Efficient and interpretable crash prediction models : An application to a French highway network

Thomas Véran

N°d'ordre NNT : 2022ISAL0096

THESE de DOCTORAT DE L'INSA LYON, membre de l'Université de Lyon

Ecole Doctorale N° 512
INFOMATHS

Spécialité/ discipline de doctorat :

Informatique

Soutenue publiquement le 04/11/2022, par :
Thomas, Marcel, Simon, Nicolas Véran

Efficient and interpretable crash prediction models: An application to a French highway network

---

Devant le jury composé de :

Prof. Dr. Florence Sèdes (University of Sciences, Toulouse 3)
Prof. Dr. Julien Jacques (University of Lyon)
Prof. Dr. Pierre Gançarski (University of Strasbourg)
Prof. Dr. Gabriele Gianini (University of Milan)
Prof. Dr. Jean-Marc Petit (INSA Lyon)
Dr. Pierre-Edouard Portier (INSA Lyon)
Dr. François Fouquet (Data New Road)

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **https://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https://edeea.universite-lyon.fr** Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**Mme Sandrine CHARLES** Université Claude Bernard Lyon 1 UFR Biosciences Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69622 Villeurbanne CEDEX **sandrine.charles@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://ediss.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **hamamache.kheddouci@univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **https://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Christian MONTES** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

Worldwide, highway accidents have important social and financial impacts. To reduce their frequency and gravity, crash prediction models (CPM) are used to identify hazardous roadway segments and to provide actionable clues about the associated risk factors. CPM are either parametric statistical models, in particular generalized linear models (GLM), or machine learning models with a large number of parameters without associated uncertainty estimates (e.g., ensemble of decision trees, support-vector machine ...). Simple parametric models tend to be more interpretable but less effective than highly flexible non-parametric models that work like black-boxes. When pondering high stake decisions, such as in the context of highway safety, field experts expect predictive models to be both effective and glass-box interpretable. The models must assist them in conceiving and deploying preventive or remedial safety actions.

As such, we contribute to enhancing the predictive performance of parametric models while maintaining their interpretability. In the first place, a well-chosen hierarchical structure can handle correlations among groups of observations and significantly improve the quality of the models' predictions and of their interpretation. We propose to learn it by leveraging the output of a post-hoc explainability framework (*viz.*, SHAP) applied to a highly flexible black-box model (*viz.*, XGBoost). In our first contribution, this hierarchical structure informs a Bayesian multilevel GLM. Moreover, in an effort to further improve the predictive performance of the model without deteriorating its interpretability, we propose to extend its linear functional form to account for major first-order interactions between explanatory variables. These interactions are learnt from the data by analyzing the results of a trained self-organized polynomial network.

In our second contribution, we exploit the hierarchical structure even better by replacing the GLM with a simulated annealing based multi-objective symbolic regression algorithm to automate feature engineering and feature selection. Thus, by computing a cluster-specific ranking of expansions of regularized linear models ordered by increasing complexity, we facilitate a dynamic interpretative process which makes it possible to discover effective, efficient and interpretable predictive models.

Experiments have been conducted on a highway safety dataset and on more than ten public datasets covering classification and regression tasks. They show promising results with our two contributions outperforming traditional glass-box interpretable models while getting close to the best non-parametric models. Finally, we illustrate the benefits of our approach by introducing, on a realistic case study, an application we designed for highway safety experts.

Résumé

Dans le monde entier, les accidents de la route ont des impacts sociaux et financiers importants. Pour réduire leur fréquence et leur gravité, les modèles de prédiction d'accidents (CPM) sont utilisés pour identifier les segments de route dangereux et fournir des indices exploitables sur les facteurs de risque associés. Les CPM sont soit des modèles statistiques paramétriques, en particulier des modèles linéaires généralisés (GLM), soit des modèles d'apprentissage automatique avec un nombre important de paramètres sans estimation d'incertitude associée (e.g., ensemble d'arbres de décision, machine à vecteurs de support ...). Les modèles paramétriques simples ont tendance à être plus interprétables mais moins performants que les modèles non paramétriques très flexibles qui fonctionnent comme des boîtes noires. Lorsqu'ils réfléchissent à des décisions à fort enjeu, comme dans le contexte de la sécurité routière, les experts métier s'attendent à ce que les modèles prédictifs soient à la fois performants et interprétables. Les modèles doivent les aider à concevoir et à déployer des actions de sécurité préventives ou correctives.

Dans ces travaux, nous contribuons à améliorer les performances prédictives des modèles paramétriques tout en conservant leur interprétabilité. En premier lieu, une structure hiérarchique bien choisie peut gérer les corrélations entre groupes d' observations et améliorer significativement la qualité des prédictions des modèles et leur interprétation. Nous proposons de l'apprendre en exploitant le résultat d'une méthode d'interprétabilité post-hoc (*viz.*, SHAP) appliquée à un modèle boîte noire flexible (*viz.*, XGBoost). Dans notre première contribution, cette structure hiérarchique informe un GLM bayésien multiniveaux. De plus, dans le but d'améliorer encore les performances prédictives du modèle sans détériorer son interprétabilité, nous proposons d'étendre sa forme fonctionnelle linéaire pour tenir compte des interactions majeures de premier ordre entre variables explicatives. Ces interactions sont apprises à partir des données en analysant les résultats d'un réseau polynomial auto-organisé.

Dans notre deuxième contribution, nous exploitons encore mieux la structure hiérarchique en remplaçant le GLM par un algorithme de régression symbolique multi-objectif basé sur le recuit simulé pour automatiser la sélection des variables explicatives et l'extraction de caractéristiques (*viz.*, interactions, transformations de

variables explicatives). Ainsi, en calculant un classement spécifique à chaque cluster des expansions de modèles linéaires régularisés ordonnés par complexité croissante, nous facilitons un processus d'interprétation dynamique qui permet de découvrir des modèles prédictifs efficaces, efficients et interprétables.

Des expériences ont été menées sur un jeu de données de sécurité routière et sur plus de dix jeux de données publics couvrant des problèmes de classification et de régression variés. Les résultats obtenus sont prometteurs étant donné que nos deux contributions surpassent les modèles interprétables traditionnels et se rapprochent des meilleurs modèles non paramétriques boîtes noires. Enfin, nous illustrons les bénéfices de notre approche en présentant, sur une étude réelle de cas, une application que nous avons conçue pour les experts de la sécurité routière.

Acknowledgments

First, I would like to express my sincere gratitude to my supervisors: Prof. Dr. Jean-Marc Petit for giving me the opportunity to do this thesis. Dr. Pierre-Edouard Portier for your immense knowledge, your invaluable guidance, and your human qualities that made the collaboration extremely easy and natural. I couldn't imagine having a better scientific supervisor. You have inspired me and motivated me for the past three years. I hope that we will continue to see each other, to discuss exciting topics, and even to work together if our paths cross again. Dr. François Fouquet for being an excellent supervisor whose continued support and advice have been key elements throughout my PhD study.

Besides my supervisors, I would also like to thank the members of IRIXYS and the DRIM team. Through workshops and team meetings, you have given me valuable advice.

My sincere thanks also go to Pascal Philip, who trusted me throughout the years, right from the internship at APRR. Speaking of APRR, I would like to thank Sylvie Dumas, Claire Dufosse and Isabelle De France for their knowledge and expertise. You allowed me to identify and understand the challenges of road safety.

Also, I would like to thank Damien Corbi for trusting me from the beginning. You gave me all the tools to conduct this thesis in the best possible way. I also take this opportunity to thank the employees of DNR, forming a team full of talents. These few years with you have passed far too quickly. I wish you all an excellent continuation.

To my friends from *la chambre 15* whom I consider to be my second family, thank you for our moments spent together laughing, having fun. Believe me, they did me a lot of good. A special mention for Thomas, my closest friend: you are an exceptional person and I feel privileged to have you as a friend.

I would like to sincerely thank my parents for their endless support, their love and encouragement during these years. Without you, I surely wouldn't be here today. I hope to give you back as much as you gave me. To my beloved brother Julien, I am extremely happy to share all these moments with you, and those that will come. I wish us success in 7b (and higher grades) boulders. To Cédric, wherever you are, know that you are by my side every day: your unwavering will, your ability to live

life to its fullest, will remain engraved in me. I feel so lucky to have known you.

The best for the end, I would like to thank my love, Manon. You can't imagine how much I admire you for your passion, your perseverance, for being you. Since you came into my life, everything seems happier and simpler. With you by my side, I feel able to reach my most unattainable goals and live my wildest dreams. *Je t'aime, du plus fort que je peux.*

Scientific environment

The research reported in this dissertation has been done within CIFRE industrial agreements between Data New Road, subsidiary of APRR group, and the laboratory of computer science for image and information systems (LIRIS) at INSA Lyon, France. The doctoral student was part of the research groups for distributed systems, information retrieval and mobility (DRIM) and data base (BD).

This work has received financial support from the French Association for Research and Technology (ANRT).

Contents

1 Introduction & problem statement **1**
- 1.1 Context and objectives . . . 1
- 1.2 Crash prediction models . . . 3
- 1.3 Model interpretability: concepts, taxonomies . . . 3
- 1.4 Machine learning formalization for crash prediction . . . 5
  - 1.4.1 Data description . . . 5
  - 1.4.2 Supervised learning . . . 5
  - 1.4.3 Application to the highway dataset . . . 6
  - 1.4.4 Challenges and research questions . . . 7
- 1.5 Publications and outline . . . 10
  - 1.5.1 Publications . . . 10
  - 1.5.2 Outline . . . 11

2 Related work **12**
- 2.1 Model interpretability . . . 12
  - 2.1.1 Properties . . . 12
  - 2.1.2 Intrinsic interpretability . . . 13
  - 2.1.3 Post hoc interpretability . . . 15
  - 2.1.4 Model interpretability in the ML community . . . 21
- 2.2 Crash prediction models and their interpretability . . . 21
  - 2.2.1 Parametric statistical models . . . 21
  - 2.2.2 Non-parametric Machine Learning models . . . 25

3 Contributions **29**
- 3.1 Overview . . . 29
  - 3.1.1 Introduction . . . 29
  - 3.1.2 Bayesian hierarchical generalized linear model . . . 30
  - 3.1.3 Interpretable hierarchical symbolic regression . . . 30
- 3.2 Supervised learning of a hierarchical structure . . . 31
  - 3.2.1 Introduction . . . 31

3.2.2 Training of a non-parametric ML model . . . 32
3.2.3 Local post hoc explanations . . . 34
3.2.4 Discovery of a hierarchical structure . . . 35
3.2.5 Representation of the hierarchical structure in highway safety 37
3.3 Bayesian hierarchical generalized linear model . . . 40
3.3.1 Model description . . . 40
3.3.2 Bayesian inference . . . 41
3.3.3 Supervised learning of interactions with a polynomial network 43
3.3.4 Model evaluation and validation . . . 46
3.3.5 Conclusion . . . 46
3.4 Interpretable hierarchical symbolic regression . . . 48
3.4.1 Related work . . . 48
3.4.2 Symbolic regression with Pareto simulated annealing . . . 50
3.4.3 Partial pooling approach for symbolic regression . . . 57
3.4.4 Uncertainty estimation . . . 59
3.4.5 Implementation details . . . 61

4 Experiments **63**
4.1 Description of the datasets . . . 64
4.1.1 French highway dataset . . . 64
4.1.2 Other datasets . . . 66
4.2 Data preparation . . . 67
4.3 Performance metrics . . . 68
4.3.1 Regression . . . 68
4.3.2 Classification . . . 68
4.4 Models used for comparison . . . 70
4.4.1 Parametric interpretable models . . . 70
4.4.2 Non-parametric black-box models . . . 71
4.4.3 Models from our proposal . . . 71
4.4.4 Hyper-parameters tuning . . . 72
4.5 Results . . . 74
4.5.1 Hierarchical structure module . . . 74
4.5.2 Regression datasets . . . 74
4.5.3 Classification datasets . . . 79
4.5.4 Discussion . . . 79
4.6 Dynamic interpretative process . . . 82
4.6.1 Introduction . . . 82
4.6.2 Use case . . . 82

5 Conclusion and perspectives **91**
5.1 Conclusion 91
5.2 Perspectives 92

List of Figures

1.1 APRR's highway network [9] 2
1.2 Bias and variance contributing to total error [37] 8

2.1 Example of a shallow decision tree 14
2.2 Partial dependence on the French highway dataset 16
2.3 Toy example of a linear SVM 26
2.4 Multilayer perceptron and the computation of a neuron output 27
2.5 Sensitivity analysis performed on a Bayesian neural network 28

