Crash Prediction using Ensemble Methods

Mohammed Ameksa[1], Hajar Mousannif[1], Hassan Al Moatassime[2] and Zouhair Elamrani Abou Elassad[1]

[1] *LISI Laboratory, Computer Science Department, FSSM, Cadi Ayyad University, Marrakech, Morocco*

[2] *OSER Research Team, Computer Science Department, FSTG, Cadi Ayyad University, Marrakech, Morocco*

Keywords: Crash Prediction, Random Forest, Gradient Boosting Machine, Extreme Gradient Boosting, Road Safety Modelling, Machine Learning

Abstract: The prediction of traffic accidents is a major concern worldwide due to its negative impact on all sectors. The human and financial losses caused by road accidents have become increasingly important. This study aims to investigate the prediction of an accident using several ensemble-based methods, including the GBM gradient boosting machine, the XGB extreme gradient boosting, and RF random forest. To achieve this, we used driver' inputs data extracted from the outputs of a driving simulator located at Cadi Ayyad University UCA. And the evaluation of the developed models was carried out for both configurations, before and after tuning of hyperparameters. Results show that the RF outperforms GBM and XGB for the default parameters with an accuracy of 93%. However, after hyperparameters optimization, it had been noticed that even if the algorithms are not the same, their performances are almost equal. The highest performance was performed after tuning of hyperparameters.

1 INTRODUCTION

Road traffic crashes and related fatalities are a rapidly increasing trend worldwide due to rapid growth in demographics. According to (Lagarde E, 2020), every year, more than 1 million people die in road traffic crashes worldwide, and more than 50 million are injured. Indeed, the human and financial losses caused by road accidents are becoming more and more important; consequently, a good prediction of the occurrence of an accident before it happens is of great benefit to humanity.

A significant effort has been made to reduce the frequency and the occurrence of accidents on the roadway. Based on the literature search, many studies have been investigated factors that contribute to the occurrence of accidents and analyzed crash predictions (Silva et al., 2020). as well as the potential of data mining techniques to predict the occurrence of a crash has been evaluated in various scientific studies using several algorithms. Indeed, several studies found that Machine learning models are more robust and powerful than statistical models in predicting future events and have been used successfully in many transportation systems.

(Abdelwahab and Abdel-Aty, 2001) built an artificial neural network model for the prediction of driver injury severity in traffic crashes using 1997 traffic accident data from the Central Florida region, the MLP neural network was found to achieve the best performance for training (65.6%) and testing (60.4%). Other machine learning algorithms including, K nearest neighbor (KNN), random forest (RF), support vector machine (SVM), decision trees (DT), Gaussian naive Bayes (Gaussian NB), and adaptive boost (AdaBoost), are used by (Osman et al., 2019) to predict near-crash based on observed in vehicle kinematics data. This mentioned article proves that the AdaBoost model achieved an excellent and consistent performance while predicting near-crashes (by accuracy of 100%) or everyday driving (by accuracy of 98%). (Zouhair et al., 2020) used Artificial Neural Networks (ANN) and Decision Trees (DT) machine learning models to analyze crash events and to outline the effect of both light and heavy rain on road safety. Using driver inputs data as throttle pedal position, brake pedal position, and wheel angle, the author found that the ANN model offered a higher precision with a value of 92.14%. In comparison, the DT model showed an accuracy of 89.90%. Among the limitations highlighted in the latter study was to take advantage of the decision tree benefits by using the ensemble methods.

Ameksa, M., Mousannif, H., Al Moatassime, H. and Elamrani Abou Elassad, Z.
Crash Prediction using Ensemble Methods.
DOI: 10.5220/0010731200003101
In *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning (BML 2021)*, pages 211-215
ISBN: 978-989-758-559-3

From the literature mentioned above and other studies, it is undoubtedly clear that the developed models' performances depend on the variety of data mining algorithms and the dataset used and differ from one study to another.

On the other hand, road accidents are a very complex issue involving different dimensions of driving, such as the driver, the vehicle, and environmental dimension. (Aljanahi et al., 1999), In an attempt to understand this human-machine system, our team has performed an interpretation framework integrating multiple dimensions influencing the driver. Although driver inputs are often thought to be a significant cause. (Treat et al., 1979) indicated that approximately 90% of road accidents were due to driving errors, demonstrating the importance of the driver's inputs in the accident occurrence process. Indeed, our previous study (Ameksa et al., 2020) found that the vehicle data was the primary focus of the researchers in terms of data collection, including data provided by drivers such as acceleration, braking, steering wheel.

