 45 minutes.	27.8%
Past delay of departures -5 to 5 minutes.	8.9%
Past number of departures.	5.7%
Planned number of the distance of arrival -15 to -5 minutes.	4.8%
Past delay of arrivals less than -15-minute.	4.2%

Table 2 Optimal hyperparameters.

Parameters	Values
learning_rate	0.1
min_split_loss	0.05
max_depth	8
subsample	0.5

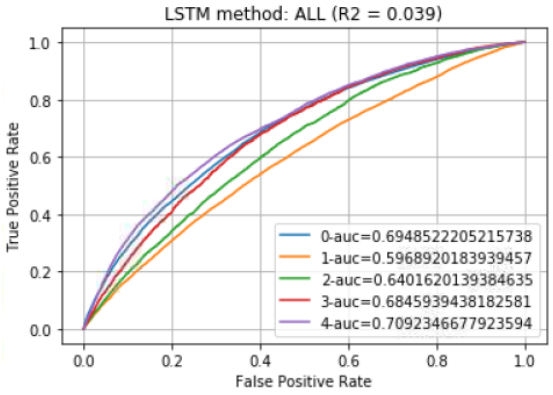

Figure 8 Average performance (LSTM)

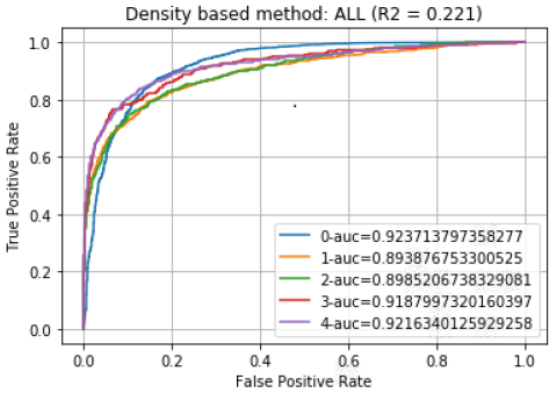

Figure 9 Average performance (Density) based method.

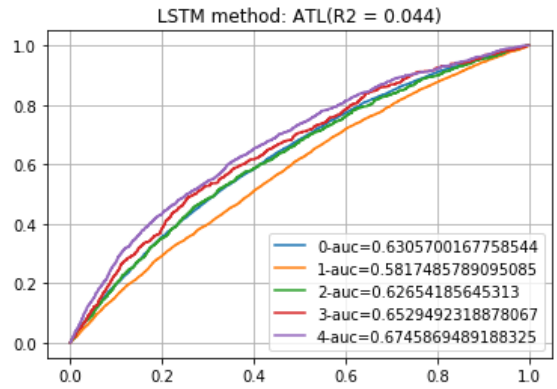

Figure 10 Performance of ATL (LSTM)

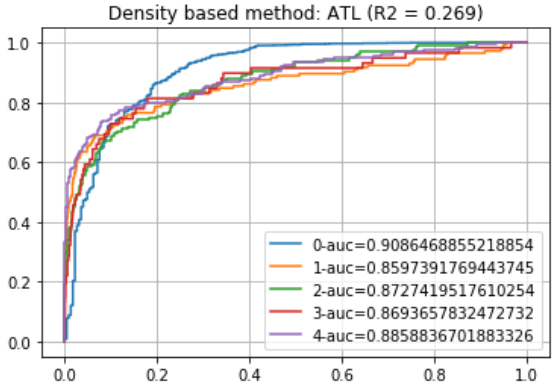

Figure 11 Performance of ATL (Density)

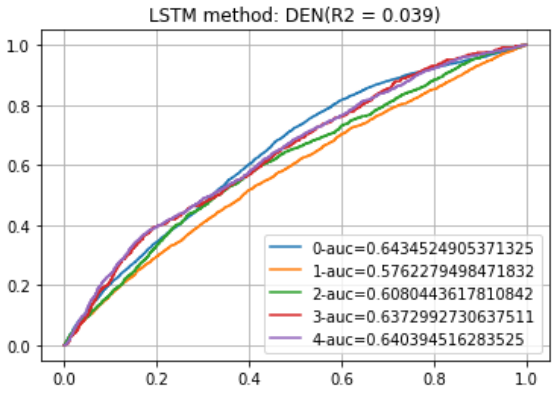

Figure 12 Performance of DEN (LSTM)