3.1 Description of a Bayesian hierarchical generalized linear model 30
3.2 Proposed framework for an interpretable hierarchical symbolic regression 31
3.3 Example of a forceplot for a specific observation 35
3.4 Observations grouped by similarity of their forceplots 36
3.5 Clusters automatically discovered by the *hierarchical structure* module 37
3.6 Variables' distributions for 4 identified clusters 38
3.7 Clusters when the number of clusters is reduced to 2 38
3.8 Variables' distributions for 2 identified clusters 39
3.9 Clusters when the number of clusters is increased to 6 39
3.10 Variables' distributions for 6 identified clusters 39
3.11 Two examples of posterior distributions 42
3.12 Structure of the polynomial network 44
3.13 Triptych plots of predicted crash counts vs. annual average daily traffic 45
3.14 Posterior Predictive Checks on two clusters 47
3.15 A functional form associated with a set of expression trees. 51
3.16 A sequence of transformations applied to an expression tree 52
3.17 Effect of the annealing temperature on the acceptance probability. . . 56
3.18 Example of an approximated Pareto front and its attainment surface 57
3.19 Extraction of a cluster-specific functional form 59

4.1 Distribution of the observed crash counts 65
4.2 Examples of ROC curve and precision-recall curve 70

4.3 Global explanation plot provided by the InterpretML framework . . . 81
4.4 Screenshot of the web application 83
4.5 Clusters identified for 2018 85
4.6 Pareto front for cluster 0 specific models 85
4.7 Crash count predictions for 2018 86
4.8 Effects plots and posteriors of the most influencing variables for model 1 87
4.9 Effects plots and posteriors of the most influencing variables for model 2 87
4.10 Histograms of partial derivatives 89
4.11 Effects plots 90

List of Tables

1.1 Illustrative description of the french highway dataset 7

3.1 Rules for interval arithmetic 53
3.2 Algebraic rules used to compute the complexity of each term 55

4.1 Description of the *French Highway* dataset 66
4.2 Datasets 67
4.3 Confusion matrix 68
4.4 Parameters and priors for the Bayesian hierarchical models 72
4.5 Hyper-parameters tuning 73
4.6 Performance of the prediction to associate a new observation to its cluster 75
4.7 Results on the regression datasets 77
4.8 Results obtained by cluster-specific interpretable models 78
4.9 Results on the classification datasets 80
4.10 Descriptive statistics of variables for each cluster 84
4.11 Description of explanatory variables for the overall cluster-specific data and for samples where partial derivatives w.r.t. the traffic variable are the lowest 89

Chapter 1

Introduction & problem statement

1.1 Context and objectives

Road safety is a socio-economic concern: according to the World Health Organization [132], approximately $1,35$ million people are killed each year on roadways around the world. Related expenses average 3% of the gross domestic product of a country. As stated by the French Road Safety Observatory [38], these costs grow exponentially with the severity of the accidents. In 2019, a property damage only accident incurred expenses up to 5 258 euros while the average cost of an accident with at least one fatality was 3 429 000 euros[1].

Since the 1970s, road safety has become a major challenge for successive French governments. Many new safety policies have been engaged in, such as reductions of speed limits, the multiplication of fixed radars, the increase in the amount of speeding fines. At the same time, vehicles became more secure. Road safety was therefore continuously improved. However, since 2010, the positive trend was attenuated and the reduction in the fatality rate started to stagnate. For this reason, the French government, supported by the European Union, motivated new safety policies whose main issuers would be local authorities and road managers. Among them, *APRR*[2] group, subsidiary of *Eiffage*, finances, maintains and manages the infrastructure of a 2 323 km-long highway network (see Fig. 1.1) in return for toll collection. The objective of the company is to offer high-performance transport infrastructure and support road users at every stage of their travel, guaranteeing them the best conditions for traffic flow and safety.

To secure their network, field experts of *APRR* leveraged an extension of the User's

[1] These amounts include the various costs incurred for territorial and local authorities (hospitalization, insurance, etc.) and for road managers (marking, road works, etc.)

[2] Autoroutes Paris-Rhin-Rhône: `https://aprr.com/en`

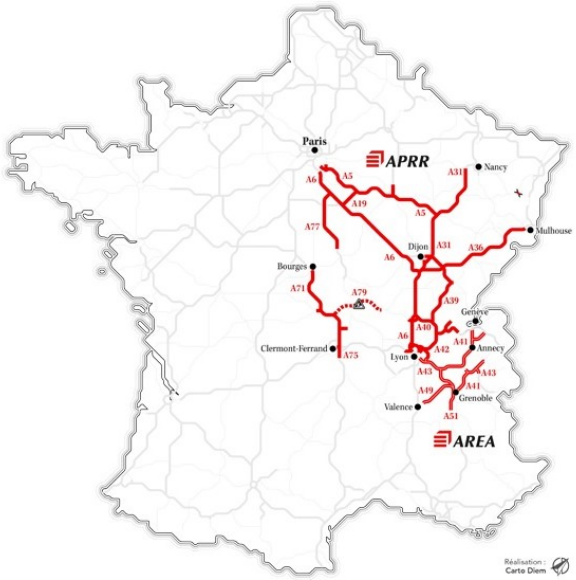


Figure 1.1: APRR's highway network [9]

Safety on Existing Roads (SURE) approach, a comprehensive and global framework for network safety management initiated by the French government. First, they implemented a multi-criteria decision analysis method to compute a ranking of segments based on the values of different crash-related criteria (e.g., annual average daily traffic, fatality or injury rates, ...). Then, after studying crash reports and conducting on-site visits, field experts elaborated pluriannual action plans (operation, maintenance, ...) on segments where the safety improvement were potentially the greatest. In the end, this method contributed to a significant reduction in fatality and severity rates.

Nevertheless, SURE does not work on reducing the crash likelihood and does not explain the mechanisms affecting crash-prone areas. Therefore, decision makers have opted for the development of new intelligent systems inspired by the latest advances in Artificial Intelligence (AI). A collaboration has been initiated between *LIRIS*[3] laboratory and *Data New Road*[4], a subsidiary of *APRR* dedicated to the harnessing of the company's historical data. The main objective of this collaboration was to build an AI system to estimate accurately future hazardous segments and to identify risk factors. A system that meets these requirements will be a fundamental element in the implementation of new proactive safety policies, and will therefore have a beneficial and directly measurable impact.

[3]Laboratoire d'InfoRmatique en Image et Systèmes d'information: `https://liris.cnrs.fr/`
[4]`https://www.data-newroad.com/`

1.2 Crash prediction models

In the field of road safety, research has focused on obtaining a better understanding of the mechanisms that affect the probability of an accident. Given the absence of precise data related to human behavior (lack of attention, fatigue, alcohol, ...), crash prediction models (CPM) do not provide cause-and-effect relationships that would explain the occurrence of an accident. Instead, they emphasize the use of factors that could influence the likelihood of an accident over a given period (e.g., year, month) and in a given geographical area. CPM are trained on historical data to discover hidden patterns between the crash-related variable and several explanatory variables (e.g., traffic, speed limit, altitude, ...). They are mainly used to identify risk factors in order to steer the evolution of safety policies.

In their survey [75], Lord and Mannering provide a broad perspective on the variety of data-related issues raised by crash count prediction: over-dispersion of count data, temporal and spatial correlations due to multiple measurements of a same location at different times, fixed parameters that cannot adapt from one roadway to the next, low sample-mean due to the sparsity of crashes, non-linear relationships between crash-frequencies and explanatory variables, etc. Most of theses issues are made more prominent with the use of parametric models, in particular generalized linear models (GLM). Indeed, GLM must undergo many transformations to adapt to the crash count prediction context (e.g., the choice of a non-normal likelihood distribution, the integration of random effects and hierarchical models, etc). Whereas non-parametric approaches (e.g., neural networks, tree-based algorithms, SVM...) will deal with most of these issues without the need for specific adaptations and will usually offer better predictive performances than parametric models. On the other hand, this improvement in predictive power comes at a cost. These models, which involve a large number of parameters, are often considered as black boxes that do not allow direct understanding of the mechanisms that led to a prediction. Thus, when designing CPM, both dimensions (*viz.*, performance and interpretability) must be considered. However, while performance is easily measured with widely adopted metrics, quantifying interpretability remains more debated.

1.3 Model interpretability: concepts, taxonomies

In our study, we consider a predictive model to be glass-box interpretable when it quantifies explicitly (i.e., not by simulation) the marginal effects of the explanatory variables and also possibly of a few simple transformations of these variables (e.g., multiplicative interactions, log transforms, etc.). Thus, we stress the importance of favoring simple functional forms namely, expansions of linear models.

On the contrary, non-parametric models allow for very flexible, but often complex, functional forms. For these models, some sense of the marginal effects can be gained by studying *ceteris paribus* profiles. They represent the influence of an explanatory variable by assuming that all the other variables remain constant. In the case of flexible functional forms, the choice of the fixed values for all the variables but the one for which the marginal effects are to be observed has a great influence on the interpretation of the effect. Often, different choices (if not all) should be considered to provide a more faithful view of the marginal effects. Of course, the number of configurations of fixed values grows quickly with the number of explanatory variables. Some heuristics have been used to manage this complexity. For example, Beck *et al.* [13] chose to "hold constant the other variables at two values: high and low probability" of the target.

To truly measure the marginal effects, one should, as stated in [47], "estimate partial derivative with respect to each input variable and at each observation through repeated simulations". This strategy is computationally prohibitive when implemented by simulating data from a complex model. Among the oldest computationally convenient variants, we find the permutation based methods applied to variable importance, partial dependence plots or individual conditional expectation plots. However, Hooker and Mentch [52] have found that "when features in the training set exhibit statistical dependence, permute-and-predict methods can be highly misleading when applied to the original model". More refined approximation methods have been proposed, such as LIME or SHAP. They are also based on observing the effects of a perturbation of a given instance on the output of the model. These approaches are often referred to as model-agnostic post-hoc explanation methods. However, recent works, in particular [114] tend to show that it can be difficult to assess their reliability and robustness. Also, Garreau and Luxburg [41], in a theoretical study of LIME, verified by simulations, showed that, while this post-hoc approach "discovers interesting features, it might forget some important features and the surrogate model is not faithful".

Generalized Additive Models (GAM) [49] can be located halfway between complex flexible functional forms and simple glass-box models. They give some access to the marginal effects by enabling plots of the expectation of the target given the values taken by each explanatory variable. Lou *et al.* [76] designed a GAM variant based on boosted trees, the Explainable Boosting Machine (EBM). It is often almost as accurate as flexible black-box models. However, Chang *et al.* [20] observed that different GAM algorithms, while offering comparable results in terms of accuracy, can lead to very different interpretations of the predictions.

Finally, we consider that simple linear models with the inclusion of some transforms of the original explanatory variables and the use of a regularization term (e.g., ridge or LASSO) to control the bias/variance trade-off remain promising among the intrinsically glass-box approaches. However, they come with key challenges. If one

doesn't want the user to be solely responsible for deciding which functions of the original variables are to be considered, a strategy must be adopted to explore a combinatorial space of potential functional forms. Also, for the model to stay interpretable in presence of a large number of explanatory variables, the number of relevant coefficients shouldn't be too large. However, variable selection is not without drawbacks. In particular, it suffers from the risk of hiding the main effects behind more complex correlated terms.

1.4 Machine learning formalization for crash prediction

1.4.1 Data description

Throughout this work, we will refer to $\boldsymbol{x}$ as the vector of explanatory variables describing a roadway segment during a given period. Each explanatory variable can be either continuous, discrete or categorical. Continuous variables can assume an infinite number of real values within a given interval. As opposed to a continuous variable, a discrete variable can assume only a finite number of numerical values within a given interval, such as the number of interchanges on a roadway segment in our context. The categorical variable refers to a variable that takes values in a discrete unordered set of categories $\{c_1, ...c_n\}$. Sometimes, categorical variables have only two categories and are therefore called binary variables. For instance, in our crash prediction problem, the variable *presence of tunnel* indicates whether a tunnel is present or not on a roadway segment.

The dependent variable, also known as the target variable, will be denoted y. In the context of crash count prediction, y is a discrete positive integer that quantifies the number of crashes observed on a roadway segment for a given time interval. Note that in the literature, the problem can be tackled with a binary categorical variable. In this case, the focus is not to predict a crash count, but to know whether or not a roadway segment will be accidental given the values of the explanatory variables. These models require the definition of a threshold that is used to differentiate crash-prone configurations from normal ones.

1.4.2 Supervised learning

In a supervised context, the predictive model must discover how to associate an observation $\boldsymbol{x}$ to a label y, based on a set of known examples $\{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$. To model these dependencies, the process goes through two phases:

- *training phase*: the model is trained to find patterns in the training data that map the vectors of explanatory variables $\boldsymbol{x}$ to the dependent variables y. To learn this mapping, the model goes through a stage called model training, where its parameters (e.g., weights, bias) are updated so the function it approximates captures the most relevant dependencies among the variables. Sometimes, models involve hyper-parameters i.e, parameters that are determined before training and remain fixed afterwards. Thus, an additional step called validation step can be performed in order to verify the correctness of their values and adjust them if necessary.

- *testing phase*: the trained model is evaluated on a set of unknown samples with appropriate performance metrics (e.g., mean squared error for regression tasks, $f1$-score for classification tasks). By doing so, one obtains a less biased estimate of the predictive performance of the model which will allow, firstly, to validate the generalization capacity of the model and secondly, to compare the predictive performance of the selected model with those of others.