Therefore, the purpose of this study is to investigate and provide some insights into the power of the supervised learning model, especially the ensemble machine learning (ML) techniques, to predict crashes using driver response features. It should be noted that those features were used as inputs so the model can predict the occurrence of a collision based on the initial information provided from a driving simulator database.

The rest of the paper is structured as follows: Section 2 present the dataset and the methodology used in model development. Section 3 highlights significant findings and discussion. The final section outlines the conclusions and presents the limitations and recommendations for future studies.

2 DATA AND METHOD

2.1 Data Description

The fact that it is very dangerous to conduct trials in a real road environment and knowing that a driving experience on a simulator is an excellent tool for collecting data in a safe environment. The database used in this work is extracted from the outputs of a driving simulator located at Cadi Ayyad University UCA. Indeed, a group of volunteers participated in an experimental study in which their driving data was continuously recorded while they drove and followed traffic rules as they usually do in real-life situations.

It is worth noting that participants viewed the simulation on a 27-inch LCD screen with a resolution of 1920x1080 pixels and heard through a surround speaker system. The computer was equipped with a Logitech® G27 Racing system.

For this study, we are interested in the driver's inputs data such as braking, steering wheel, and speed being independent variables and the crash state as the target variable. And regarding the size of the database, it contains about 10500 rows.

2.2 Modelling

In this study, we used ensemble-based methods, including the GBM gradient boosting machine (Friedman, 2001), the XGB extreme gradient boosting (Chen and Guestrin, 2016), and RF random forest (BREIMAN, 2001) to address the aim of this paper. The following should be highlighted, the three algorithms used in this study are principally based on two techniques, bagging, and boosting methods. The RF algorithm is based on the first technique (bagging), while the GBM and XGB algorithms are built based on the second one (boosting). Both bagging and boosting are ensemble methods, where a collection of individual weak learners are combined to produce a strong learner who performs better than a single one. The difference between them is in the training phase. For the bagging technique, each model is built in parallel and independently, while boosting improves the performance of the model by adding new models that work well where previous models have failed; it creates new learners sequentially. The final decision is made for both methods by averaging the N learners. However, it is a Weighted Average for Boosting; and an Equal Weighted Average for Bagging, with higher weights for the better performing learners on the training data. (Bari et al., 2020).

Concerning the methodology adopted in this work includes the two significant steps:

1) The development of ensemble-based algorithms was carried out using the default hyperparameters. This first step was achieved through the process described below for each algorithm (example of RF algorithm, see the Appendix):
   - Import the libraries and the data file.
   - Defined Independent variables and the target.
   - Split the dataset into 70% for the training set and 30% for the test.
   - Creating an instance of the model.
   - Evaluating the model using the test data.
2) The tuning of hyperparameters was performed using the random search[1] (see Appendix) and grid

[1] Select random combinations to train the model

search [2] (see Appendix) technique for each algorithm, and by setting up a grid of hyperparameter values for the most powerful (the Appendix illustrated the most powerful hyperparameters for the RF algorithm). This process helped us to obtain the optimum hyperparameters, and subsequently, the best model for each algorithm. This is the most challenging part of the machine learning model-building process.

To evaluate the performance of the crash prediction model based on the test dataset, the precision, recall, and F1 score (scikit-learn) are used to measure the accuracy of the algorithms.

3 RESULTS AND DISCUSSION

Analysing the results of the three following algorithms RF, GBM, and XGB (Figure 1) indicates that the random forest provides better performance than the other ensemble methods GBM and XGB for the default configuration. To be more precise, 93% was the accuracy of RF and 82% for both GBM and XGB. Then, after hyper-parameters tuning, the accuracy score of the three algorithms was found to be similar and equals 95%.

The random search optimization method is used to randomly select a combination of hyperparameters among all ranges given at the beginning. In our case, XGB was found to be the best model to predict crashes using this method with an accuracy of 93%.

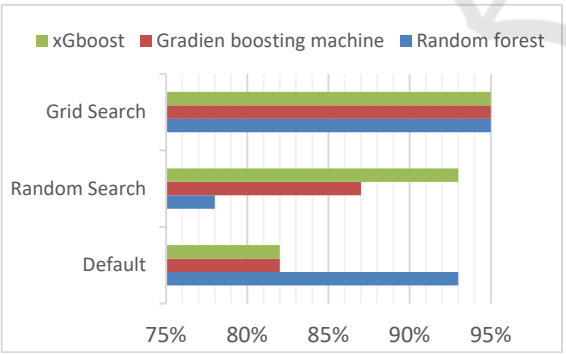

Figure 1: The accuracy of the three developed models

Since we are forecasting the occurrence of an accident or not, our problem is a binary classification case. And comparing the performance indicator has a great benefit on the selection of algorithms. Therefore, after the hyperparameters tuning process,

[2] Train every combination of hyperparameter values on a model, and select only the one with the highest score.

