Boosting Methods for Predicting Firemen Interventions

1st Selene Cerna
Femto-ST Institute,
University of Bourgogne Franche-Comté, CNRS
Belfort, France
selene_leya.cerna_nahuis@univ-fcomte.fr

2nd Christophe Guyeux
Femto-ST Institute,
University of Bourgogne-Franche-Comté, CNRS
Belfort, France
christophe.guyeux@univ-fcomte.fr

3rd Héber H. Arcolezi
Femto-ST Institute,
University of Bourgogne-Franche-Comté, CNRS
Belfort, France
heber.hwang_arcolezi@univ-fcomte.fr

4th Guillaume Royer
SDIS 25
Besançon, France
guillaume.royer@sdis25.fr

2020 11th International Conference on Information and Communication Systems (ICICS) 978-1-7281-6227-0/20/$31.00 ©2020 IEEE 10.1109/ICICS49469.2020.239488

***Abstract*—Forecasting future incidents to the next hours is of great importance for fire brigades, which allows improving their response time to interventions, one direct cause to guarantee proper attendance to victims. Moreover, for many departments around the world, it exists problems such as the high increment of interventions through the years, which requires more personnel and machinery resources. However, due to budget limitations fire brigades have to face this increment with the same resources. Therefore, the objective of this paper is to implement and compare three boosting methods in the specific task of predicting the number of firemen interventions in the next hour. A dataset with specific temporal information of interventions from 2006-2018 was provided by the department fire and rescue SDIS25 in Doubs-France. Great efforts were concentrated on arranging and collecting more data (e.g., meteorological data, road traffic conditions). As it is presented in this work, they were preprocessed and learned in a supervised way. As shown in results, such methods are mature enough to provide a good solution with an acceptable margin of error for real-life implementations.**

***Index Terms*—XGBoost, AdaBoost, Gradient Boosting, Firemen, Prediction**

I. Introduction

During the last years and with the help of the technology, several fire departments around the world have been collecting detailed information about their interventions, engines and human resources used in order to monitor their process. With the goal of providing a better service to society and improve their strategies in the administration of supplies, incident detection, and support request, the collected information could be used as a principal source to build a tool with the main objective of forecasting the number of interventions for a certain future time.

On the other hand, machine learning techniques such as decision trees, which are schematic representations of alternatives, facilitate making decisions. They are the basis for boosting methods, which their goal is to solve for net error from the prior tree while adding trees sequentially. The first successful boosting algorithm developed for binary classification was AdaBoost, used in researches such as the detection of fire smoke using a robust AdaBoost classifier [1] and the forecast of accidents on road traffic by a trichotomy AdaBoost algorithm [2]. Another boosting method that stands out is the Gradient Boosting, which tweaks residual errors generated by a previous weak learner in each iteration. This method was used in a new methodology to predict the severity of a traffic accident, presented in [3], where the Gradient Boosting is employed to measure the weights of the features of a traffic accident, that later will be converted in a grey image to be the input of a CNN (Convolutional Neural Network). Furthermore, one of the most remarkable boosting methods available nowadays is the Extreme Gradient Boosting (XGBoost), due to its fast construction of trees and the penalization of the models' complexity during the training phase. In [4], the authors proposed an approach based on XGBoost to predict urban fire accidents using ten million samples as data set, an algorithm based on association rules to select features, as well as the Box-Cox transformation to clean outliers. What is more, in [5] the three aforementioned techniques with the random forest method are compared together and implemented to forecast the future driving risk of crash-involved drivers.

Finally, in the literature, few studies are related to the forecast of firemen interventions. While in [6], [7] neural network models (Multilayer Perceptron and Long Short-Term Memory) are applied to this task, in [8], the machine learning algorithm XGBoost with default parameters is used to compare different anonymized datasets. The tree boosting approach has revealed to be the best compromise between performance and accuracy. However, XGBoost is not the only tree boosting method of the literature, thus it is needed a deeper investigation of its competitors, such as AdaBoost and Gradient Boosting.

This is why the present work makes a performance comparison of these three boosting techniques, namely AdaBoost, Gradient Boosting, and XGBoost, to explore more deeply the ability of machine learning techniques to achieve predictions in

this context. And, to evaluate the one who better fits to future implementation in a real data-driven system. For this task, the department fire and rescue SDIS25 located in Doubs-France has provided us with a dataset about interventions attended from 2006 to 2018. Therefore, this main data and acquired ones from external sources that can influence one intervention (e.g., traffic accidents can be linked to weather conditions) were appropriately manipulated to apply supervised learning.