However, doing a single *train-test* split will give, most of the time, an overoptimistic estimate of the predictive performance of the model. One way to mitigate this is to use k-fold cross-validation. In k-fold cross-validation, the dataset is split into k subsets of data (known as folds). The model is trained on $k-1$ subsets, and then evaluated on the remaining subset that was not used for training. This process is repeated k times, with a different test subset each time. The average performance obtained on the k test subsets gives an estimate of the model's predictive performance. In practice, k-fold cross-validation can serve multiple purposes such as defining the best set of hyper-parameters for a machine learning model or comparing multiple predictive models and selecting the one that has the best generalization ability.

1.4.3 Application to the highway dataset

In our work for highway safety analysis, the objective is to predict efficiently the crash count observed on highway segments, given the value of spatial explanatory variables (e.g., number of lanes, speed limit, ...) and temporal explanatory variables (annual average daily traffic, percentage of heavy vehicles). To learn these relationships, we leverage supervised learning: first, we start by generating a dataset with samples representing the crash count and the values of explanatory variables observed on a segment for a particular year (see table 1.1). The process of generating the dataset will be described in more detail in section 4.1. Then, we train a predictive model on a subset of randomly selected samples from the overall dataset (say for example 80%) to learn a function that maps the explanatory variables to the crash count.

Segment	Year	Crash count	AADT[a]	%HV[b]	#Interchanges	...	Presence of tollgates
A6-2-354-364[c]	2008	20	26456	19.59	0	...	1
A6-2-354-364	2009	20	26343	16.9	0	...	1
A6-2-354-364	2010	27	26656	17.31	0	...	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
A40-1-102-112	2017	10	12701	14	0	...	0
A40-1-102-112	2018	7	13015	14	0	...	0

[a] annual average daily traffic; [b] percentage of heavy vehicles
[c] highway name - direction - reference point begin - reference point end

Table 1.1: Illustrative description of the french highway dataset

To efficiently help field experts, this function will have to meet different challenges presented in the next section. Finally, we evaluate the performance of the resulting function on the remaining samples.

1.4.4 Challenges and research questions

If we assume the observed data are generated from an unknown data-generating process f such that $y = f(\boldsymbol{x}) + \varepsilon$, with ε being a zero-centered Gaussian noise of variance σ^2, our objective is to find a function $\hat{f}$ that is as close as possible to the true but unknown process which generated the data. Given the available training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, we estimate $\hat{f}$ so it minimizes an objective function such as the mean squared error $\sum_i (y_i - f(\boldsymbol{x}_i))^2$.

Bias-variance trade-off To measure the ability of $\hat{f}$ to generalize to points outside of the training data, we compute the expected prediction error (EPE) of the prediction $y^* = \hat{f}(\boldsymbol{x}^*) + \epsilon$, with $\boldsymbol{x}^*$ being an unseen sample. As shown by Hastie *et al.* [48], EPE can be decomposed as:

$$\begin{aligned} EPE &= \mathbb{E}[(y^* - \hat{f}(\boldsymbol{x}^*))^2] \\ &= (f(\boldsymbol{x}^*) - \mathbb{E}[\hat{f}(\boldsymbol{x}^*)])^2 + \mathbb{E}[\varepsilon^2] + \mathbb{E}[\hat{f}(\boldsymbol{x}^*)^2] - \mathbb{E}[\hat{f}(\boldsymbol{x}^*)]^2 \\ &= bias(\hat{f})^2 + \sigma^2 + var(\hat{f}) \end{aligned} \tag{1.1}$$

We observe that EPE embeds three different sources of error: σ^2 is the irreducible error that cannot fundamentally be reduced by any model, the *bias* measures the average error of $\hat{f}(\boldsymbol{x}^*)$, and the variance *var* gives a measure of the variation of the prediction $\hat{f}(\boldsymbol{x}^*)$ from one training dataset to another. An ideal model minimizes both the bias and the variance. However, in a world with imperfect models and finite data, it is nearly impossible to do both simultaneously. On the one hand, complex models

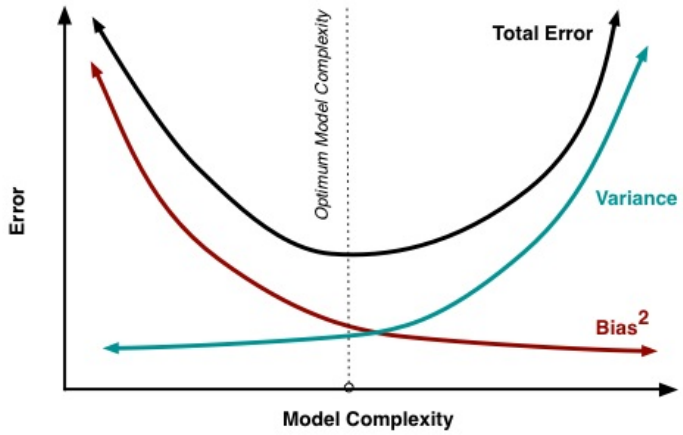


Figure 1.2: Bias and variance contributing to total error [37]

(e.g., deep neural networks) approximate precisely training data but are exposed to over-fitting. On the other hand, simpler models with high bias, such as linear models, fail to capture important regularities in the data and are therefore prone to under-fitting. Thus, efforts should be made to reach optimum model complexity for which the expected prediction error is generally close to its lowest level (see Fig. 1.2). One way of doing this is to use regularization to find a good trade-off between bias and variance. Likewise, dimensionality reduction and feature selection also decrease the variance of complex models.

Finally, when designing predictive models in a high-stakes context, it is necessary to not focus only on mitigating the bias-variance trade-off, but also to consider model interpretability because the understanding of the mechanisms that lead to a prediction is as valuable as the prediction itself.

Performance and interpretability Broadly speaking, non-parametric complex models, which cover big hypothesis spaces, often obtain very good performance on many predictive tasks. However, as they involve a large amount of parameters, they're often considered as black-boxes [10]. To gain insights into their underlying behaviors, post hoc explanation tools must be used. Nonetheless, these tools have several drawbacks w.r.t. the trust and veracity of the explanations they deliver [104, 114]. Indeed, the representation of the model can be inaccurate in parts of the feature space and may even be misleading in the under-represented parts [91]. Moreover, even though these tools can reproduce accurately the predictions of the original model, they may use completely different features. A well-known example is the proprietary prediction model COMPAS, a recidivism-risk scoring model used widely throughout the U.S. criminal justice system for parole and bail decisions, which has been criticized

as being dependent on race based on the interpretations of an explanation model [60] whereas it has been demonstrated afterwards that COMPAS does not seem to "depend strongly on either criminal history or proxies for race" [106].

Moreover, when black-box models do not behave as expected (e.g., excellent accuracy during training, but very bad performance in production or with testing data), trying to gain insights into the model is required for troubleshooting (debugging). Post hoc explanations are commonly used. However, combining a black-box model and an explanation model (or tool) makes the process of troubleshooting very sensitive. As the explanation is not always correct, it can be difficult to tell whether or not the black-box model is wrong [105].

Furthermore, as stated by Rudin [105], "black-box models often predict the right answer for the wrong reason, leading to excellent performance in training but poor performance in practice". In psychology, this is known as the clever Hans phenomenon [98], that comes from a horse (of the same name) claimed to perform arithmetic and other intellectual tasks, but actually was watching the reactions of his trainer. In the AI context, this phenomenon describes black-box models that can learn different relationships between variables than the ones of the true data generating process and, at the same time, provide very accurate predictions. Thus, efforts should be made to avoid this phenomenon in domains where high-stakes decisions can be made from the analysis of predictive models such as healthcare, road safety, or criminal justice.

On the other hand, researchers working in the field of interpretable AI observe that linear models can perform very well on tabular data, especially when the full data science process is considered (e.g., variable selection, data preparation, model selection, ...) [104]. This could arise from the Rashomon Effect [16], which characterize problems where many accurate-but-different models exist to describe the same data [110]. In [110], the authors show empirically that when the Rashomon set (i.e. the set of almost-equally-accurate models) is large, "most machine learning methods tend to perform similarly, and also in these cases, interpretable or sparse (yet accurate) models exist".

Lastly, interpretable models provide a simple and intelligible picture of the relationship between the explanatory variables and the dependent variable. This not only ease the process of elaborating new actions, but also allow end users to be confident in what they plan.

1.5 Publications and outline

1.5.1 Publications

In this thesis, we try to meet these challenges by extracting, from data and without prior expert knowledge, the information necessary to build efficient and highly interpretable models. In our first proposal, we consider that data can often be partitioned so that refined predictive models can apply to different parts more efficiently and more meaningfully than a global model. We discover such a structure by clustering the instances based on the features' scores returned by the SHAP [79] post-hoc analysis of a flexible black-box model. This clustering then informs a Bayesian multilevel – hierarchical – generalized linear model. Moreover, we propose to integrate nonlinearities by adding to its underlying functional form, the most important interactions between explanatory variables learnt by a self-organized polynomial network. This first contribution, described in section 3.2 and section 3.3, was published in:

Crash prediction for a French highway network with an XAI-informed Bayesian hierarchical model, *2020 IEEE International Conference on Big Data (Big Data)* [126]

In our second proposal, we propose to exploit the hierarchical structure even better by computing a cluster-specific ranking of expansions of regularized linear models ordered by increasing complexity. We design a symbolic regression approach for the automatic discovery of a mapping from the original explanatory variables to a basis expansion which adds both sparsity and flexibility by including transforms of the variables. This exploration of the space of potential expansions of linear models is guided by simulated annealing. More precisely, given a predictive performance metric and a complexity metric, the meta-heuristic search conducts a multi-objective optimization to return a list of Pareto optimal models. Through a dynamic interpretative process, end users can navigate around these models and select the best model according to their needs, from the simplest one highlighting main effects to a more specific one depicting particular configurations related to few instances in the dataset. This contribution, described in section 3.4, was published in:

Interpretable hierarchical symbolic regression for safety-critical systems with an application to highway crash prediction. *Engineering Applications of Artificial Intelligence*, 117:105534, 2023. [127]

1.5.2 Outline

In chapter 1, we started by introducing the context and stakes of road safety analysis and then described the numerous challenges crash prediction models have to face. In the next chapter, we will first provide more details on the different concepts of model interpretability through a review of the state of the art before introducing how crash predictions and explanations have been tackled in the community. In chapter 3, we present our contributions. More precisely, after giving an overview of our methodology in section 3.1, we will describe in section 3.2 the first module of our methodology, which consists in finding a hierarchical structure in the dataset. In section 3.3, we will explain how we integrate this hierarchical structure to enhance both the interpretability and efficiency of GLM. We will also explain how non-linearities are discovered and afterwards embedded in the GLM to enhance, yet again, its predictive performance. In section 3.4, we will describe how the aforementioned methodology can be improved by replacing the GLM by a symbolic regression approach that allows us to capture more relevant non-linearities while providing sparse and highly interpretable models. Then, in chapter 4, we will present the experiments carried out on the highway network dataset and on thirteen public datasets covering different tasks (*viz.*, regression and classification) which underline very promising results as our framework outperforms fully interpretable models (e.g., linear models, shallow decision trees) while getting close to non-parametric models. Finally, in section 4.6, we unveil its potential for guiding a dynamic interpretative process with a realistic case study on the highway network dataset which highlights the benefit of using multi-objective optimization to help field experts develop new safety policies.

Chapter 2

Related work

2.1 Model interpretability

2.1.1 Properties

AI systems are used to assist field experts in making high stake decisions which may indirectly affect humans' lives. Thus, understanding how does the system behave is of paramount importance. In many areas, knowing the predictions made by a model is not enough. There is an emerging need that these models, besides being efficient, compel to numerous properties regarding their explainability [10]:

- *trustworthiness*: the models will act as intended when facing a given problem
- *informativeness*: to support decision making, they provide a great amount of information on the problem being tackled
- *confidence*: they communicate on the confidence of their working regime
- *accessibility*: they allow end users to get more involved in their development cycle
- *interactivity*: they allow straightforward interactions with end users.

The design of interpretable AI systems has been tackled with two distinctive approaches rallied around the field of explainable Artificial intelligence (XAI). On the one hand, there are models that are inherently interpretable as their complexity is restricted by their design (e.g., GLM, decision trees). On the other hand, complex models (e.g., deep Neural Network, ensemble of decision trees) act as black-boxes and do not provide, at least initially, any information on the relationship that connects the explanatory variables to the dependent variable. Knowledge is often obtained by using post hoc explainability techniques.

2.1.2 Intrinsic interpretability

Some models are inherently interpretable and indeed meet the above properties. They are often perceived as glass-box models as the relation between the explanatory variables and the dependent variable is explicit, thus simplifying the understanding of the marginal effects. Among them, we can find shallow decision trees, generalized linear models (GLM), generalized additive models (GAM).