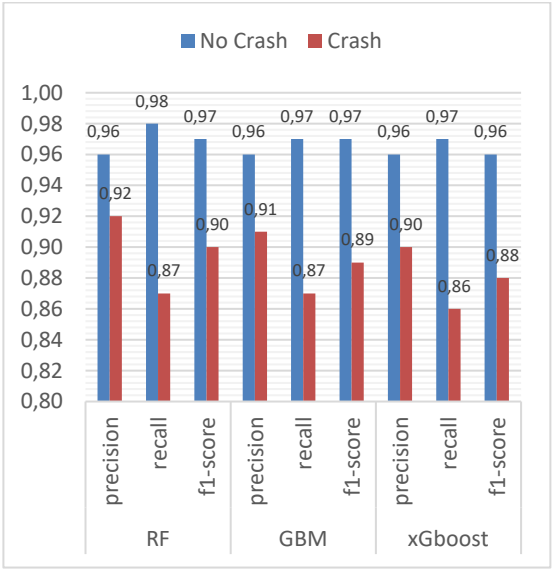

Figure 2: The performance indicators of RF, GBM and XGB after hyperparameters tuning

the other performance benchmarks presented in section 2.2 have been calculated, and the results are illustrated in Figure 2.

The precision score of the developed models had a slight difference between them in terms of accident class (i.e., 0.90, 0.91, and 0.92 for XGB, GBM, and RF, respectively). And for the no accident class, the algorithms have the same precision. For the recall score, we noticed that for both developed models based on GBM and XGB had a similar score of 0.97 for the no-crash class, 0.87 and 0.86 were recorded for GBM and XGB respectively regarding the crash class, while the RF model proved to be the best with a score of 0.98 for the no crash class and 0.87 for the class with crashes (see Figure 3).

		Defaults hyperparameters			Tuned hyperparameters		
		precision	recall	f1-score	precision	recall	f1-score
Random Forest	No Crash	0.94	0.97	0.96	0.96	0.98	0.97
	Crash	0.91	0.80	0.85	0.92	0.87	0.90
	accuracy	93%			95%		
Gradient boosting Machine	No Crash	0.82	0.98	0.90	0.96	0.97	0.97
	Crash	0.85	0.30	0.45	0.91	0.87	0.89
	accuracy	82%			95%		
Gradient boosting Machine	No Crash	0.82	0.98	0.89	0.96	0.97	0.96
	Crash	0.82	0.28	0.41	0.90	0.86	0.88
	accuracy	82%			95%		

Figure 3: Entire results of 3 algorithms for both configurations

4 CONCLUSIONS

Crash prediction offers a proactive solution to increase road safety and save lives. For this reason, it has been a long-standing interest of researchers, industry, and policymakers. However, crash prediction remains complex and requires high resolution and large data sets to develop powerful models that effectively predict accidents. According to the scientific literature, different approaches have been used to address this topic. Although, as far as the authors know, few works have been focused on investigating crash prediction based on driver inputs and ensemble methods.

In the present work, the performance of ensemble methods machine learning algorithms has been assessed in a classification case. in fact, the developed models in this research consist of predicting the occurrence of a crash from the simulator-based dataset of some driver inputs as features.

Results show that the Random Forest outperforms the gradient boosting machine and extreme gradient boosting for the default parameters. But after the optimization of hyperparameters, it was noticed that even if the algorithms are not the same, their performances are almost equal. Therefore, tuning hyperparameters impacts the machine learning developed model and offers the highest improvement and benefit in accident prediction accuracy.

One of the limitations of this paper is that the comparison is made only between ensemble methods-based algorithms. Thus, future work should include more algorithms, as well as integrating other variables such as the vehicle kinematics and the surrounding environment. In addition, the necessity for having a large, and comprehensive training data sets presents a clear challenge that should be mitigated to ensure that machine learning applications remain accurate.

ACKNOWLEDGEMENTS

The Moroccan Ministry of Equipment, Transport, and Logistics; and the Moroccan National Centre for Scientific and Technical Research (CNRST) are gratefully acknowledged for their valuable support of this research.

REFERENCES

Abdelwahab HT, Abdel-Aty MA. Development of artificial neural network models to predict driver injury severity in traffic accidents at signalized intersections. Transp Res Rec 2001:6–13. https://doi.org/10.3141/1746-02.