This article is structured as follows: Section II-A presents the way in which the data was collected, the extraction of features and how the instances were organized. Section II-B describes brief definitions of the three boosting methods used. Section III shows and analyzes the results of the predictions. Lastly, in Section IV, the final conclusion, and future work are provided.

II. Materials and Methods

A. Data Preprocessing

1) Gathering Data: The principal source was provided by the fire brigade SDIS 25 localized in the department of Doubs, France. It contains detailed information about the interventions attended from 2006 to 2018, considering the time and date of the initiation and termination of the intervention.

The collecting of data followed the process presented in [6], the initial part of Section III and the entire subsection A. To this work, several changes were made as it is described below:

- The dictionary keys were established from the time "01/01/2006 00:00:00" to "31/12/2018 23:00:00", to include interventions from 2006-2018. Therefore, the values of each existed explanatory variable in [6] were extended to complete the keys.
- From the height of the six most important rivers in the region, it was taken the average, the standard deviation and the maximum high of the records captured in one hour. Moreover, an alert variable with value 1 for the non-normal increase in the river height or 0 if it is kept under the limit was added too. The source used was [9].
- Data like the distance to the moon was considered to analyze its effects on the generation of incidents like natural phenomena.
- The festivities dates such as the Ramadan month, Percée du Vin Jaune, the Eurockéennes and the International University Music Festival (FIMU) of the last twelve years were collected to be included as indicators in the dictionary with value 1 when it is the festivity day, and 0 for a normal day.
- Air pollution variables such as PM2.5 fine particles, PM10 fine particles, ozone and nitrogen oxide of air pollution statistics were taken from the Besançon Prévoyance and Montbéliard Centre stations [10] for years 2017 and 2018. The reminder years were completed with the first value of the year 2017 to not destroy the sequence of the data distribution.
- Additionally, the date February 29 of leap years was eliminated to not consider the impact on the "day in the year" variable.

2) Extracting Features: To enrich our data set, features of each explanatory variable were extracted using the Scikit-learn library [11]. The standard scaler method, that normalizes distributions by removing the average and rescaling to unit variance, was used for numerical variables such as: year, hour, temperature, nebulosity, dew point, wind direction, humidity, moon phase, moon distance, precipitations, bursts, visibility, wind speed, acute diarrhea statistics, influenza statistics, chickenpox statistics, rivers height variables (except by the alert variable) and air pollution variables. In order to recognize changes through the time, these variables were divided into a learning set and testing set, depending on the year to be predicted. It was used the learning set to calculate the mean and standard deviation, and the data transformation was applied to both sets.

The One Hot Encoder method, that codifies the variable in a binary vector, where one attribute is equal to one and the other values are zero, was fitted with each completed variable distribution to generate all the categories. It was employed with categorical variables such as: time variables, bison futé variables, holiday indicator, night indicator, river height alert variable, barometric trend, and festivities indicators.

Finally, it was obtained 840 features and the target feature "number of interventions" was not standardized.

3) Structuring Features: It was considered a windowing using the number of interventions of the 169 past hours (one week plus one hour of history) since there is a high correlation with the number of incidents recorded in the last week. Furthermore, the data were structured sequentially, i.e., the target feature represents the number of interventions that will be attended in the next hour $(t+1)$, considering a present sample (t).

B. Boosting Methods

Boosting is an ensemble technique that trains predictors one after another, trying to correct the previous one. In the search of the best results, AdaBoost, Gradient Boosting and XGBoost, which are efficient and well-known techniques, are applied to our problem to evaluate their performances and outcomes.

