Workshops at 18th International Conference on Intelligent Environments (IE2022)
H.H. Alvarez Valera and M. Luštrek (Eds.)

doi:10.3233/AISE220059

Explainable Decision Tree on Smart Human Mobility

Luís Rosa [a,1], Miguel Guimarães [b], Davide Carneiro [a,b], Fábio Silva [a,b] and César Analide [a]

[a] *ALGORITMI Center, Dep. of Informatics, University of Minho, Braga, Portugal*
[b] *CIICESI, ESTG, Politécnico do Porto, Felgueiras, Portugal*

Abstract. Artificial Intelligence is a hot topic and Machine Learning is one of the most fluent approaches and practices. The problem with many AI models is that they can be useful for predicting but they are bad at explaining why they behave a certain way. In some contexts, the explanation may even be more important than the prediction itself, mainly in systems in which decisions are made based on their predictions. Therefore, it is increasingly necessary to provide a forecast accompanied by an explanation, when decisions are made automatically. This paper aims to contribute to the solution of problem based on human mobility research, or at least, to be a starting point for its solution.

Keywords. Explainable Artificial Intelligence, Machine Learning, Smart Cities, Smart Human Mobility

1. Introduction

As a step to improve the quality of life of people, many cities are developing strategies to monitor and predict how spaces are used. Over the last decade, the ever-increasing capacity of devices and network infrastructure has led to new opportunities to optimise the quality of life by tracking the digital footprint of citizens.

Collected data is interpreted as large streams of data and analysed by complex machine learning algorithms for several purposes like traffic regulation, personalized suggestions, targeted publicity, space optimization and others. The accuracy and efficiency of the models is considered good for most use cases, however, most models lack fundamental support for decision making and detailed explanation on how a determined machine learning model generates a decision. This trend is worsened with the frequent use of deep learning models which are built upon the assumption of black box models that make decision interpretability difficult.

Explainable AI (xAI), is a field of artificial intelligence which aims to deal with the process of providing cues for the decision making process and interpretability of such algorithms, preferably explaining the whole decision making process. Modern concerns such as privacy, representativity or non-discrimination also need to be monitored and enforced in the machine learning models used. Being able to interpret and explain deci-

[1]Corresponding Author: ALGORITMI Center, Department of Informatics, University of Minho, Braga, Portugal; E-mail: id8123@alunos.uminho.pt.

sions in light of these topics is also a concern of xAI. Although often incomplete in its implementation it is true that there are some standardized algorithms implementations that help xAI, mostly based on instance-based interpretability and explainability.

As many xAI algorithms proposed in the literature are focused on instance based decision interpretability they are not specialized on the relationships between data instances or temporal aspects in the decision making process. Moreover, the state of the art on timeseries machine learning algorithms are mostly based on deep learning models which, again, are not friendly to interpretability and explanation.

Our work details the use of xAI algorithms with deep learning timeseries algorithms in an effort to specialize an approach to represent and interpret the decision making process through a dedicated xAI algorithm [1].

The rest of the paper is organized as follows. In Section 2, it explores projects that use white explainable models to explain timeseries decision based on black-box models. Then, in Section 3, it discusses the methodologies and techniques to translate some timeseries theory such as data windows, inter instance dependencies and data sequences into the explainable models. Finally, this paper shows that the developed xAI algorithm contributes to data preparation and analysis of human interpretable explanations.

2. Related Work

A rapid increase in the complexity and sophistication of AI-powered systems has evolved to such an extent that humans do not understand the complex mechanisms by which AI-based systems make certain decisions. In these fully automated systems, a critical aspect is the decision making (i.e. planning and acting), based on robust and reliable detection and perception — something that is particularly challenging when these systems compute outputs that are unexpected or seemingly unpredictable. xAI has a huge potential to advance this process [2,3,4]. Additionally, it should ensure that AI technologies work to serve people and society.