Decision trees

Decision trees can be used to answer regression problems and classification problems. They are commonly found under the term Classification and Regression Trees (CART) introduced by Breiman [17]. These models aim at predicting the value of a dependent variable by learning simple decision rules from the features. They divide the input space by recursively splitting the data into subsets according to tests on features (see Fig. 2.1). The procedure to select the best split varies according to the problem at hand. For regression tasks, CART takes a feature and determines which cut-off minimizes the variance of the dependent variable. For classification tasks, CART seeks to minimize the Gini impurity of the class distribution defined as:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

where p_i is the probability of an observation to be classified to one of the C classes. The algorithm continues the search-and-split in the new nodes until a stopping criterion is reached (e.g., minimum number of observations in a node before splitting, minimum number of observations in a terminal node). Finally, to predict the outcome of a new observation, CART averages all training instances in the terminal node reached by the new observation after browsing the tree.

Interpreting the prediction is straightforward as it combines, from the root node to the terminal node, simple decision rules. However, for deep trees, the high number of decision rules used for a prediction makes it hard to understand. Thus, in our study, we consider a CART model to be interpretable if its depth is, at most, equal to 5.

Generalized linear model

In a GLM [80], the dependent variable Y is assumed to be generated from a particular distribution in an exponential family (e.g., normal, binomial, Poisson distributions). The mean μ of the distribution is connected to a linear predictor $\mathbf{X}\beta$ such that:

$$g(E[Y|\mathbf{X}]) = g(\mu) = \mathbf{X}\beta \tag{2.1}$$

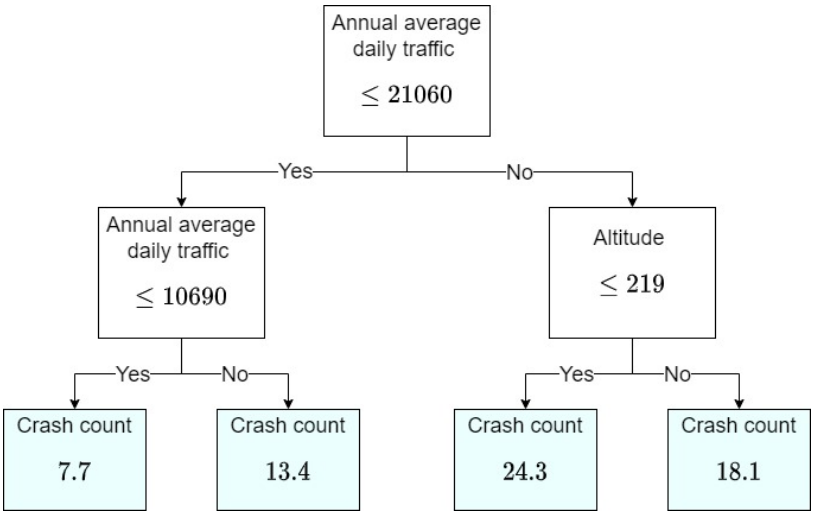


Figure 2.1: Example of a shallow decision tree trained on the French highway dataset

where g is a link function introduced to map the non-linear transformed mean $g(\mu)$ to the definition space of the linear predictor, the real line. The choice of a relevant link function depends on the problem at hand. For instance, for a dependent variable generated from a binomial distribution, the preferable link function is the *logit*. For that model, Eq. 2.1 becomes:

$$logit(\mu) = \ln(\frac{\mu}{1-\mu}) = \mathbf{X}\beta$$

which is also referred to as logistic regression. For a normally distributed dependent variable, the link function is simply the identity function *id*:

$$id(\mu) = \mu = \mathbf{X}\beta$$

The coefficient β can be estimated by maximum likelihood estimation or Bayesian inference (see section 3.3.2 for a detail explanation of these techniques).

Generalized additive model

To model nonlinear but still interpretable relationships between the dependent variable and the explanatory variables, one can apply transformations (e.g., logarithm, categorization) to the original variables. Another option is to use GAM [49], an extension of GLM where the linear predictor embeds unknown smooth functions of m explanatory variables:

$$g(E[Y|X_1, ..., X_m]) = \beta_0 + \sum_{i=1}^{m} f_i(X_i) \tag{2.2}$$

To learn the nonlinear flexible functions f_i, different approaches have been proposed:

Spline basis The original one is based on spline basis and makes use of the backfitting algorithm [49], which works by iterative smoothing of partial residuals. However, it is difficult to estimate the degree of smoothness of splines f_i. This parameter can be either set by the users or estimated with cross-validation during model training. However, this original approach is computationally expensive ($\mathcal{O}(n^3)$ with n the number of observations). Thus, several methods try to reduce the size of the basis used for smoothing[33, 64], or find sparse representations of the smooth functions using Markov random fields [107].

Modern machine learning techniques In [122], the authors proposed to train GAM with a likelihood-based boosting procedure. Lately, Lou *et al.* [76], in a framework called Explainable Boosting Machine (EBM), proposed a tree-based extension of GAM that introduces a small number of two-dimensional interactions:

$$g(E[Y|X_1, ..., X_m]) = \beta_0 + \sum f_i(X_i) + \sum f_{ij}(x_i, x_j) \tag{2.3}$$

To learn the nonlinear function f, they combine bagging [15], gradient boosting [39] (further explained in section 3.2.2) and automatic detection of pairwise interactions. During training, features are trained one at a time in round-robin fashion to mitigate the effects of multicollinearity. When applied to tabular data, EBM outperforms standard GAM based on regression splines and is often almost as accurate as flexible black-box models. Moreover, EBM is also considered to be interpretable because each feature contributes to the prediction in an additive way (see Eq. 2.3) thus simplifying the understanding of its influence on the dependent variable.

2.1.3 Post hoc interpretability

Highly flexible models usually reach better predictive performance on some datasets mainly because they manage to capture nonlinearities in the data. However, this comes along with more complex design that makes them act like black-boxes. Post-hoc explanation tools are required to provide some degree of interpretability. Different approaches have been proposed such as global model-agnostic explanation methods, simulation-based methods or local explanation methods.

Global model-agnostic explanation methods

These post hoc methods are used to describe the average behavior of a machine learning model. They quantify the main effects and the interaction effects of the

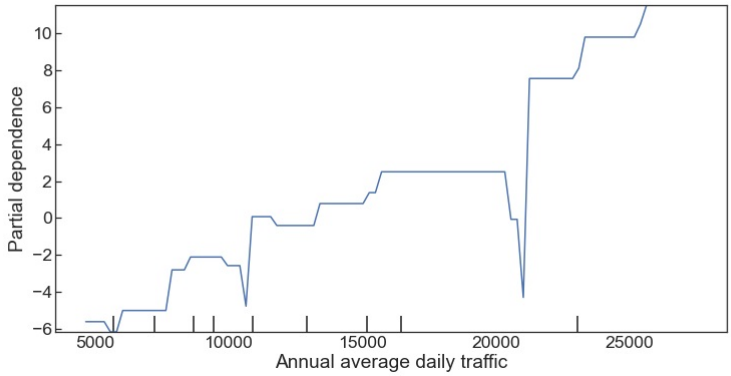


Figure 2.2: Partial dependence of the annual average daily traffic for a gradient boosting tree model on the highway dataset

explanatory variables on the dependent variable. Let us introduce the most prominent representatives of these methods.

Partial Dependence plots First introduced by Friedman [39], partial dependence plots (PDP) show the dependence between the dependent variable and a set of features of interest, marginalizing over the values of the complementary features:

$$pd(x_S) = E_{X_C}[\hat{f}(x_S, X_C)] = \int \hat{f}(x_S, x_C)p(x_C)dx_C$$

where $\hat{f}$ is the ML model, x_S are the features of interest and X_C are the complementary features here treated as random variables. As we marginalize over the other features, we get a function that depends only on features in S [91]. We obtain the PDP by computing the integral for various values of x_S. This plot provides insights into the nature of the relationship between the target and a feature (see Fig. 2.2). However, one main limitation of PDP lies in its assumption of independence between features which may result in the transmission of misleading information.

Sobol indices In [116], based on a finite expansion of a multivariate function $\hat{f}$ and under several constraints to maintain its unicity, Sobol was able to decompose the global variance $V(Y)$ into parts attributable to input features and combinations of features:

$$V(Y) = \sum_{i=1}^{n} V_i + \sum_{i}^{n} \sum_{j>i}^{n} V_{ij} + \cdots + V_{12\ldots n} \tag{2.4}$$

where

$$V_i = V(\hat{f}_i(x_i)) = V(E[Y|x_i])$$
$$V_{ij} = V(\hat{f}_{ij}(x_i, x_j)) = V(E[Y|x_i, x_j]) - V_i - V_j$$
$$\dots$$

From Eq. 2.4, one can estimate the main effect of a feature x_i with the first-order Sobol indices, which is the amount of variance in the output explained by that feature:

$$S_i = \frac{V(E[Y|X_i])}{V(Y)}$$

Simulation-based methods

These methods seek to obtain a global understanding of a black-box model, not by using global post hoc analysis tools, but by simulating its prediction function with a simpler one. Tan *et al.*[120] proposed to leverage model distillation techniques [18, 50] to learn global additive explanations of the form:

$$\hat{H}(\mathbf{x}) = h_0 + \sum_i h_i(x_i) + \sum_{i \neq j} h_{ij}(x_i, x_j) + \sum_{i \neq j} \sum_{j \neq k} h_{ijk}(x_i, x_j, x_k) + \dots \tag{2.5}$$

where h are either splines or bagged trees. During training, they seek to minimize the mean squared error between the prediction function of the black-box model and $\hat{H}(\mathbf{x})$. If h are splines, they use penalized maximum likelihood and estimate the smoothing parameters with cross-validation. If h are bagged trees, cyclic gradient boosting [76] is used. To understand the behavior of the model, they analyze the feature shapes by plotting the feature's contribution $h_i(x_i)$ against the domain of x_i.

Another approach, proposed by Lakkaraju *et al.*[68], use sub-space explanations to mimic the behavior of black-box models in classification problems. They leverage a two-level decision set representation, where the outer decision rules describe the sub-spaces (i.e. specific regions of the feature space), and the inner decision rules explain the decision logic of the black-box model within the related sub-spaces. Moreover, their approach allow end users to customize explanations by adding certain features of interest in the outer level representing the sub-spaces.

Local model-agnostic explanation methods

The objective of local explanation method is to learn local approximations to explain individual predictions. A broad variety of methods has been proposed and will be discussed in this section. First, we start by explaining how a contribution of a feature

is computed on a simple linear model prediction and how it is extended to any model by means of Shapley values, a game theoretic approach. Then, we introduce LIME, a method to interpret individual model predictions based on locally approximating the model around a given prediction. Finally, we explain how Shapley values and LIME have been connected together to provide an efficient tool for local model-agnostic explanations.

Simple computation of contributions in a linear model setting A simple linear model prediction for an instance x is computed with:

$$\hat{f}(x) = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$$

where β_i is the weight corresponding to variable i, and x_i the value of variable i on this instance. The contribution ϕ_i of the i-th feature can be computed by:

$$\phi_i(\hat{f}) = \beta_i x_i - E[\beta_i X_i]$$

where $E[\beta_i X_i]$ is the average effect of feature i [91]. Thus, with a linear model, computing a feature's contribution to a prediction is straightforward. To measure how much a feature affects the prediction of more complex nonlinear models, a game theoretic approach based on the computation of Shapley values has been developed.

Model-agnostic computation of a contribution with Shapley regression value Shapley regression values [112] measure how much feature i contributes to the prediction of any model $\hat{f}$. Specifically, if S is a subset of the p features considered in the model, then the Shapley value ϕ_i of a feature i is defined as:

$$\phi_i(\hat{f}) = \sum_{S \subseteq \{1,\dots,p\} \setminus \{i\}} \frac{|S|!(p - |S| - 1)!}{p!} (v(S \cup \{i\}) - v(S)) \tag{2.6}$$

where, for a given instance x, $v_x(S)$ is the prediction of feature values in S that are marginalized over features not included in S [91]:

$$v_x(S) = \int \hat{f}(x_1, ..., x_p) d\mathbb{P}_{x \notin S} - E_X[\hat{f}(X)]$$

Local interpretable model-agnostic explanations (LIME) Ribeiro *et al.* [102] introduced LIME to approximate the model's behavior in the neighborhood of a given

instance. For an instance x, they build a locally faithful and interpretable surrogate model g by solving:

$$\xi(x) = \arg\min_{g \in G} \mathcal{L}(\hat{f}, g, \pi_x) + \Omega(g) \tag{2.7}$$

with $\mathcal{L}$ a loss function, $\hat{f}$ the model being explained, G a set of interpretable models (e.g., decision trees, linear models), and Ω a complexity measure. In Eq. 2.7, Ω can be perceived as a regularization term that constrains the surrogate model g to provide short explanations. Moreover, to ensure local fidelity, Ribeiro *et al.* define $\mathcal{L}$ as being the locality-aware square loss:

$$\mathcal{L}(\hat{f}, g, \pi_x) = \sum_{z,z'} \pi_x(z)(f(z) - g(z'))^2 \tag{2.8}$$

where z' are binary vectors sampled at random from $x' \in \{0,1\}^p$ — *viz.*, the interpretable representation of the instance $x \in \mathbb{R}^p$, p being the number of input features, which indicates if the feature i is present ($x'_i = 1$) or missing ($x'_i = 0$) in the surrogate model — and z are the original representations of z'. In Eq. 2.8, π_x is a proximity measure defined by the authors as being the exponential kernel: $\pi_x = \exp(-d(x,z)^2/\sigma^2)$, with d a distance function and σ the kernel width.