1) AdaBoost: Short for Adaptive Boosting. It increases the relative weights of the misclassified instances at every iteration to tweak them with the predictor of the next iteration [12], as it is explained in [13]. Initially, all instance weights start with values $\frac{1}{m}$, where m is the number of training instances. In the following iterations, each weight is updated according to Eq. 1, where $w^{(i)}$ is the i^{th} training instance weight, $\hat{y}_j^{(i)}$ is the prediction of the i^{th} sample using the j^{th} predictor, $y^{(i)}$ is the real target value and α_j is the predictor weight. If the predictor is more accurate, the greater its weight. And if it is less accurate, its weight will be negative. Then, all weights are standardized and the process is repeated until the best predictor is found. To compute predictions, each weak classifier makes predictions that will be weighted using the predictor weight α_j. The resulting class is the one that has the majority of weighted votes. This is represented in Eq. 2, where x is the instance to be predicted, N is the number of weak classifiers

and k is the prediction of the j^{th} predictor. In practice, the *AdaBoostRegressor* method from the Scikit-learn library [11], [14] has been used in this study.

$$w^{(i)} \leftarrow \begin{cases} w^{(i)}, & \text{if } \hat{y}_j^{(i)} = y^{(i)} \\ w^{(i)} \exp(\alpha_j), & \text{if } \hat{y}_j^{(i)} \neq y^{(i)} \end{cases} \quad (1)$$

$$\hat{y}(x) = \underset{k}{\operatorname{argmax}} \sum_{\substack{j=1 \\ \hat{y}_j(x)=k}}^{N} \alpha_j \quad (2)$$

2) Gradient Boosting: It takes the residual errors made by the previous weak learner, which is a decision tree, to fit the new predictor, improving the model at each iteration. Thus, the prediction for each sample results on adding up the predictions of all the trees used [13], [15]. As it is described in [11] documentation and [16], Eq. 3 depicts the additive strategy with the created model F_m in the iteration m, where F_{m-1} is the previous ensembled model, h_m is the new tree added that is built in Eq. 4 while tries to reduce the loss L, γ_m is the step length chosen by using line search in Eq. 5, x_i is the i^{th} instance and y_i is the target value of the i^{th} instance. The first model F_0 is the mean of the target values when the least-squares regression is used as loss function. In the present work, the *GradientBoostingRegressor* method from the Scikit-learn library [11] has been used.

$$F_m = F_{m-1} + \gamma_m h_m \quad (3)$$

$$h_m = \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h(x_i)) \quad (4)$$

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) - \gamma \frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}) \quad (5)$$

3) XGBoost: Stands for Extreme Gradient Boosting. It uses a new perspective compared to the previous ones to minimize the complexity of the model in each iteration. XGBoost establishes an objective function considering the loss function $L(\theta)$ and the regularization $\Omega(\theta)$ on the model, penalizing its complexity to avoid overfitting, as can be seen in Eq. 6, where θ means the parameters found during the training. The model $\hat{y}_i^{(t)}$ at iteration t is the combination of k trees (Eq. 7), i.e., a new tree that optimizes the system $f_t(x_i)$ is added to the model $\hat{y}_i^{(t-1)}$ created in the previous round, where x_i is the input instance. To evaluate the complexity of the tree $\Omega(f)$, [17] presented an approach depicted in Eq. 8. The first term γT evaluates the number of leaves T, where γ is a constant, and the second term calculates $L2$ norm of leaves scores w_j. Taking the Mean Squared Error (MSE) as instance for the loss function and computing its Taylor expansion to the second order, the objective function outcomes in Eq. 9 and describes how the partition of the nodes is done. G and H are defined as Eq. 10 and Eq. 11 respectively, where g_i and h_i are the first and the second order partial derivative after applying the Taylor expansion, $I_j = \{i|q(x_i) = j\}$ are the samples assigned to the j-th leaf and $q(x)$ is the tree structure. Lastly, from the objective function, the argument of the minimum and the minimum of the quadratic function for the variable w_j are taken, where $q(x)$ is fixed and λ is a small constant, the outcomes are Eq. 12 and Eq. 13, where the latter evaluates the score of the tree structure, i.e., if it is smaller, it is better [17]. In practice, the library built by the author [17] for python language was used in this research work.

$$obj(\theta) = L(\theta) + \Omega(\theta) \quad (6)$$

$$\widehat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \widehat{y}_i^{(t-1)} + f_t(x_i) \quad (7)$$

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \quad (8)$$

$$obj^{(t)} = \sum_{j=1}^{T} [G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2] + \gamma T \quad (9)$$

$$G_j = \sum_{i \in I_j} g_i \quad (10)$$

$$H_j = \sum_{i \in I_j} h_i \quad (11)$$

$$w_j^* = -\frac{G_j}{H_j + \lambda} \quad (12)$$

$$obj^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \quad (13)$$

III. Results and Analysis

Many efforts have been concentrated on the acquisition of information, treatment, and construction of a set of data from past interventions and various integrated sources, taking into consideration the significant contribution of each variable. Hence, as previously mentioned, the purpose of this work is to continue exploring the efficacy of different machine learning techniques and future implementation of the one who better fits the problem in a real data-driven system.