Aljanahi AAM, Rhodes AH, Metcalfe A V. Speed, speed limits and road traffic accidents under free flow conditions 1999;31:161–8.

Ameksa M, Mousannif H, Al Moatassime H, Elamrani Abou Elassad Z. Toward Flexible Data Collection of Driving Behaviour. ISPRS - Int Arch Photogramm Remote Sens Spat Inf Sci 2020; XLIV-4/W3-2020:33–43. https://doi.org/10.5194/isprs-archives-xliv-4-w3-2020-33-2020.

Bari D, Ameksa M, Ouagabi A. A comparison of datamining tools for geo-spatial estimation of visibility from AROME-Morocco model outputs in regression framework. Proc - 2020 IEEE Int Conf Moroccan Geomatics, MORGEO 2020 2020. https://doi.org/10.1109/Morgeo49228.2020.9121909.

BREIMAN L. Random forests. Random For 2001;45:5–32. https://doi.org/10.1201/9780429469275-8.

Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., vol. 13-17- Augu, 2016, p. 785–94. https://doi.org/10.1145/2939672.2939785.

Friedman JH. Greedy function approximation: A gradient boosting machine. Ann Stat 2001;29:1189–232. https://doi.org/10.1214/aos/1013203451.

Lagarde E. Road traffic injuries. Encycl Environ Heal 2020:572–80. https://doi.org/10.1016/B978-0-444-63951-6.00623-9.

Osman OA, Hajij M, Bakhit PR, Ishak S. Prediction of Near-Crashes from Observed Vehicle Kinematics using Machine Learning. Transp Res Rec 2019. https://doi.org/10.1177/0361198119862629.

scikit-learn. classification report n.d. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html.

Silva PB, Andrade M, Ferreira S. Machine learning applied to road safety modeling: A systematic literature review. J Traffic Transp Eng (English Ed 2020;7:775–90. https://doi.org/10.1016/j.jtte.2020.07.004.

Treat JR, Tumba NS, McDonald ST, Shinar D, Hume RD, Mayer RE, et al. Tri-Level Study of the Causes of Traffic Accidents: An overview of final results. Proc Am Assoc Automot Med Annu Conf 1979;21:391–403.

Zouhair EAE, Mousannif H, Al Moatassime H. Towards analyzing crash events for novice drivers under reduced-visibility settings: A simulator study. ACM Int. Conf. Proceeding Ser., 2020. https://doi.org/10.1145/3386723.3387849.

Abdelwahab HT, Abdel-Aty MA. Development of artificial neural network models to predict driver injury severity in traffic accidents at signalized intersections. Transp Res Rec 2001:6–13. https://doi.org/10.3141/1746-02.

Aljanahi AAM, Rhodes AH, Metcalfe A V. Speed, speed limits and road traffic accidents under free flow conditions 1999;31:161–8.

Ameksa M, Mousannif H, Al Moatassime H, Elamrani Abou Elassad Z. Toward Flexible Data Collection of

Driving Behaviour. ISPRS - Int Arch Photogramm Remote Sens Spat Inf Sci 2020;XLIV-4/W3-2020:33–43. https://doi.org/10.5194/isprs-archives-xliv-4-w3-2020-33-2020.

Bari D, Ameksa M, Ouagabi A. A comparison of datamining tools for geo-spatial estimation of visibility from AROME-Morocco model outputs in regression framework. Proc - 2020 IEEE Int Conf Moroccan Geomatics, MORGEO 2020 2020. https://doi.org/10.1109/Morgeo49228.2020.9121909.

BREIMAN L. Random forests. Random For 2001;45:5–32. https://doi.org/10.1201/9780429469275-8.

Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., vol. 13-17- Augu, 2016, p. 785–94. https://doi.org/10.1145/2939672.2939785.

Friedman JH. Greedy function approximation: A gradient boosting machine. Ann Stat 2001;29:1189–232. https://doi.org/10.1214/aos/1013203451.

Lagarde E. Road traffic injuries. Encycl Environ Heal 2020:572–80. https://doi.org/10.1016/B978-0-444-63951-6.00623-9.

Osman OA, Hajij M, Bakhit PR, Ishak S. Prediction of Near-Crashes from Observed Vehicle Kinematics using Machine Learning. Transp Res Rec 2019. https://doi.org/10.1177/0361198119862629.

scikit-learn. classification report n.d. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html.

Silva PB, Andrade M, Ferreira S. Machine learning applied to road safety modeling: A systematic literature review. J Traffic Transp Eng (English Ed 2020;7:775–90. https://doi.org/10.1016/j.jtte.2020.07.004.