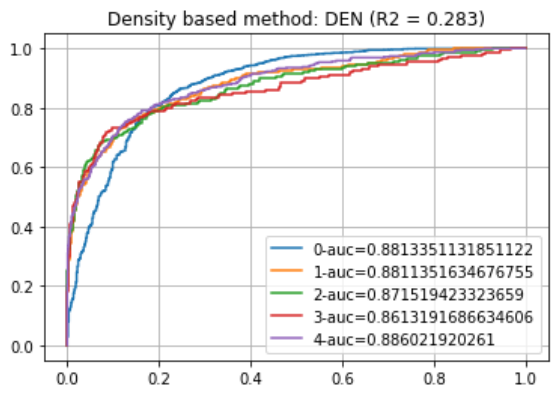

Figure 13 Performance of DEN (Density)

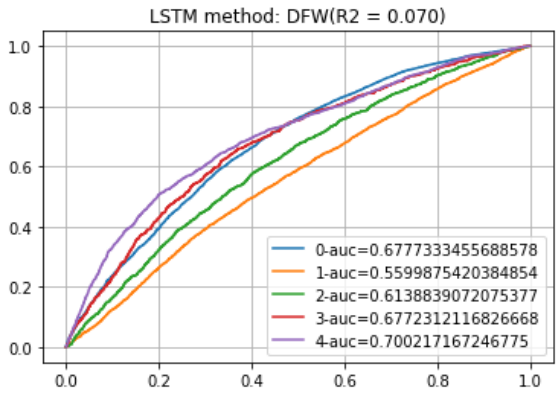

Figure 14 Performance of DFW (LSTM)

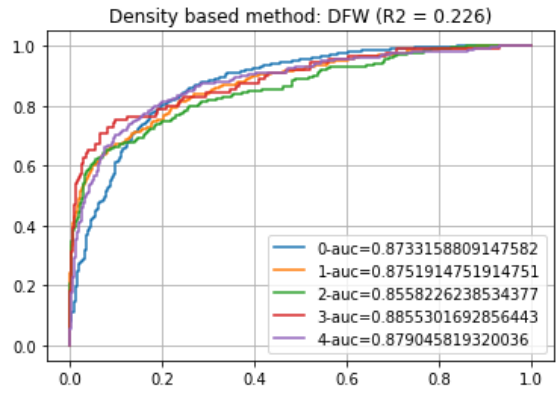

Figure 15 Performance of DFW (Density)

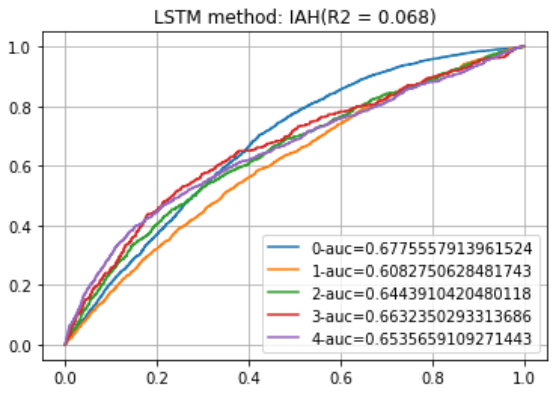

Figure 16 Performance of IAH (LSTM)

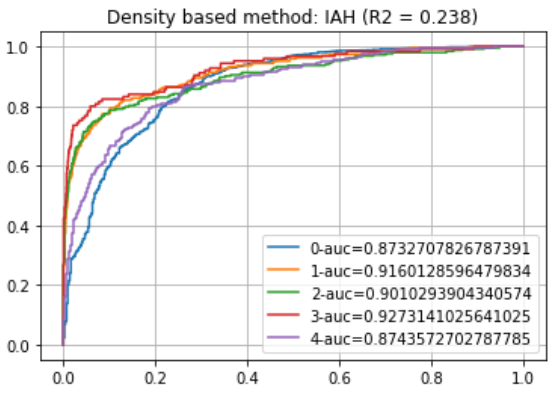

Figure 17 Performance of IAH (Density)