Several Research and Development (R&D) actions propose ways and approaches to solve the limitations of AI systems. Supporting the development and integration of AI in human mobility with explainable, trustworthy and human-centric and unbiased concepts, techniques and models have been established for many authors. They discuss xAI for Smart Healthcare, Smart Transportation, Smart Environment, Smart Governance, and Cyber Security.

Regarding Smart Healthcare, M. Abdur Rahman, et al. wrote an article titled "Human AI Teaming for the Next Generation Smart City Healthcare Systems" [5]. Since this kind of "smart" has near-zero tolerance for inexplicability (i.e, conditions must be fully understood, and solutions must be clear and correct), it is among the most cautious sectors for adopting AI technologies. Doctors can require the system to list out the symptoms where information can be presented in different granularities.

Another xAI application is in Intelligent Transportation Systems (ITS) or Smart Transportation. One such application is autonomous vehicles (AVs), which come under the category of AI in ITS. In the paper [6], Vehicular Adhoc Networks (VANET) make communication possible between AVs in the system. The performance of each vehicle depends upon the information exchanged between AVs. False or malicious information can disrupt the whole system leading to severe consequences. Hence, the detection of malicious vehicles is of utmost importance. This work uses a particular model interface of the evaluation metrics to explain and measure the model's performance. Applying xAI to these complex AI models allows a cautious use of AI for AVs.

Governments and local authorities can also be influenced on decision-making from xAI. This [7,8] aims help of social innovation and influences decision-making in smart cities. The authors help academics and government officials better understand the need for social innovation and how it influences the interaction between artificial intelligence and smart decision-making in smart governance systems.

Another relevant area where xAI plays a vital role is Cyber Security. In the last few decades, we have embraced AI in our daily life to solve a plethora of problems, one of the most notable being cyber security. In coming years, the traditional AI algorithms are not able to address zero-day cyber attacks, and hence, to capitalize on AI algorithms, it is absolutely important to focus more on xAI. Hence, [3] serves as a reference for those who are working in cyber security and AI.

Commonly, in previous projects, xAI vastly improves the quality of decision makings and hold the respective stakeholders accountable. This translates to system requirements that can be designed, measured, and continuously tested. They will depend on the domain where they are applied. As we rely more on automated systems for making decisions, it gives us an unprecedented opportunity to be more explicit and systematic about the principles or values that guide us on how we make decisions.

3. Experimental Case Study

In order to explore xAI from the Decision Tree, we generate a dataset from several sources. Then, we apply a set of data processing techniques to address the collected data. One of the last steps in data preprocessing is to split it in a training and testing subset, describing the difference between univariate and multivariate analysis. And we train the Deep Learning model (or ”black-box” model) on the training subset, and evaluate it with an unseen test set. These stages are also some of the main contributors of this work to explain the predictions of the ”black-box” model. Additionally, other processes and methods to allow human users to comprehend and trust the predictions and output created by Machine Learning (ML) are considered in the following subtopics.

3.1. Research Environments

A script was created, in Python language, which generates a dataset from Open Data of New York City. This open data is free public data published by New York City agencies and other partners. The Department of Information Technology and Telecommunications (DoITT) manages the technical operations with Socrata Open Data API, ensuring that technological capabilities are always evolving to better meet user needs. This API allows to programmatically access a wealth of open data resources from governments, non-profits, and Non-Governmental Organizations (NGOs) around the world. In this work, we use the following resources:

- 311 Service Requests - A public service that handles all requests for government and non-emergency services, accepting a wide range of issues, including over 500 complaint types. It can help to get a pothole fixed in your neighborhood, the heating device turned on in your apartment, or a refund for an overpaid parking ticket;

- Calls for Service to NYPD's 911 system - This service uses the ICAD system which call takers and dispatchers use to communicate with callers and the New York Police Department (NYPD). Each record represents an entry into the system. The data includes entries generated by members of the public as well as self-initiated entries by NYPD Members of Service;
- LinkNYC Kiosk Status - This application provides the most current listing of kiosks, their location, and the status of the Link's WiFi, tablet, and phone;
- TLC Trip Record Data - The yellow and green taxi trip records include fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. The data is collected and provided to the NYC Taxi and Limousine Commission (TLC) by technology providers authorized under the Taxicab & Livery Passenger Enhancement Programs (TPEP/LPEP).