LIME creates sparse linear explanations with a measure of faithfulness in the neighborhood of a given instance thanks to the introduction of the kernel π_x. However, LIME highlights some limitations regarding the definition of this neighborhood. Defining a good neighborhood in an automatic manner with a proper kernel is still in development. For now, the selection of appropriate kernel settings is made heuristically. Also, as the neighborhood is computed based on points sampled uniformly at random, the analysis can be performed on unlikely data points. Moreover, recent work [114] tend to show that it can be difficult to assess its reliability and robustness. In a theoretical study of LIME verified by simulations, Garreau and Luxburg [114] showed that, while this post-hoc approach "discovers interesting features, it might forget some important features and the surrogate model is not faithful".

Shapley additive explanation (SHAP) Lundberg and Lee [79] unify existing approaches such as LIME [102], Shapley values [112], DeepLIFT [113], etc. SHAP quantifies the contribution of each original explanatory variable to each prediction. The authors proposed to represent Shapley value explanations as an additive feature attribution method. Let h_x be an input mapping from simplified inputs x', indicating feature presence, to original inputs x such that $x = h_x(x')$, and $\hat{f}$ a black-box model. An additive feature attribution model tends to reach:

$$g(z') \approx \hat{f}(h_x(z')) \quad \text{whenever} \quad z' \approx x'$$

Moreover, a model g in this class is a linear function of binary variables:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i \tag{2.9}$$

where the z'_i variables indicate if the feature is observed ($z'_i = 1$) or unknown ($z'_i = 0$), $\phi_i \in \mathbb{R}$ are the feature attribution values, and M is the number of input features.

The authors state that there is a single unique solution that resolves this linear function while having the three following desirable properties [79]:

- *local accuracy*: the explanation model g shall match the output of the original model $\hat{f}$ for the simplified input x'
- *missingness*: a missing feature has no impact
- *consistency*: "if a model changes so that some simplified input's contribution increases or stays the same regardless of the other inputs, that input's attribution should not decrease" [79]

They prove that the coefficients ϕ of this linear function in fact corresponds to the Shapley values of the features:

$$\phi_i(\hat{f}, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} [\hat{f}_x(z') - \hat{f}_x(z' \setminus i)]$$

with M the number of simplified input features, $|z'|$ the number of non-zero entries in z', and $z' \subseteq x'$ represents all z' vectors where the non-zero entries are a subset of the non-zero entries in x'.

In the same paper, the authors introduce KernelSHAP, a model-agnostic approximation method that connects linear LIME and Shapley values. They prove that, given an appropriate weight function $\pi_{x'}(z')$, for an instance x, the Shapley values of the original features are the coefficients of the linear model g mimimizing the loss function defined in eq. 2.7:

$$\mathcal{L}(\hat{f}, g, \pi_x) = \sum_{z' \in Z} [\hat{f}(h_x(z')) - g(z')]^2 \pi_{x'}(z')$$

where:

$$\pi_{x'}(z') = \frac{M - 1}{\binom{M}{|z'|} |z'|(M - |z'|)}$$

2.1.4 Model interpretability in the ML community

The machine learning research community offers nuanced perspectives about the merits of post-hoc explanations. As a representative example, Lipton [72] suggests that post-hoc explanations should not be ruled-out as valid, although indirect, means of knowledge about the underlying data generating process. Lipton also underlines the potential risk of focusing on misleading information when relying on post-hoc explanations. Moreover, he considers that transparent linear models may not always be more interpretable than deep neural networks (DNN) because they often need heavily engineered features to obtain similar performances. Likewise, Poursabzi-Sangdeh *et al.* [99] observe that practitioners can be affected by the *information overload* phenomenon [3, 63] when the number of features becomes too large. Otherwise, Rudin [104] emphasizes the importance of taking into account the whole data analysis process, including the preprocessing steps: "when considering problems that have structured data with meaningful features, there is often no significant difference in performance between more complex classifiers (DNN, boosted decision trees, random forests) and much simpler classifiers (logistic regression, decision lists) after preprocessing". Rudin points out that there is not necessarily a trade-off between accuracy and interpretability: performance gaps can be reduced iteratively through better data processing and model understanding. The latter is facilitated by the use of interpretable models.

2.2 Crash prediction models and their interpretability

In their survey, Lord and Mannering [75] describe the different methods used for long-term crash frequency analysis. These methods can be classified in two categories: on one side, parametric statistical models explicitly associate the crash related variable to a vector of input explanatory variables. On the other side, non-parametric models use more complex design to better fit the data at the expense of their interpretability. In this section, we describe both approaches, starting with the parametric statistical models.

2.2.1 Parametric statistical models

Poisson regressions

Crash-count data observations y_i being positive integers, they have originally been modelled by Poisson regressions [58, 59, 88]. Poisson distribution is a special shape of the binomial distribution with a small probability of an event and a large but

unknown number of trials. GLM have been used to link a linear predictor made of p explanatory variables to the rate λ_i of a Poisson likelihood. As explained in section 2.1.2, an appropriate canonical link function is chosen to map the definition space of crash data (*viz.*, discrete and positive only) to the real line covered by the linear predictor. For Poisson regressions, a *log*-link function is selected. The model can be formulated as:

$$y_i \sim Poisson(\lambda_i)$$
$$log(\lambda_i) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}$$

Negative Binomial regression

The Poisson distribution has a unique free parameter, the rate λ_i. The variance cannot be adjusted independently of the mean. Yet, the observed variance of crash count data often exceeds this amount. A negative binomial (NB) distribution can be proposed to better deal with this over-dispersion phenomenon. Given the probability of crash, the NB gives the probability of observing n crashes before the α-th non-crash. With λ representing the mean, its probability mass function can be parameterized as follows [130]:

$$\mathrm{NB}(n; \lambda, \alpha) = \binom{n+\alpha-1}{n} \left(\frac{\lambda}{\lambda+\alpha}\right)^n \left(\frac{\alpha}{\lambda+\alpha}\right)^\alpha$$

The variance of the NB is $\lambda + \frac{\lambda^2}{\alpha}$. Therefore, the parameter α controls the amount of over-dispersion. When $\alpha \to \infty$, the NB likelihood approaches a $Poisson(\lambda)$ distribution. Moreover, as shown in [130], a $\mathrm{NB}(\lambda, \alpha)$ corresponds to a $Poisson(\lambda)$ where λ comes from a gamma distribution $Gamma(a = \frac{\alpha}{\lambda}, b = \alpha)$ whose probability density function is defined as:

$$Gamma(x; a, b) = \frac{a^b x^{b-1} e^{-ax}}{\Gamma(b)}$$

for $x > 0$, $\Gamma(b)$ being the gamma function.

The Poisson-gamma model — a continuous mixture of Poisson distributions with rates distributed as a gamma distribution — has been identified as a reference model by road safety experts (see for example the AASHTO Highway Safety Manual [8]). We find it in the core of many related work [89, 74, 31, 73] where crash count regression is done with a linear model attached to the λ parameter of the NB through a log-link function.

Generalized linear mixed model

In some studies, correlation among observations can be taken into account on the temporal level. This arises in the context of panel data analysis, where crash data are considered to be collected through a series of repeated observations of the same groups (e.g., roadway segments) over a time period. To account for the differences between segments, GLM are accommodated to introduce random effects in addition to the usual fixed effects. The resulting model is known as a generalized linear mixed model (GLMM). In this configuration, Poisson regressions are adapted as follows:

$$y_{ik} \sim Poisson(\lambda_{ik})$$
$$log(\lambda_{ik}) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + u_k$$

where u_k is the random effect for group k, supposed to be distributed according a zero-centered Gaussian distribution with variance σ^2 ($\sigma > 0$).

GLMM have found many applications in road safety analysis. Amongst them, Johansson *et al.* [56] analyzed the effect of a lowered speed limit on the number of crashes on Swedish highways. In a study of crashes caused by median crossovers in Washington State, Shankar *et al.* [111] compared a standard NB and a NB modified to account for random effects relating to each site. However, they did not observe any benefits from using a NB with random effects.

Hierarchical model

GLM can also be refined into hierarchical models to take into account clusters of related observations. For example, crashes occurring in a given geographical region may possess specific characteristics while not differing entirely from crashes in other regions. A simple Poisson regression, by pooling all the observations together, would assume an invariant population and couldn't benefit from regional peculiarities. Otherwise, k clusters could be modeled with the addition of $k-1$ mutually exclusive binary variables to the linear model, but this would correspond to no pooling at all and the clusters would be assumed independent of one another. Contrariwise, hierarchical models, also known as multilevel models, offer partial pooling through an adaptive regularizing prior. Thus, in the following multilevel Poisson regression with k clusters, hyperpriors μ and σ will allow an adaptive shrinkage of the cluster-specific β_{jk} towards a common mean:

level 0 - model

$$y_{ik} \sim Poisson(\lambda_{ik})$$
$$log(\lambda_{ik}) = \beta_{0k} + \sum_j \beta_{jk} x_{ijk}$$

level 1 - priors

$$\beta_{jk} \sim \mathcal{N}(\mu, \sigma)$$

level 2 - hyperpriors

$$\mu \sim \mathcal{N}(0, 100)$$
$$\sigma \sim \mathcal{HN}(100)$$

with $\mathcal{HN}$ being the half-normal distribution. Gelman indicates in [42] that a half-normal distribution with high standard deviation is a non-informative but proper prior for the variance parameter σ.

In this way, Jones and Jørgensen [57] design a multilevel model to predict the severity of an incident given the involved casualties (level 1), their respective vehicles (level 2) and the accident location (level 3). Ahmed *et al.* [4] conceive a multi-level model to predict crashes on a mountainous freeway by modeling both the dry or snow seasons and spatial correlation between adjacent sites. Deublein *et al.* [28] propose a multilevel model to manage simultaneously 4 response variables (resp. injury accidents, light injuries, severe injuries and fatalities) through gamma updating of the parameters. Finally, Fawcett *et al.* [34] predict future safety hotspots with a multilevel model where, first, the variance of the rate of a NB increases with the timestamp of an observation and, second, a global trend effect is altered by site-specific ones.

Intrinsic interpretability

The aforementioned models are interpretable by nature. Indeed, the relation between the explanatory variables and the dependent one is explicit. Nevertheless, as Poisson regressions and their variants use a log-link function to map the linear predictor to the crash-related variable, the effects of the coefficients β are not as obvious as those of a linear model. To mitigate this, Kweon and Kockelmam [65] proposed to use the incidence rate ratio which measures the percentage of change in the crash related variable when an explanatory variable is increased by a unit.

Moreover, analyzing the effects of coefficients shall not be based solely on point estimates but must come along with the knowledge of the uncertainty associated with them. Most of studies mitigate this by using Bayesian inference. By associating a probability distribution to each coefficient, they can measure the underlying uncertainty of coefficients.

2.2.2 Non-parametric Machine Learning models

Support vector machines

SVM were originally introduced by Cortes and Vapnik [25] for classification tasks. A version for regression has been developed by Drucker *et al.* [30]. Among the variants proposed for regressions, the epsilon-insensitive SVM (ε-SVM) is the most common. Given a vector $y \in \mathbb{R}^n$ of dependent variables and $\mathbf{X} \in \mathbb{R}^{n \times p}$ the associated explanatory variables, ε-SVM aims to find a function $f(\boldsymbol{x}_i)$ that deviates from y_i by a value inferior to ε for each training point $\boldsymbol{x}_i, i \in \{1, ..., n\}$ (see Fig. 2.3). In addition, f has to be as flat as possible. To achieve this, ε-SVM solves the following problem:

$$\begin{aligned} \min_{w,b,\zeta,\zeta^*} \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^*) \\ \text{subject to} \quad y_i - w^T \phi(\boldsymbol{x}_i) - b \leq \varepsilon + \zeta_i, \\ w^T \phi(\boldsymbol{x}_i) + b - y_i \leq \varepsilon + \zeta_i^*, \\ \zeta_i, \zeta_i^* \geq 0, i = 1, ..., n \end{aligned} \tag{2.10}$$

where ζ_i, ζ_i^* penalize observations lying outside the ε margin and C is a positive regularization constant that controls the penalty imposed by ζ_i and ζ_i^*. In this problem, ϕ is a transformation function that maps x to a higher dimensional space where the problem resolution is linear. Without it, ε-SVM won't have the ability to model non-linear relationship between explanatory variables and the dependent variable in the original space.