Treat JR, Tumba NS, McDonald ST, Shinar D, Hume RD, Mayer RE, et al. Tri-Level Study of the Causes of Traffic Accidents: An overview of final results. Proc Am Assoc Automot Med Annu Conf 1979;21:391–403.

Zouhair EAE, Mousannif H, Al Moatassime H. Towards analyzing crash events for novice drivers under reduced-visibility settings: A simulator study. ACM Int. Conf. Proceeding Ser., 2020. https://doi.org/10.1145/3386723.3387849.

APPENDIX

0.1 Import libraries and data

```
In [ ]: import numpy as np
        import pandas as pd
        from sklearn.model_selection import train_test_split
        # Import random forest classifier package
        from sklearn.ensemble import RandomForestClassifier
        #for evaluation
        from sklearn import metrics
        # Import the data file
        data = pd.read_csv('dataset.csv')
        # Determine of independent variables and target variable
        X = data.iloc[:, 0:-1].values
        y = data.iloc[:, -1].values
        # split the dataset into 70% for training and 30% for the test
        X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                            test_size=0.30,
                                                            random_state=42)
```

0.2 Random Forest

0.2.1 default settings

```
In [ ]: # Create Random forest classifier
        rfC_defaut = RandomForestClassifier(random_state=42)
        # Fitting the model to the training data
        rfC_defaut.fit(X_train, y_train)
        # evaluate the random forest classifier
        # predict test data and compare it with the real one
        y_pred = rfC_defaut.predict(X_test)
        target_names = ['No Crash', 'Crash']
        print(metrics.classification_report(y_test, y_pred,target_names=target_names))
```

Optimization of hyperparameters using Grid Search and Random Search

the most important hyperparameters chosen for Random forest are:

* n_estimators = number of trees
* max_features = maximum number of features taken into account to split a node
* max_depth = maximum number of levels in each decision tree
* min_samples_split = Minimum number of samples required to split a node
* min_samples_leaf = minimum number of data points allowed for a sheet
* bootstrap = data point sampling method (with or without replacement)

```
In [ ]: # parameters #  number of trees for random forest
        n_estimators_rf = [int(x) for x in np.linspace(start = 60, stop = 80, num = 5)]
        # max number of features considered for splitting a node
        features = data.columns[:-1]
        max_features_rf = ['auto', 'sqrt', 'log2']
        # Maximum number of levels in a tree
        # in this study we have chosen a range between 1 and 3
        max_depth_rf = [int(x) for x in np.linspace(1, 3)]
        # data point sampling method
        bootstrap_rf = [True, False]
        # Minimum number of samples required to split a node
        min_samples_split_rf = list(range(2,5))
        #  minimum number of data points allowed for a sheet
        min_samples_leaf_rf= list(range(1,5))
In [ ]: # Creating a dictionary (random grid)
        rf_random_grid = {'n_estimators': n_estimators_rf,
                          'max_features': max_features_rf,'max_depth': max_depth_rf,
                          'bootstrap':bootstrap_rf,
                          'min_samples_split': min_samples_split_rf,
                          'min_samples_leaf': min_samples_leaf_rf,  }
rf_random_search = RandomizedSearchCV(estimator = RandomForestClassifier(),
                                      param_distributions = rf_random_grid,
                                      n_iter = 10,cv = 3,verbose=2,
                                      scoring='accuracy')

# Fitting the model to the training data
rf_random_search.fit(X_train, y_train)
# Display the selected hyperparameter values
print(grid_search_forest.best_params_)
# evaluation of the Random Forest model with the chosen parameters
evaluate_Classifier(grid_search_forest)
```

Grid Search

* Since Grid Search takes a long time we tried to do the optimization as follows:
* first we optimized (n_estimators) and (max_features)
* use the optimal values of n_estimators and max_features to optimize min_samples_split and min_samples_leaf
* Use the four precursors to optimize the rest of the parameters.

```
In [ ]: # Optimize n_estimators and max_features, then use the optimal values
        # chosen to optimize the others
        param_grid_rf = [{'n_estimators': n_estimators_rf,
                          'max_features': max_features_rf, }]
        # model creation
        grid_search_forest = GridSearchCV(RandomForestClassifier(), param_grid_rf,
                                          cv=3, scoring='accuracy')
        # Fitting the model to the training data
        grid_search_forest.fit(X_train, y_train)
        # Display the selected hyperparameter values
        print(grid_search_forest.best_params_)
        # evaluation of the Random Forest model with the chosen parameters
        # by Grid search method
        evaluate_Classifier(grid_search_forest)
```