All our data is generated via previous sources and should be stored on a system that maintains data consistency and availability. Nonetheless, the project settings, aggregated human tracking by day and remaining steps such as data processing and analysis are explained in the next subtopics.

3.2. Data Collection

As we mentioned, our dataset is generated via public REST API services provided by the NYC government. Advantages can be pointed out such as forecasting the number of people in a given location (i.e census attribute). Other features were captured geolocation (latitude/longitude), sentiment of people (i.e, -1 - Negative, 0 - Neutral, 1 - Positive), area type (e.g. animal services, art school, bank, bus stop, coffee/pub, college/university and others), name and description. Additionally, if geolocation is associated to an event, we store the following attributes: event type (e.g. incident, Farmers Market, Theater Load in and Load Outs, Special Event, Sport - Adult and others), theme, start date, finish date, name and description.

Other information can be considered, like the weather. Our intention was to use an API so that the information gathered was even wider. This includes, for example, the OpenWeatherMap API that enables the collection of a vast amount of data associated with weather conditions such as clouds, feels like, humidity, pressure, temperature, temperature minimum, temperature maximum and speed[9]. The archived data is provided for several stations.

After selecting more relevant features for our study, we save the API information on PostgreSQL database. In addition to this procedure, we develop a REST API in Java Spring Boot (Spring Boot). With the REST API created together with Swagger[10], we can generate a CSV file with all these features. But first we aggregate the rows of original table via SQL query, reducing memory consumption and processing time. Finally, after downloading data from the API, we have our dataset.

3.3. Data Preprocessing

We extracted data from multiple sources between 1 January 2020 to 31 December 2021. This process required contemplation and planning. The first step is to identify which data we want to extract. Once we have decided on the data sources to use, and the frequency with which we perform the extraction, we come up with an estimate of how much data will be extracted during each run of the API process. But we need to combine and merge it before loading it into the target destination (a CSV file). The collected dataset contains 2254 lines and a total of 20 features. Therefore, first of all, the following steps of data processing were done:

- Data Filtering - Ignore irrelevant information by selecting only certain rows, columns, and fields from a larger dataset;
- Data Aggregation - Combine data from multiple sources so that it can be presented in a more digestible, understandable format. In this case we indicate the coordinates of New York City center, and aggregate all data based on radius 1km;
- Elimination of irrelevant variables: Some variables like the name and description area, as well as, name and description event were deleted;
- Handling of missing values: In the treatment of missing values, we applied the mean in the case of meteorology, or median for sentiment;
- Feature scaling: We frame the values between -1 and 1 using Normalization method. They aim to reduce the discrepancy between values.

Data preprocessing includes the steps we need to follow to transform data so that it may be easily parsed by the Machine Learning algorithms. The main agenda for a model to be accurate and precise in predictions is that the algorithm should be able to easily interpret the data's features. In this case, we want to study if the treated data is relevant to predict the number of people in a specific location. However, other special precautions regarding the way data had to be processed were taken, which we will detail in the next subsection.

3.4. Building the models

This step is the most important and most meticulous requirement of the entire research. At this stage, we develop an understanding of the problem which we are trying to solve. Now our data is also in its usable shape. Then, in the model selection step, we choose a Deep Learning (DL) algorithm, such as Long Short-Term Memory (LSTM) - appropriate model for timeseries and regression problems. Now it's time to select and train our Machine Learning (ML) model. To perform the execution of the model, we must indicate usual parameters of algorithm.