The optimization problem described above is computationally simpler to solve by constructing a Lagrangian function which introduce positive multipliers α_i and α_i^* for each observation x_i [30]:

$$\begin{aligned} \min_{\alpha,\alpha^*} \frac{1}{2} (\alpha - \alpha^*)^T G (\alpha - \alpha^*) + \varepsilon \mathbf{1}^T (\alpha + \alpha^*) - y^T (\alpha^* - \alpha) \\ \text{subject to} \quad \mathbf{1}^T (\alpha - \alpha^*) = 0 \\ 0 \leq \alpha, \alpha^* \leq C, i = 1, ..., n \end{aligned} \tag{2.11}$$

In eq. 2.11, G is a $n \times n$ positive semidefinite matrix whose elements are computed with the kernel function k: $G_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ ($\langle \cdot, \cdot \rangle$ being the inner product). Here, the explicit mapping ϕ can be avoided as we can obtain G by computing G_{ij} directly with the kernel function. This process is known as the kernel trick. By doing so, ε-SVM estimates f in the transformed input space and not the original one. To generate this transformation, various kernels have been proposed: linear ($k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i \boldsymbol{x}_j$), polynomial ($k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (1 + \boldsymbol{x}_i^T \boldsymbol{x}_j)^d$ with d the

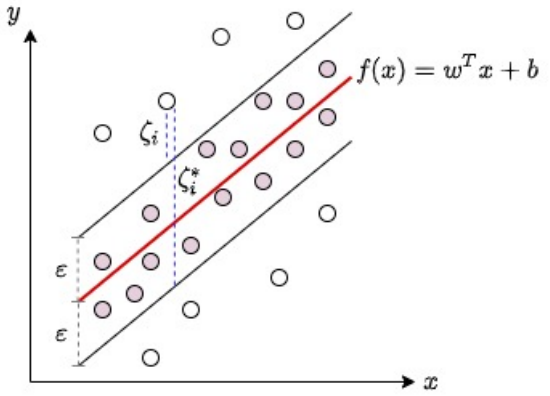


Figure 2.3: Toy example of a linear SVM, with the red line being the optimal hyperplane.

degree of the polynomial), or based on radial basis function ($k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/\sigma^2)$.

Finally, to make a prediction, ε-SVM computes the contribution of each support vector (i.e. a vector $\boldsymbol{x}_i$ for which α_i or α_i^* is equal to zero):

$$f(\boldsymbol{x}) = \sum_{i \in S} (\alpha_i - \alpha_i^*) k(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

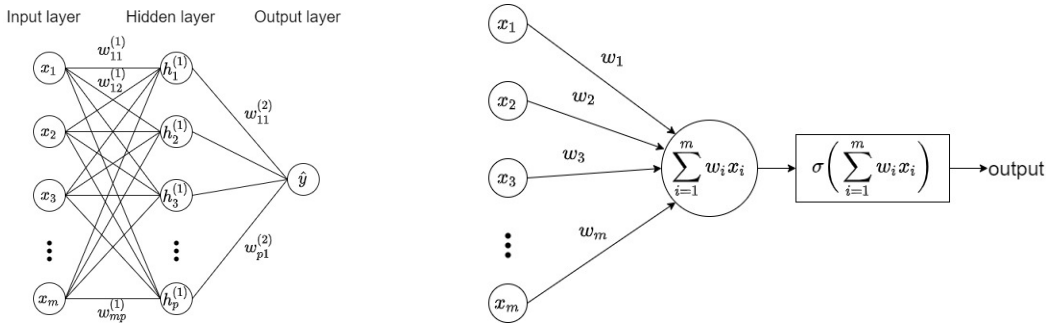
where S is the set of support vectors.

In the binary classification problem, the objective is not to find an hyperplane that contains a maximum number of observations within its ε margin, but to determine the best decision boundary between the two classes. In this case, the goal is to maximize the margin between the closest vectors of each class and the hyperplane.

In highway safety analysis, SVM have been applied to various problems. Li *et al.* [70] compare a NB and a SVM on crash-count predictions for rural roadway segments located in Texas, USA. They observe that SVM predict crash data more effectively and accurately than traditional NB. In a study on crash data collected at 326 freeway diverge areas, Li *et al.* [71] apply SVM to the task of predicting crash severities. When compared to an ordered probit model, they found that SVM produces better prediction performance. Lately, Dong *et al.*[29] study the effects on crash-count predictions of spatial correlations at different scales on a dataset including more than 57 000 crashes in the Hillsborough county of Florida.

Artificial neural networks

Artificial neural networks (ANN) are biologically inspired computational networks first introduced by McCulloch [81]. ANN are composed of artificial neurons typically



(a) (b)

Figure 2.4: Multilayer perceptron (a) simple architecture with one single hidden layer (b) toy example of the computation of an output

organized into a multi-layer structure. Neurons of one layer are connected to neurons of the next layer. An ANN generally consists of three types of layers: the input layer receives the external data, then the hidden layer(s) process the data and finally, the output layer produces a result (e.g. crash-count prediction). In the well-known configuration of multilayer perceptrons (MLP), neurons are fully connected i.e., every neuron in one layer is connected to every neuron in the next layer (see Fig.2.4a). Each neuron has inputs and produces a single output by first computing weighted sum of its inputs and then applying a nonlinear activation function σ such as sigmoid or hyperbolic tangent (see Fig. 2.4b).

When training an ANN, the objective is to adjust the connection weights to minimize the cost function that measures the overall performance of the network. One of the most common approach is backpropagation: first, it calculates the gradient of the cost function associated with a given state with respect to the weights, and then updates the weights in the opposite direction of the gradient, as it is the direction of the steepest descent. Several variants of gradient descent have been developed and are well explained in [103].

ANN have found numerous applications in highway safety analysis. Abdelwahab and Abdel-Aty [1] compares a backpropagation neural network (BPNN) with an ordered logit statistical model to predict the severity of accidents at intersections. Chang [21] compares a one hidden layer BPNN with NB regression for crash frequencies prediction. BPNN slightly outperforms the NB model with a difference of 0.6% of accuracy on testing data. Huang *et al.* [53] compare a radial basis functions neural network (RBFNN) with BPNN and NB regression for crash frequencies prediction. RBFNN obtains the best results. Xie *et al.* [133] use a bayesian neural network for crash-count prediction. The neural network model outperforms BPNN and NB regression.

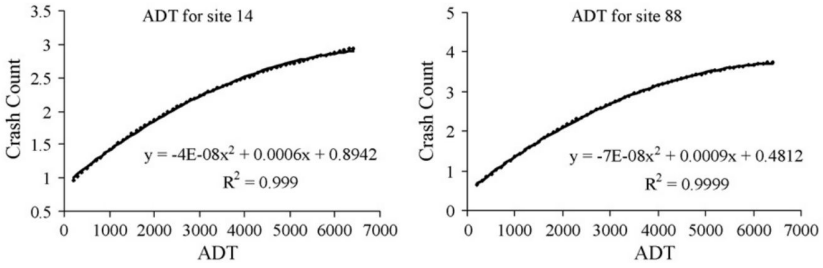


Figure 2.5: Sensitivity analysis performed on a Bayesian neural network for the traffic variable, from Xie *et al.* [133, p.930]

Post-hoc interpretability

SVM and ANN are opaque decision systems. The former, by introducing the kernel function, models complex relationships in a high-dimensional space. The later often involves a large number of parameters growing exponentially with the number of hidden layers. To reduce the complexity of the models, one can restrict the kernel of SVM to be linear or define a simpler architecture for the ANN (e.g. a perceptron with a single hidden layer with few neurons). However, trying to reduce complexity in this way will tend to deteriorate the predictive results. To balance performance and interpretability, all studies of the previous section opted instead for the use of post-hoc explanation methods. Authors perform a sensitivity analysis of the black-box model: for each explanatory variable, while keeping all other variables unchanged, they record the effect on the output prediction of a perturbation of the current variable (see Fig. 2.5). However, as stated in [133], the relationship between the current explanatory variable and crash frequency may vary due to correlated variables, making it difficult to interpret the sensitivity plots. Moreover, by generating simulated data while assuming the explanatory variables to be independent, sensitivity analysis will indiscriminately generate potentially misleading hypothetical predictions for unlikely data points. Finally, this method only provides global interpretations and does not allow domain experts to identify roadway segments where the predictive model behaves singularly. To mitigate this, some studies used local model-agnostic explanation methods. Mihaita *et al.* [90] investigate with SHAP the impact of different features on arterial incident duration. They observe that the number of affected lanes and the hour of the day seem to be the most important features which increase the incident duration prediction. Parsa *et al.* [93] extend the use of SHAP by analyzing complex and nonlinear joint impacts of features on the output of a XGBoost model, in the context of real-time accident detection.

Chapter 3

Contributions

3.1 Overview

3.1.1 Introduction

In previous chapters, we stressed the need to be able to provide explanations in addition to predictions when actions are pondered based on the analysis of predictive models. To obtain such explanations, we saw that two approaches have been developed: intrinsically interpretable models that offer a direct understanding of their internal decision process, versus post-hoc analysis of black-box models. However, we have seen that using post-hoc explanation methods comes with some limitations. For data-driven high-stakes decisions, it is imperative that the explanations transparently and truthfully reflect what the model has learned. Moreover, as observed by Rudin [104], simple and interpretable models can achieve predictive performance similar to black-box models on well-structured tabular data. For these reasons, we favored the use of simple interpretable models.

The focus of our work is to enhance the predictive power of simple models without deteriorating their high degree of interpretability. Our main contributions aim to achieve this goal in two steps. First, we introduce a supervised method to discover a partition of the original observations and build a hierarchical model above it. Second, we introduce two algorithmic approaches (*viz.*, a polynomial neural network, and an extension of multi-objective symbolic regression) to discover highly discriminant non-linear transforms of the original variables. The former can handle correlations among groups of observations which usually lead to improvements in the quality of the models' predictions and of their interpretation. The latter, while remaining simple (e.g. first-order interactions), allow the models to capture more of the variability in the dependent variable.

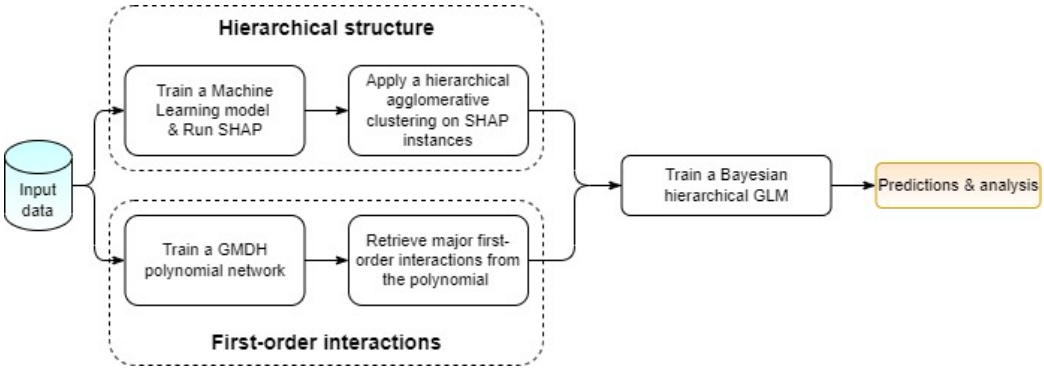


Figure 3.1: Description of a Bayesian hierarchical generalized linear model

3.1.2 Bayesian hierarchical generalized linear model

In our first proposal presented in Fig. 3.1, we describe how to extract, from data and without prior expert knowledge, the information necessary to build a hierarchical Bayesian model with first-order interactions between explanatory variables to efficiently solve regression or classification problems while preserving interpretability. First, in the *hierarchical structure* module described in section 3.2, we elucidate how Shapley values of the variables give rise to a clustering of the original observations that is likely to make sense in terms of the problem to be solved. This clustering then informs a multilevel Bayesian model (see section 3.3). Second, we explain how a self-organized neural network reveals the most important interactions between explanatory variables (*viz.*, *first-order interactions* module). These interactions are then integrated to the functional form of the multilevel model (see section 3.3).

3.1.3 Interpretable hierarchical symbolic regression

In our second proposal (see Fig. 3.2), the discovered hierarchical structure is even better exploited by using symbolic regression to capture sparser, more complex but still interpretable relationships between the explanatory variables and the crash count. To do so, to the whole training dataset and to each cluster of the hierarchical structure, we apply a variant of the symbolic regression (SR) method to find expansions of linear models with effective and interpretable functional forms. To this end, presented in section 3.4, we designed a multi-objective simulated annealing algorithm to solve the SR problem. Thus, we can discover Pareto optimal predictive models[1] with various trade-offs between accuracy and complexity. In our case, symbolic regression serves two purposes. First, it discovers *global models*, learned from the whole training

[1]A model is said to be Pareto optimal if there is no alternative model that can improve one of its objective function without deteriorating the others.