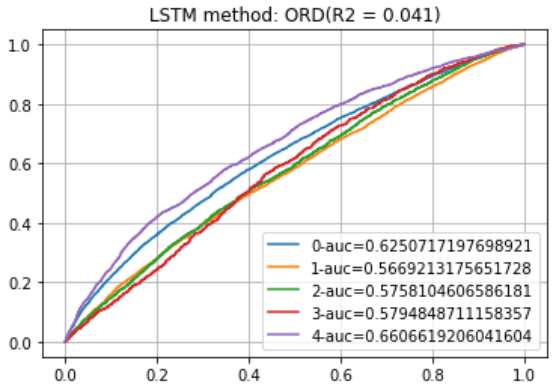

Figure 18 Performance of ORD (LSTM)

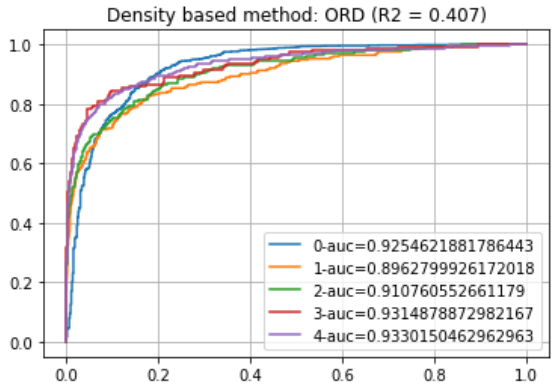

Figure 19 Performance of ORD (Density)

5. Conclusion

In this research, the density based feature extraction for machine learning (ML) with GA-hyperparameter tuning was proposed. A probability mass function of the features was derived. Then, the features were modeled by the XGBoost algorithm. Moreover, the model was optimized by using the genetic algorithm. Therefore, the results showed that the proposed method achieved a significantly higher R-squared and AUC than the latest research deep learning method.

References

Chandraa, Pranalli, Prabakaran, N., & Kannadasan, R. (2018). Airline delay predictions using supervised machine learning. *International Journal of Pure and Applied Mathematics*, 329-337.

Chen, Jun, & Li, Meng. (2019). *Chained predictions of flight delay using machine learning*. San Diego State University & Purdue University. DOI: 10.2514/6.2019-1661.

Chen, Tianqi, & Guestrin, Carlos. (2016). *XGBoost: A scalable tree boosting system*. Retrieved From https://arxiv.org/abs/1603.02754.

Di Francescomarino, Chiara, Dumas, Marlon, Federici, Marco, Ghidini, Chiara, Maggi, Fabrizio Maria, Rizzi, Williams, & Simonetto, Luca. (2018). *Genetic algorithms for hyperparameter optimization in predictive business process monitoring*. DOI: 10.1016/j.is.2018.01.003

Du, Wen-Bo, Zhang, Ming-Yuan, Zhang, Yu, Cao, Xian-Bin, & Zhang, Jun. (2018). "Delay causality network in air transport systems". *Transportation Research Part E: Logistics and Transportation Review*, *118*, 466-476.

Goon, Tirthankar. (2018, February 23). *Difference between random forest and gradient boosting*. algo. inkedin.com.

Grus, Joel. (2015). *Data science from scratch*. O'Reilly Media, Inc.

Gui, G., Liu, F., Sun, J., Yang, J., Zhou, Z., & Zhao, D. (2019). Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology. 1-1,* DOI: 10.1109/tvt.2019.2954094

Hinterding, R., Gielewski, H., & Peachey T. (1995). *The nature of mutation in genetic algorithm*. University of Pittsburgh.

Mohd Aszemi, Nurshazlyn & Dominic, P.D.D. (2019). “Hyperparameter optimization in convolutional neural network using genetic algorithms”, *International Journal of Advanced Computer Science and Applications*, *10*(6), 269-278.

Sternberg, Alice, Soares, Jorge, Calvalho, Diego, & Ogasawara, Eduardo. (2017). *A review on flight delay prediction.* https://arxiv.org/abs/1703.06118.

Yua, Bin, Guoa, Zhen, Asianb, Sobhan, Wangc, Huaizhu, & Chend, Gang. (2019). “Flight delay prediction for commercial air transport: A deep learning approach”. *Transportation Research Part E Logistics and Transportation Review*. DOI: 10.1016/j.tre.2019.03.013.

Zhang, Du. (2001). *Applying machine learning algorithms in software development*. Department of Computer Science, California State University Sacramento.