With this, the goal of this paper is to use univariate and multivariate analysis to understand the number of citizens on New York City center, playing an important role in solving urban problems. It can help local authorities to understand the underlying driving forces of people in cities allocating resources to improve public space efficiency. In univariate time series dataset is generally provided as a single column of data, in this study, it's "census" column. On the other hand, a multivariate time series covers several variables like census, sentiment, area type, area name, event type, event name, clouds, feels like, humidity, pressure, temperature, temperature minimum, temperature maximum and speed. Other elements such as area description, event description, event theme, event start date and event finish date can be discarded. A sample of collected data can be seen in Table 1.

Loading the dataset is a high priority requirement. Using a Python language library (pandas), we have the ability to open a CSV file and load it to a "DataFrame" object.

Table 1. Example of dataset.

Date	Census	Clouds	FeelsLike	Speed	Pressure	Temp (°C)	TempMin (°C)	TempMax (°C)
2020-02-12	12	4	250.19	2.02	4	15	13	18
...	...	...	...	...	...	...	...	...

From this object, we used a correlation matrix to analyze correlation coefficients between variables. Attributes as census, clouds, feelslike, speed, pressure, temp, tempmin and tempmax presents a high correlate with each other. Then, we enter the number of times the dataset should be divided, performing 10-fold cross-validation in the model tests. Subsequently, data is separated into training data (650 days) and test data (80 days).

In deep learning predictions and being a multiclass classification problem, the loss function is therefore categorical_crossentropy. Furthermore, in the final layer, a softmax activation function was used. Then, we apply MinMaxScaler technique to normalize our data. The final step was validating and tuning the models. In these approaches, the objective was to experiment some combinations in order to find a good fit. Once we are using a regression model, components are specified in the model as a parameter such as the number of layers and neurons, windows size, epochs and batch size. Each combination of parameters improve the results of the model.

3.5. Decision Tree API

Learning is done through Decision Tree (DT) and one of the predictive action modeling techniques, used in fields such as statistics, Data Mining (MD) and Machine Learning (ML). This approach/technique was chosen because it is naturally explainable and understood, in a certain way to imitate human thought. There are some well-known decision tree implementation variants by the scientific community, to highlight the Iterative Dichotomiser 3 (ID3), C4.5, C5.0 and Classification and Regression Trees (CART) algorithms[11].

Inspired by the ML library, scikit-learn, also used in this work, which has an optimized version of the CART algorithm, despite not yet supporting variables[12], and due to the explanatory potential of that algorithm, [1] developed an implementation of the CART algorithm. The Decision Tree script developed, by itself, is independent, since it is also in the origin of the project and if any programmer so wishes, they can download the source code from GitHub Repository (contact the authors for permission to access it) and modify it, without necessarily relying on the API. Either through the API or the execution of the Decision Tree script, even if the values are invalid, the model is trained. The user is also informed if there are possible improvements to be made in the settings.

3.6. Results

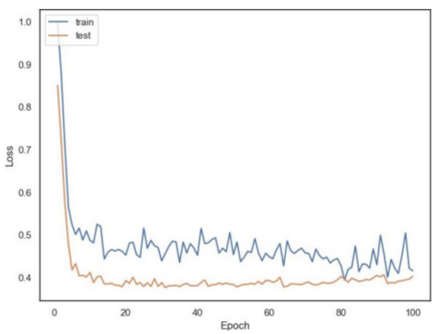
Since we want to classify the number of people in New York City some precautions have to be taken when we use a Neural Network (NN) model. The first phase of the experiment includes a dataset. Then, the models' performance is analyzed with metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Squared Error (MSE) and Mean Absolute Percentage Error (MAPE) errors for two different perceptions (i.e, univariate and multivariate time series).

Based on the Table 2, the four metrics show different performances between the proposed types of variate study. In the univariate model, the RMSE, MAE, and MSE values are higher than in the multivariate model. Additionally, MAPE value is worse in univariate than multivariate model.