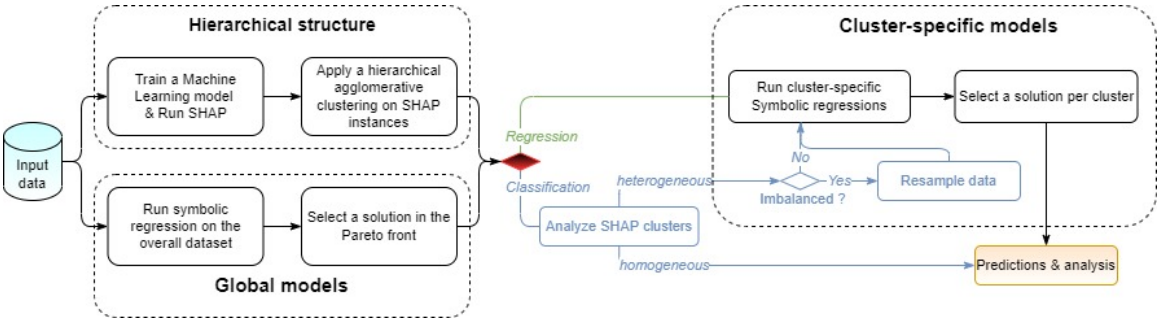


Figure 3.2: Proposed framework for an interpretable hierarchical symbolic regression

dataset, that capture the associations between the explanatory variables and the target. Second, based on the *hierarchical structure* of the instances, our SR-based algorithm implements a partial pooling strategy to refine a global model into *cluster-specific* ones.

3.2 Supervised learning of a hierarchical structure

3.2.1 Introduction

As explained in section 2.2.1, a well-chosen hierarchical structure can handle correlations among groups of observations and significantly improve the quality of the models' predictions and of their interpretations. However, in the literature, this structure, which is dependent on the dataset, is the result of either expert knowledge or unsupervised clustering. The former is not always available, and the latter does not account for the task being solved (e.g., prediction of crash counts) and tries to group features without knowing if they're relevant for an outcome of interest [77]. Moreover, in our work, we would like to group roadway segments that share a similar crash severity for similar reasons. In other words, when computing clusters, we do not want to focus only on the distribution of the crash-related variable, but also want to account for the influence of each explanatory variable. Indeed, a similar crash severity between segments may be explained by different risk factors. Thus, clustering segments by considering only the dependent variable may result in segments in a same cluster that are in fact hazardous for different reasons.

To obtain such a hierarchical structure, we propose to learn it from the data in two stages. First, we analyze the results of a black-box machine learning model with a local post hoc explanation tool (*viz.*, SHAP). More specifically, for each instance of the dataset, SHAP computes the contribution of each explanatory variable to the prediction. Then, we apply a clustering on these SHAP explanation instances. In

other terms, we make use of an unsupervised clustering where samples are grouped together based on their *explanations* (*viz.*, the SHAP explanation instances), thus alleviating the aforementioned shortcomings of unsupervised clustering.

In this section, we describe how we successfully retrieve a relevant hierarchical structure with an unsupervised clustering on labeled datasets[2]. First, we start by presenting gradient boosting trees, the models we selected to discover a latent structure in the data. Then, we explain how the black-box effect induced by these models is mitigated by leveraging a local explanation tool (*viz.*, SHAP) to measure the features' contributions to each observation. Finally, we explain how a hierarchical agglomerative clustering applied to these SHAP explanation instances help to discover a relevant structure made up of similarities of explanations between observations.

3.2.2 Training of a non-parametric ML model

Model selection

We are interested in highly flexible, efficient models with fast training time on tabular data. To answer the first two requirements, numerous models can be considered: nonlinear SVM, ensemble of trees, deep learning models, ... However, training a nonlinear SVM is computationally infeasible on large dataset (between $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$, n being the number of samples, for a standard implementation of SVM [2]) which make them poorly suited to the preliminary task of discovering a hierarchical structure in the data. On their side, deep learning models are better adapted to image recognition, natural language processing or speech recognition [77].

In their experiments, Chen *et al.* [24] observe that tree-based models consistently outperform standard deep learning models on tabular datasets where "features are individually meaningful and do not have a strong multi-scale temporal or spatial structures". Recently, Borisov *et al.* [14] compare algorithms based on gradient-boosted trees with a broad variety of deep learning models on three middle and large size datasets covering two classifications and one regression task. They observe that gradient-boosted trees outperform deep learning models on all datasets, while having lower training time.

A balance of computational efficiency, ease of use, and high accuracy have made tree-based models the most popular non-linear model type. In 2021, 75% of most popular ML models used by data scientists and engineers are based on trees [61]. Among ensemble of trees, random forest or gradient boosting trees are the most popular. However, the former tends to reduce only the variance with a bagging

[2]Note that, in [77], Lundberg *et al.* call this process a *supervised clustering*. However, this usually refers to clustering with access to a "teacher" that knows the right cluster for some of the instances [12]. In our case, our approach for clustering is unsupervised.

strategy while the latter, by combining multiple models and gradient descent on residuals, reduces both the variance and bias. Plus, gradient boosting trees often outperform random forests [14]. For these reasons, we make use of gradient boosting trees as a first step to discover the hierarchical structure.

Gradient tree boosting

Given a dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, gradient tree boosting aims to build, through additive training, an ensemble of trees T that minimizes a loss function L (e.g., mean squared error for regression tasks, logistic loss for classification tasks). The algorithm starts by initializing the model with the most accurate constant α:

$$T_0(\boldsymbol{x}) = \arg\min_{\alpha} \sum_{i=1}^{n} L(y_i, \alpha)$$

Then, at each iteration m, the model is expanded in a greedy fashion such that:

$$T_m(\boldsymbol{x}) = T_{m-1}(\boldsymbol{x}) + \arg\min_{t_m} \left[\sum_{i=1}^{n} L(y_i, T_{m-1}(\boldsymbol{x}_i) + t_m(\boldsymbol{x}_i)) \right]$$

where t_m is a decision tree. In other words, the model T_m tries to correct the error of its predecessor T_{m-1} by adding a new decision tree, also called weak learner. However, choosing the decision tree that best reduces the loss function at each iteration has a high computational cost. Thus, assuming L is differentiable, the optimization process is simplified by leveraging gradient descent. The new decision tree t_m is trained on $\{(\boldsymbol{x}_i, r_{im})\}_{i=1}^n$ with r_{im} the residuals indicating locally the steepest direction. Specifically, r_{im} are defined as:

$$r_{im} = -\left[\frac{\partial L(y_i, T(\boldsymbol{x}_i))}{\partial T(\boldsymbol{x}_i)} \right]_{T(\boldsymbol{x})=T_{m-1}(\boldsymbol{x})}, \quad i = 1, \ldots, n$$

Then, t_m is added to the expansion such that:

$$T_m(\boldsymbol{x}) = T_{m-1}(\boldsymbol{x}) + \gamma_m(t_m(\boldsymbol{x}))$$

where γ_m is selected by minimizing the loss function of the new expansion:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, T_{m-1}(\boldsymbol{x}) + \gamma_m(t_m(\boldsymbol{x})))$$

The expansion process stops when a stopping criterion is reached (e.g., maximum number of iteration, satisfactory value of the loss).

Implementation

We select the `XGBoost`[3]'s implementation of gradient tree boosting. The major benefits of using this library lie in its high portability, scalability, and high performance as it provides state-of-the-art results on many problems [24, 14].

Moreover, the authors use three different methods to prevent overfitting [24]. First, they penalize the complexity of the model when optimizing the loss function. Their definition of model complexity penalizes both the number of leaves in the trained trees, and the leaf values by means of a $l2$-regularization term. Then, they add shrinkage, a technique proposed by Friedman [40], to control the influence of new added trees during the update process. Finally, they use features subsampling to lower the number of features available to each new tree and thereby increase the variance between trees.

3.2.3 Local post hoc explanations

The aforementioned model is an ensemble of trees which consists of hundreds, if not thousands, of trees grown sequentially (in our experiments, the number of trees is set to 100, the default value of the `XGBoost` package). Even though the authors penalize the complexity of each tree while training, the resulting model is hardly interpretable. To mitigate this black-box effect, we compute local explanations with SHAP [79] (see section 2.1.3), to understand how the model maps the original explanatory variables to an outcome (e.g., a crash count in the case of highway safety). In addition to retaining local faithfulness, SHAP presents important properties described in section 2.1.3. To each observation, SHAP associates a linear function g:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i \tag{3.1}$$

where M is the number of input features, $z'_i \in \{0,1\}^M$ and $\phi_i \in \mathbb{R}$ are their contributions which correspond to the game theoretic concept of Shapley values.

Among the different implementations of SHAP, we selected TreeSHAP, an efficient tree-based algorithm for fast and consistent computations of exact Shapley values [78, 77]. Compared to KernelSHAP, TreeSHAP reduces the complexity of Shapley value computation from exponential to low order polynomial time [78]. Aside from ensuring the properties defined in section 2.1.3 (*viz.*, local accuracy, missingness and consistency), TreeSHAP is also able to account for feature dependence. Moreover, in [77], the authors observed that TreeSHAP consistently outperforms alternative methods across a benchmark of 21 different local explanation metrics.

[3] `https://xgboost.readthedocs.io/en/stable/`

Figure 3.3: Force plot of the explanatory variables' contributions to the estimated crash count for a specific observation. On this example, the traffic has the biggest positive contribution and explains most of the crash count shift from its overall expected value

Furthermore, in Lundberg and Lee [79], the authors propose a *force plot* visualization (see Fig. 3.3) to materialize how much each contribution shifts the output relatively to the overall expected value of the `XGBoost` model. These contributions can be perceived as being *forces* that shift the output towards or away from the expected value of the model (hence the name of the plot): the higher the contribution, the larger the magnitude of the force is in Fig. 3.3. In practice, these plots help us understand visually how the black-box model behaves on single instances of our dataset. In the next section, we will describe how they also reveal a hierarchical structure in the data.

3.2.4 Discovery of a hierarchical structure

When applied to all observations, the SHAP forceplots can be clustered by similarities of their profiles. On the highway network dataset, we discover clusters of roadway segments which are similar based on their SHAP explanation instances (see Fig. 3.4): the left part of the figure characterizes roadway segments that are on average moderately hazardous, while the middle and far right parts represent highly hazardous segments.

To obtain such a structure in our tabular dataset, we use a hierarchical agglomerative clustering (HAC) of the observations based on the explanatory variables' contributions as provided by the SHAP analysis. We select the squared euclidean distance as a measure of distance between pairs of instances and the ward linkage criterion to measure the dissimilarity of groups of instances. Unlike other distance measures, one characteristic of the ward linkage criterion lies in the fact that it considers the inner-variability of clusters when computing distances between two clusters. More specifically, given A and B two clusters, it states that the distance d between these

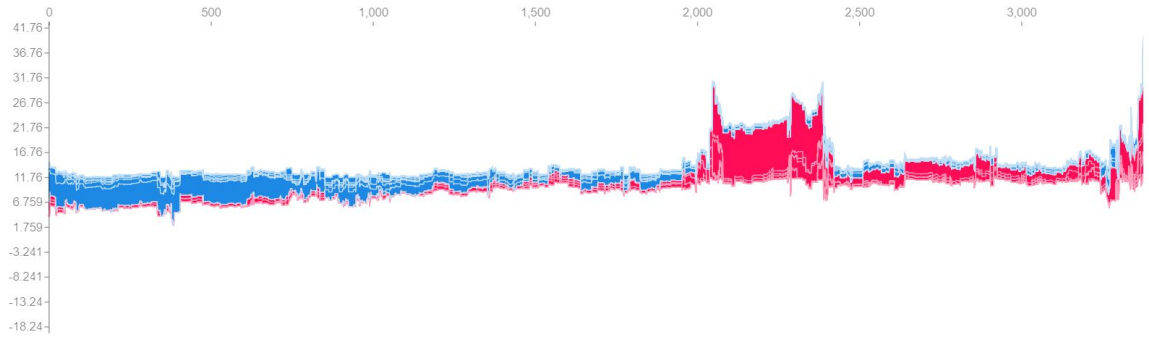


Figure 3.4: Observations grouped by similarity of their forceplots

two clusters is:

$$\begin{aligned} d(A,B) &= \sum_{i \in A \cup B} |x_i - m_{A\cup B}|^2 - \sum_{i \in A} |x_i - m_A|^2 - \sum_{i \in B} |x_i - m_B|^2 \\ &= \frac{n_A n_B}{n_A + n_B} |m_A - m_B|^2 \end{aligned}$$

where $|\cdot|^2$ is the norm of the vector, m_J the center of cluster J and n_J the number of SHAP instances in it. At the beginning, each cluster consists of a unique instance of SHAP and d is equal to zero. Then, at each iteration, clusters are merged together in a way that minimizes the increase in d. The procedure is repeated until a unique cluster is obtained, composed of the whole set of SHAP instances. Finally, we estimate the optimal number of clusters by detecting the greatest increase in the squared Euclidean distance between clusters when their number decreases (see Thorndike [121] for the original presentation of this widely used method).

Furthermore, in production, we do not have a straightforward way of knowing to which cluster a new observation belongs to. However, we need this information as our underlying predictive models require this prior knowledge. Thus, to learn how to associate an observation of the test dataset[4] to a cluster, we propose to train a decision tree classifier to approximate a good mapping between the original explanatory variables and the cluster of SHAP instances. In section 4.5.1, we report on cross-validation measures showing, on various datasets, that this association is very accurate.