To evaluate the performance of the three methods, the two last years (2017 and 2018) were selected as the testing set independently, where each year represents 8760 samples. The first division of the data is made considering 2006-2017 as the training set, and 2018 as the testing set. Similarly, for forecasting the year 2017, the data from 2006-2016 were used as the training set and the year 2018 was not considered in this case.

The metrics used to evaluate the models' prediction were the prediction execution time in seconds (Tsec), the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). Moreover, three accuracy scores are considered: a margin of error zero (A0), a margin of error less or equal to one (A1) and two (A2). The A0 represents the accuracy of

the exact number of predictions reached. To run the codes, a machine with a Titan X, Intel(R) Xeon(R) CPU E5-2623 v4 @ 2.60GHz with 64Gb of Ram and with a GPU of 3,072 cores and 12Gb of Ram was used.

Moreover, for comparison purposes, it is assumed as a baseline the best result achieved for the next hour in the paper [6] with the Long Short-Term Memory neural network for year 2017, which is a MAE of 1.69, RMSE of 2.30 and with 55.41% of correct predictions with a margin of error zero or equal to one.

All three models were tuned via a grid search procedure, which is a technique that tests every combination of a set of attributes to find the best solution. Thereby, Table I presents the hyperparameters found and used for each method to achieve its respective better results. Table II are presented metrics results for each method to both 2017 and 2018 years, where the numbers in bold indicate the overall best result. Furthermore, Figures 1 and 3 illustrate forecasting results for all three methods, zooming in 175 samples; and Figures 2 and 4 exhibit in a bar plot the comparison to the exact number of predictions with errors from 0 to 14.

As can be seen in Table I, unlike the other methods, the best AdaBoost model used fewer estimators and a greater tree depth, which allowed to reach a greater number of exact predictions (A0). However, for both years, AdaBoost model was the method that consumed the largest time in prediction with 0.91 seconds and higher MAE and RMSE metrics as shown in Table II, which implies that high numbers of interventions were not well recognized by the method, this is shown in Figures 1 and 3.

On the other hand, Gradient Boosting took a sample of 80% of the instances (Instance sampling = 0.8) and analyzed all the features (Feature sampling = 1.0) before selecting any for a tree construction in each iteration. Besides, it used the least-squares regression as a loss function, that allowed analyzing and minimizing residual errors from previous predictions. As shown in Figures 1 and 3, the best Gradient Boosting model reached more peaks than AdaBoost, what is more, from the three methods, it was the one that consumed less time when performing the prediction for both years, it showed an MAE equal to the baseline for the year 2017 and a higher A1 for the mentioned year. Also, in Figures 2 and 4, Gradient Boosting presented a larger number of predictions with a margin of error two, i.e., for 2017, it reached 1856 exact predictions, while AdaBoost and XGBoost models obtained 1804 and 1790 respectively; whereas for 2018, AdaBoost achieved 1839 and XGBoost 1950 exact predictions, being surpassed by Gradient Boosting with a value of 1953 exact estimates, taking into consideration an error of ± 2 interventions when predicting the number of incidents attended in one hour.

TABLE I: Hyperparameters

Hyperparameter	AdaBoost	Gradient Boosting	XGBoost
Estimators	50	100	200
Maximum tree depth	7	5	5
Learning rate	0.05	0.04	0.02
Instance sampling	1.0	0.8	1.0
Features sampling	0.8	1.0	0.9
Loss/Objective	linear	least squares regression	count:poisson

TABLE II: Prediction results for 2017 and 2018

Year	Technique	TSec	MAE	RMSE	A0 (%)	A1 (%)	A2 (%)
2017	AdaBoost	0.91	1.71	2.35	20.88	55.79	75.38
	GradientB.	0.16	1.69	2.29	20.55	55.66	76.85
	XGBoost	0.24	**1.68**	**2.28**	20.76	**56.47**	**76.91**
2018	AdaBoost	0.91	1.85	2.60	19.20	53.04	74.02
	GradientB.	0.06	1.81	2.50	18.78	52.68	74.98
	XGBoost	0.24	**1.80**	**2.50**	19.02	**53.08**	**75.34**

However, from the three methods, XGBoost presented a better performance in the recognition of the high numbers of interventions over time, which is visualized more clearly in the predictions for the year 2018, Figure 3, where it reached a peak of 12 interventions. Also, for 2017, XGBoost overcame the baseline metrics generated by the LSTM [6] with an MAE of 1.68 and an A1 of 56.47%. For both years, the metric A2 of the XGBoost model stood out among the models. It should be noted that the model generated by XGBoost used the lowest learning rate and the highest number of estimators, considering the objective function "count:poisson" that is used for data counting, i.e., it takes a distribution of non-negative integer values such as the number of interventions at a certain hour.