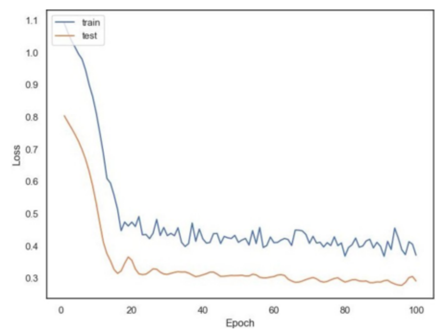
Table 2. RMSE, MAE, MSE and MAPE for Deep Learning model with univariate and multivariate time series.

	Univariate				Multivariate			
Model	RMSE	MAE	MSE	MAPE	RMSE	MAE	MSE	MAPE
LSTM	6.8	5.2	45.8	57.13	5.8	4.4	33.7	59.9

Although each metric has its own pros and cons, they are useful to address problems such as underfitting and overfitting which can lead to a poor performance on the final model despite the accuracy value. Basically, three-fold approaches for univariate time series and two-fold approaches for multivariate time series enable (i) presenting performance bounds of MAE, and (ii) demonstrating new properties of MAE that make it more appropriate than MSE as a loss function for Deep Neural Network. The quality assurance of results was only possible based on the loss functions.



(a) Univariate



(b) Multivariate

Figure 1. Univariate and multivariate quality loss functions.

Fig. 1 shows that, initially, the univariate model has good performance, after 80 epochs it tends to converge, then it degrades. In its turn, multivariate loss function presents reasonable values and acceptable to be used for prediction and forecasting human mobility, but comparing with the other type of variant it does not tend to converge. Both functions can be a good alternative for predictive modelling of human mobility. However, they differ at the level of the metrics that present the best results with one or more attributes. This means that multivariate predicted values closely match the actual values of census.

Now, in the context of human decision making based on model outcomes (i.e, number instance), metrics as the problem of predicting the number of people on New York City center, with an attempt to explain a ”black-box” model (i.e, LSTM model) with Decision Tree (DT). In other words, the next step is to try to explain the predictions of the ”black-box” model with the predictions and explanations of the Decision Tree API presented in Subtopic 3.5.

The first prediction in the test dataset in Deep Learning model gave approximately 16 people in univariate domain and 17 people in multivariate domain, equal to the real value for the first instance (see value of the last attribute of the instance in Table 3). The same instance in DT gave an approximated prediction both for the scenario when we are predicting with just one attribute or multiple attributes. Therefore, the difference can be as little as 0.08 units. The biggest difference is the fact that the DT's prediction is substantiated.

Table 3. Prediction based on Deep Learning and Decision Tree models.

	Deep Learning		Decision Tree	
	Univariate	Multivariate	Univariate	Multivariate
Number Instance	278	278	396	396
Prediction	18.8	17.2	16.22	16.76

In Fig. 2, in addition to the forecast, DT explains that the number of people (or census) is approximately 19 because the clouds is above 26. It also issues an alert if the value of census was approximately 7 units lower, the new forecast would be approximately 12, which in this case remained the same. Continuing the explanation, the census of the New York City center is 10 because the temperature normal present in the city exceeds 8.2(°C) and 14 people below this value. Still in relation to temperature above 8.2, the new forecast for a number of people approximately is 10 if pressure is above 1019. Otherwise, target value grows to 18, representing a difference of almost 8 people, which is significant and useful to know this kind of extra information about the forecast. The explanation continues until the temperature attribute, which is below 8.2(°C). In the next forecast, the census value can be influenced by temperature minimum. When it is below 7.31(°C) the forecast is aproximately 17 people. On the other hand, when it is above or equal 7.31(°C) the forecast is aproximately 13 people.

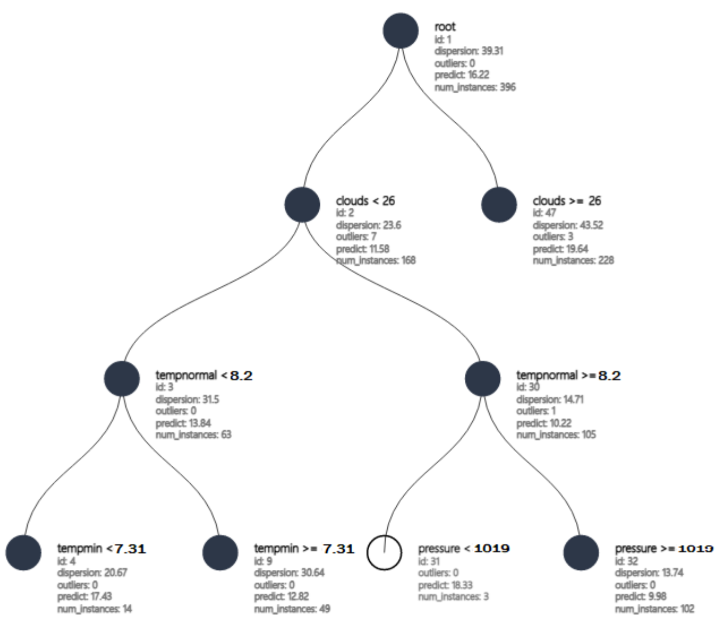


Figure 2. Decision Tree of univariate model.

In its turn, Fig. 3 shows a sample forecast path from the root node to the leaf node generated via DT from multivarite model outcome. In root, the DT predicts a value, in this case 16, for the number of people. In the next forecast, with feelslike above or equal

269.442, DT predicts a number of 21, but argues the prediction based on the feelslike values below 269 with a slight error, 3 attributes (i.e, feelslike, tempmin and speed) are used to model this particular problem. In addition to this explanation, it also generates automatic counterfactual analysis. When the value of tempmin attribute is below 7.4(°C), tree gives a tree level and predicts 14 people in center of New York City. In case of three level, feelslike attribute predicts 13 people, otherwise 18 people. In its turn, when tempmin is above or equal 7.4(°C) has also impact on census prediction, estimating 10 people. On child nodes, it should be noted that when speed value is below 0.54 the prediction of the number of people is 19. In speed value above or equal 0.54 the prediction is 10.

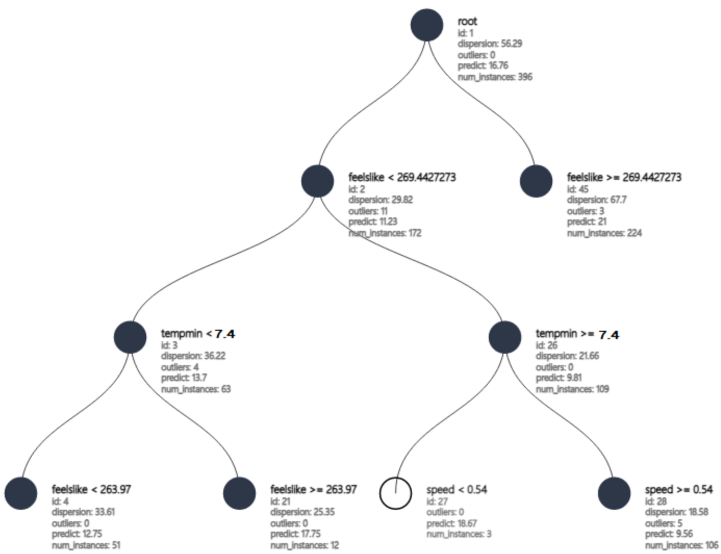


Figure 3. Decision Tree of multivariate model.

In summary, it should be noted that each decision rule on our trees is associated with a node that contains more than a dozen useful attributes to understand the distribution of data.