Finally, it can be noticed in Table II that forecasting results for 2018 decreased performance comparing to 2017. Such result can be explained by the fact that, in 2017, the total number of interventions was 37,710 with 4.3 interventions per hour on average and standard deviation of 2.95, while for 2018 the total number of interventions increased to 40,957 with 4.68 interventions per hour on average and standard deviation of 3.23.

IV. Conclusion

The ability to predict the number of interventions in the next hour is a more precise scenario that fire brigades could take advantage of. That is if it is known that for the next hour will occur a big number of events and at the present moment the team is not complete or machinery resources have been assigned for other incidents, better strategies can be implemented for splitting up teams and machinery allocation for the next incidents.

In this context, this paper proposed to investigate three machine learning techniques, namely AdaBoost, Gradient Boosting, and XGBoost, to the particular task of forecasting the future number of firemen interventions. In the literature, these techniques have proven to be very effective in modeling high nonlinear behaviors. Previous efforts were made to prepare a dataset with specific information about interventions from 2006-2018 in the department of Doubs-France, taking into account external sources such as meteorological data, traffic conditions, epidemiological data, air pollution, festivities and many more.

As shown in results, forecasting the number of interventions for the next hour with boosting techniques is possible for

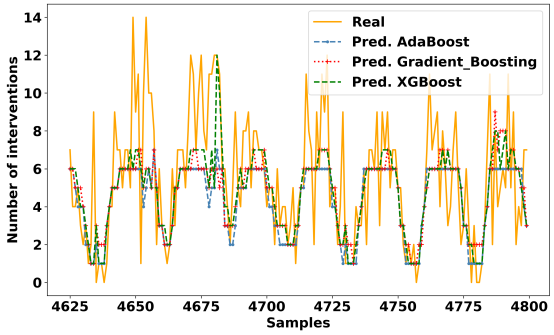

Fig. 1: Predictions for 1h - 2017

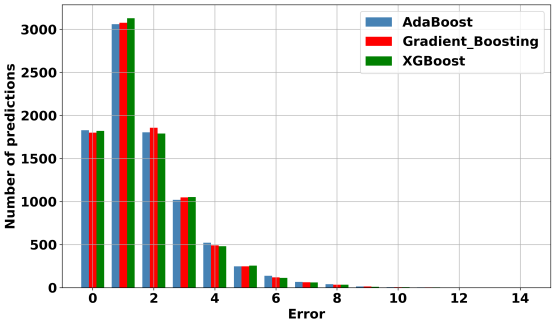

Fig. 2: Exact predictions for 1h - 2017

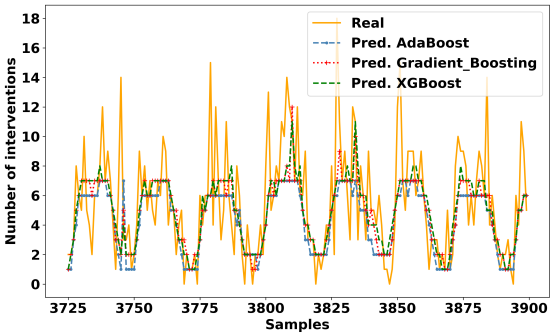

Fig. 3: Predictions for 1h - 2018

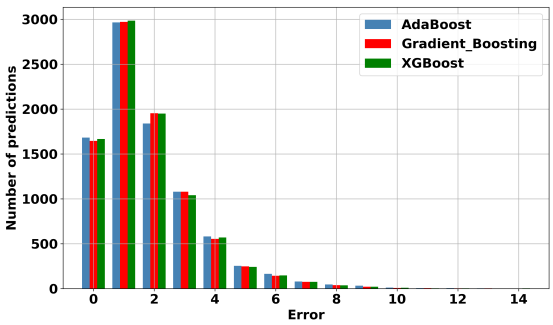

Fig. 4: Exact predictions for 1h - 2018

practical implementations. That is, decision trees methods as those applied in this paper have attributes such as simplicity, robustness, less time consumption and less computational costs, which are well appreciated for real-time applications. Moreover, at present, the results show a fairly acceptable margin of error for helping a decision-making approach for fire brigades.