4. Conclusions and Future Work

We consider that this work is another contribution to the Explainable Artificial Intelligence (xAI) area that can be applied in Human Mobility. At a time when the topic of Artificial Intelligence (AI) is our daily choice, this is another successful project that guarantees that Deep Learning model does not necessarily need to be considered a ”black-box”, because this approach aims precisely at the construction of xAI systems, in a way that the human being can understand what makes the model behave in a certain way. Thus, making it possible for us to understand human mobility in New York city center. Therefore, we concluded that the user can make a decision based on the Machine Learning algorithm which is generally the best option for predictive modeling, arguing it based on the explanation of the Decision Tree (DT) which in turn is also the more explanatory approach.

In the present work, we did not apply the same concept to explain other algorithms. In order to complement the project, we can research the similarity of the predictions made between DT and other algorithms not addressed in the present project such as Convolutional Neural Network (CNN), hybrid model using LSTM and CNN model or statistics like ARIMAX and SARIMAX. With these new challenges we may find a model even more similar to DT.

Acknowledgments

This work is funded by “FCT—Fundação para a Ciência e Tecnologia” within the RD Units Project Scope: UIDB/00319/2020. This work is funded by CCDR-N Project: NORTE-01-0145-FEDER-000086. This work has also been supported by national funds through FCT – Fundação para a Ciência e Tecnologia through project UIDB/04728/2020.

References

[1] Carneiro D, Silva F, Guimarães M, Sousa D, Novais P. Explainable Intelligent Environments. In: Advances in Intelligent Systems and Computing. vol. 1239 AISC; 2021. p. 34-43.

[2] Angelov B. Explainable AI (Part I): Explanations and Opportunities;. Available from: https://towardsdatascience.com/explainable-ai-part-i-explanations-and-opportunities-5c2734c23d6e.

[3] Kabir M, Hasan K, Hasan M, arXiv preprint ArXiv KA, 2021 U. Explainable Artificial Intelligence for Smart City Application: A Secure and Trusted Platform. Springer Nature. 2021;1. Available from: https://arxiv.org/abs/2111.00601.

[4] Maiya A, Mendes T, Mecanico P. Explainable AI in Practice. Safety Critical AI. 2021:1. Available from: https://partnershiponai.org/workstream/explainable-ai-in-practice/.

[5] M Abdur Rahman AJS M Shamim Hossain, Alrajeh NA. Human-AI Teaming for the Next Generation Smart City Healthcare Systems; 2021. Available from: https://smartcities.ieee.org/newsletter/june-2021/human-ai-teaming-for-the-next-generation-smart-city-healthcare-systems.

[6] Ahangar MN, Ahmed QZ, Khan FA, Hafeez M. A survey of autonomous vehicles: Enabling communication technologies and challenges. Sensors (Switzerland). 2021;21(3):1-33.

[7] Bokhari SAA, Myeong S. Use of Artificial Intelligence in Smart Cities for Smart Decision-Making: A Social Innovation Perspective. Sustainability (Switzerland). 2022 jan;14(2):620. Available from: https://www.mdpi.com/2071-1050/14/2/620/htmhttps://www.mdpi.com/2071-1050/14/2/620.

[8] Ribeiro MT, Singh S, Guestrin C. ”why should i trust you?” explaining the predictions of any classifier. In: NAACL-HLT 2016 - 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Demonstrations Session. Association for Computational Linguistics (ACL); 2016. p. 97-101.

[9] Weather Underground. Weather API; 2018. Available from: https://www.wunderground.com/weather/api.

[10] Baeldung. Generate Spring Boot Project with Swagger; 2021. Available from: https://www.baeldung.com/spring-boot-rest-client-swagger-codegen.

[11] Javed Mehedi Shamrat FM, Ranjan R, Hasib KM, Yadav A, Siddique AH. Performance Evaluation Among ID3, C4.5, and CART Decision Tree Algorithm. In: Lecture Notes in Networks and Systems. vol. 317. Springer Science and Business Media Deutschland GmbH; 2022. p. 127-42.

[12] Scikit-learn. 1.10. Decision Trees — scikit-learn 1.0.2 documentation; 2021. Available from: https://scikit-learn.org/stable/modules/tree.html.