For future work, it is planned different approaches to rearrange the dataset, include more explanatory variables and categorize the events, expand the prediction time, locate the incidents and continue testing and proposing algorithms for solving such a important social problem.

Acknowledgment

This work was supported by the EIPHI Graduate School (contract "ANR-17-EURE-0002"), by the regional council of Franche-Comté, by the Interreg RESponSE project, and by the SDIS25 firemen brigade.

References

[1] X. Wu, X. Lu, and H. Leung, "A video based fire smoke detection using robust AdaBoost," *Sensors*, vol. 18, no. 11, p. 3780, Nov. 2018. [Online]. Available: https://doi.org/10.3390/s18113780

[2] H. Zhao, H. Yu, D. Li, T. Mao, and H. Zhu, "Vehicle accident risk prediction based on AdaBoost-SO in VANETs," *IEEE Access*, vol. 7, pp. 14 549–14 557, 2019. [Online]. Available: https://doi.org/10.1109/access.2019.2894176

[3] M. Zheng, T. Li, R. Zhu, J. Chen, Z. Ma, M. Tang, Z. Cui, and Z. Wang, "Traffic accident's severity prediction: A deep-learning approach-based CNN network," *IEEE Access*, vol. 7, pp. 39 897–39 910, 2019. [Online]. Available: https://doi.org/10.1109/access.2019.2903319

[4] X. Shi, Q. Li, Y. Qi, T. Huang, and J. Li, "An accident prediction approach based on XGBoost," in *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*. IEEE, nov 2017.

[5] C. Wang, L. Liu, C. Xu, and W. Lv, "Predicting future driving risk of crash-involved drivers based on a systematic machine learning framework," *International Journal of Environmental Research and Public Health*, vol. 16, no. 3, p. 334, Jan. 2019. [Online]. Available: https://doi.org/10.3390/ijerph16030334

[6] S. Cerna, C. Guyeux, H. H. Arcolezi, A. D. P. Lotufo, R. Couturier, and G. Royer, "Long short-term memory for predicting firemen interventions," in *6th International Conference on Control, Decision and Information Technologies (CoDIT 2019)*, Paris, France, apr 2019. [Online]. Available: https://doi.org/10.1109/codit.2019.8820671

[7] C. Guyeux, J.-M. Nicod, C. Varnier, Z. A. Masry, N. Zerhouny, N. Omri, and G. Royer, "Firemen prediction by using neural networks: A real case study," in *Advances in Intelligent Systems and Computing*. Springer International Publishing, Aug. 2019, pp. 541–552. [Online]. Available: https://doi.org/10.1007/978-3-030-29516-5_42

[8] J.-F. Couchot, C. Guyeux, and G. Royer, "Anonymously forecasting the number and nature of firefighting operations," in *Proceedings of the 23rd International Database Applications & Engineering Symposium on - IDEAS19*. ACM Press, 2019. [Online]. Available: https://doi.org/10.1145/3331076.3331085

[9] M. de l'Ecologie-du Développement Durable et de l'Energie, "Ministére de l'ecologie- du développement durable et de l'energie," http://www.hydro.eaufrance.fr/, 2019, online; accessed 25 September 2019.

[10] F. des Associations Agréées de Surveillance de la Qualité de l'Air, "Indices de la qualité de l'air," https://www.atmo-bfc.org/, 2019, online; accessed 25 September 2019.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[12] 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, Aug. 1997. [Online]. Available: https://doi.org/10.1006/jcss.1997.1504

[13] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*, 1st ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472: OReilly Media, 7 2017, vol. 1.

[14] H. Drucker, "Improving regressors using boosting techniques," *Proceedings of the 14th International Conference on Machine Learning*, 08 1997.

[15] L. Breiman, “Arcing the edge (technical report),” *University of California, Berkeley, CA*, 1997.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer New York, 2009. [Online]. Available: https://doi.org/10.1007/978-0-387-84858-7

[17] 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*, ser. KDD ’16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785