[4]The test dataset materializes a production context as our model did not use these data for training.

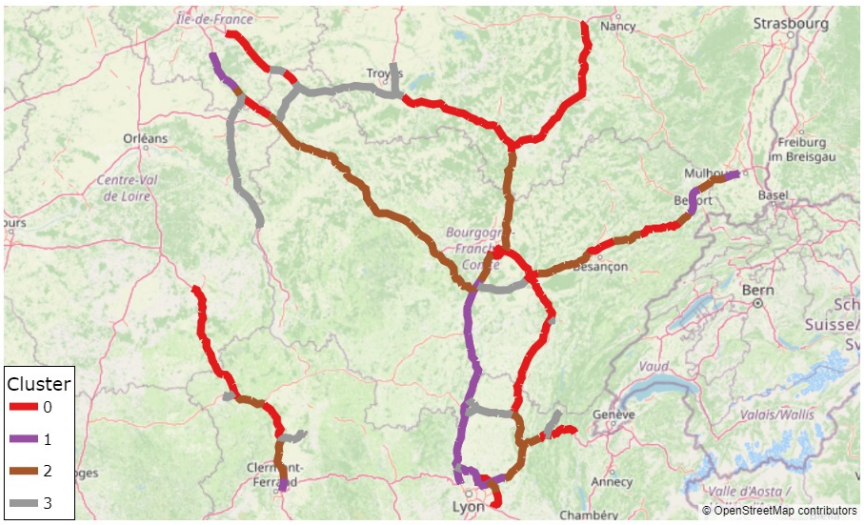


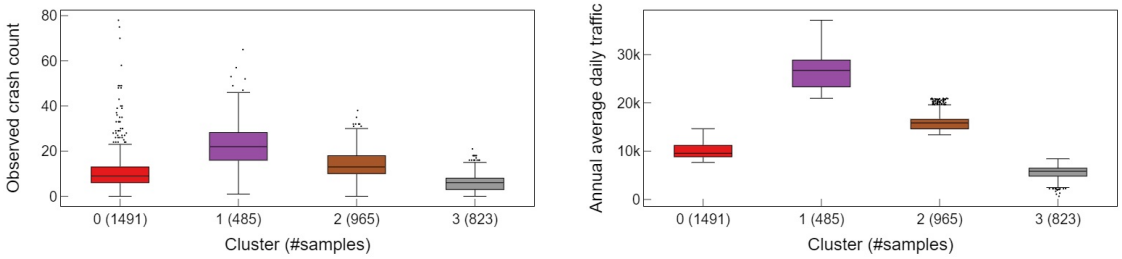
Figure 3.5: Clusters automatically discovered by the *hierarchical structure* module, for 2018

3.2.5 Representation of the hierarchical structure in highway safety

In this section, we suppose that a hierarchical structure has been discovered by using ten years of data, from January 1st, 2008 to December 31th, 2017. Four clusters have been identified and are presented in Fig. 3.5. To obtain additional information on these automatically discovered clusters, we propose to visualize the distributions of explanatory variables and the dependent variable. In Fig. 3.6a and Fig. 3.6b, we illustrate this for the observed crash counts and the annual average daily traffic (AADT) which allows us to understand that clusters do not have the same amount of historical crash counts, and present as well differences regarding the explanatory variables such as the traffic related one.

If necessary, experts can fine-tune the number clusters with the help of a dendrogram to obtain clusters that better match with their domain knowledge. They may be interested in broader analysis (with less clusters) or more detailed analysis (with more clusters) that highlight particular group of roadway segments.

When reducing the number of clusters from 4 to 2, one remains identical (*viz.*, Cluster 1 of Fig. 3.7) while the others are merged together. The difference between the two resulting clusters mainly lies in the Shapley values of the AADT, which explains most of the critical nature of cluster 1. When looking at the historical data, we observe that this cluster is more hazardous than cluster 0 (see Fig. 3.8a) and is composed of roadway segments with high traffic (see Fig. 3.8b) between Lyon and Dijon and in the vicinity of Paris and Belfort.



(a) Observed crash count

(b) Annual average daily traffic

Figure 3.6: Variables' distributions for 4 identified clusters

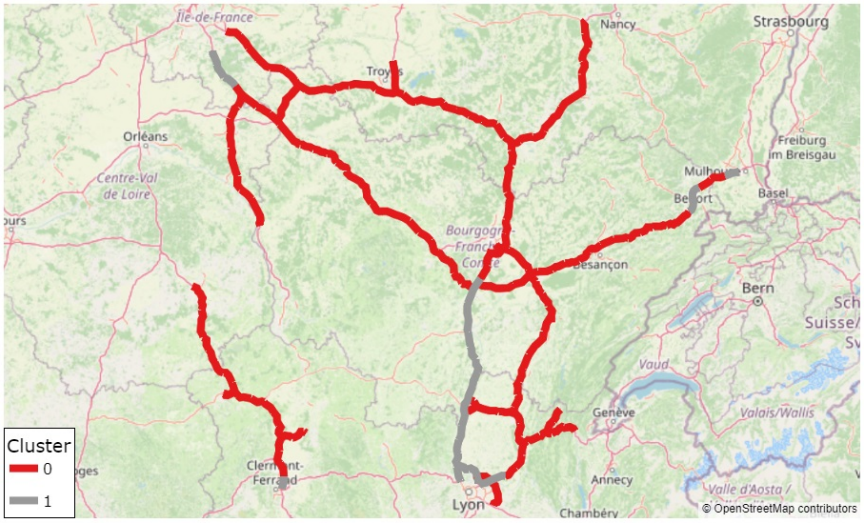
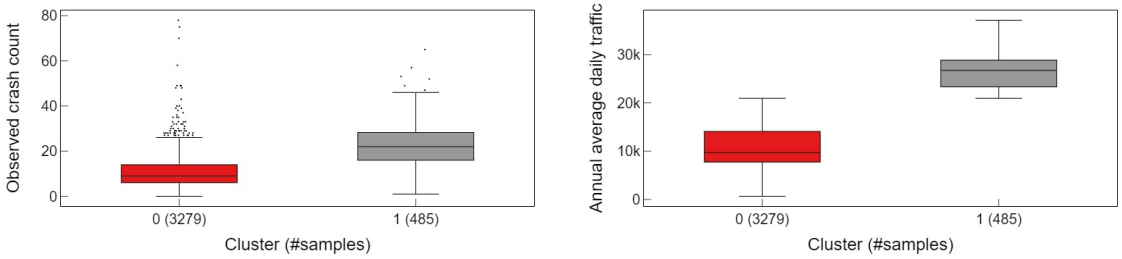


Figure 3.7: Clusters when the number of clusters is reduced to 2

Conversely, when increasing the number of clusters, some particular hazardous configurations, explained differently by the SHAP analysis, appear on the highway network. For instance, cluster 0 of Fig 3.5, a moderately hazardous cluster composed mainly of rural and mountainous segments, is now divided into cluster 1 and 4 in Fig. 3.9. The most hazardous segments of the previous cluster (see Fig. 3.6a) are now grouped into cluster 4. These segments are located in the mountainous part of the network (see Fig.3.10b).

However, moving away from the initial clusters defined by the automatic process is not without risks, especially when the number of clusters is increased. Indeed, increasing the number of clusters will cut the explanation space in a way that some clusters will only be represented by a few samples which will bring a high source



(a) Observed crash count (b) Annual average daily traffic

Figure 3.8: Variables' distributions for 2 identified clusters

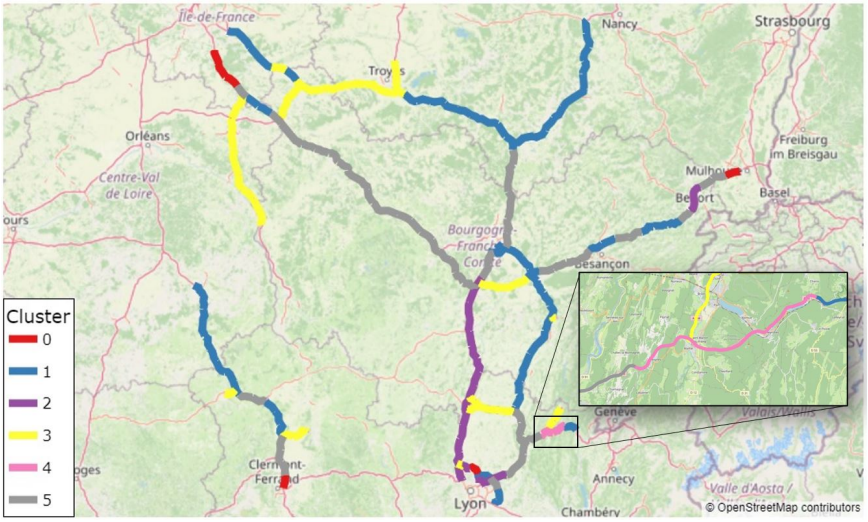
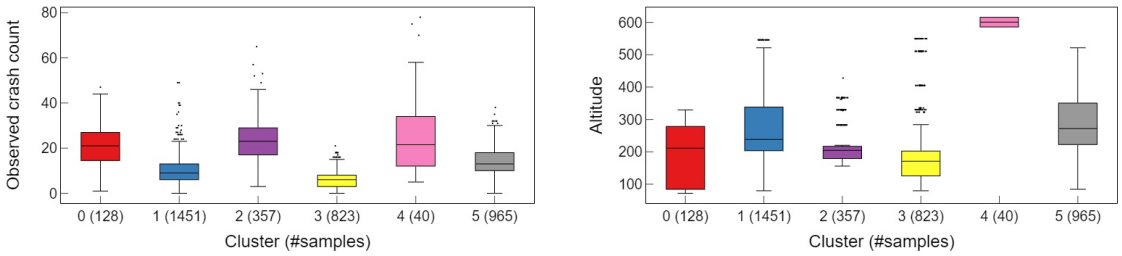


Figure 3.9: Clusters when the number of clusters is increased to 6



(a) Observed crash count (b) Altitude

Figure 3.10: Variables' distributions for 6 identified clusters

of uncertainty in the models. Moreover, we observed in our experiments that the performance is altered when the number of clusters moves away from the original value defined in the supervised way. We recommend to stay close to the original value as it represents a good trade-off between performance and interpretability.

Finally, road safety experts confirm the relevance of these automatically discovered clusters. They also point out that the time saved can advantageously be spent on, for example, planning remedial actions.

3.3 Bayesian hierarchical generalized linear model

In our first proposal, the hierarchical structure discovered in the previous chapter is used to enhance the predictive performance and interpretability of generalized linear models (GLM). To account for such a structure, we use a Bayesian inferred hierarchical GLM. In section 3.3.1, we start by describing the multilevel structure of such models and then explain how Bayesian inference, with advances in probabilistic programming, infers efficiently the coefficients of these models (see section 3.3.2).

Moreover, as the standard formulation of GLM generally attaches a linear functional form to a parameter of the likelihood distribution, interactions between the explanatory variables are not taken into account. In section 3.3.3, we also describe how we mitigate this by first learning simple interactions from the analysis of the results of a special kind of polynomial neural network, and then by adding them to the linear functional form.

We finish this chapter by explaining how we evaluate and validate the models with a well-known technique, the Posterior Predictive Check.

3.3.1 Model description

To integrate the hierarchical structure, made of k clusters, into a GLM, we design the following multilevel model:
level 0 - model

$$
\begin{aligned}
Y_{ik} &\sim \mathcal{L}(Y_{ik}|\mu_{ik}, \Omega) \\
\mu_{ik} &= E[Y_{ik}|\mathbf{X}_{ik}] = g^{-1}(\eta_{ik}) \\
\eta_{ik} &= \beta_{0k} + \sum_{j=1}^{N} \beta_{jk} X_{ijk}
\end{aligned}
$$

level 1 - priors

$$\beta_{jk} \sim \mathcal{N}(\mu, \sigma)$$

level 2 - hyperpriors

$$\mu \sim \mathcal{N}(0, 100)$$
$$\sigma \sim \mathcal{HN}(100)$$

According to this model, output Y_{ik}, for observation i in cluster k, is generated from a likelihood distribution $\mathcal{L}$ parameterized with a set of specific parameters Ω and also μ_{ik}, the expected value of Y_{ik} conditioned on the observations. In section 2.2.1, we saw that, for crash prediction, $\mathcal{L}$ can be a $NegativeBinomial(Y_{ik}|\lambda, \alpha)$ where λ is the expected number of crashes and α controls the amount of allowed over-dispersion. Next, η_{ik} is a linear transformation of the explanatory variables. The inverse link function g^{-1} is necessary to map the domain of η_{ik} (*viz.*, the real line) to the one of μ_{ik}. For example, since the rate λ of a negative binomial must be positive, the exponential function is used for crash-count prediction. In case of binary classification, when the $Bernouilli(p)$ likelihood is used, p being a probability, g can be set to the logit function. Indeed, the logit maps a parameter constrained between 0 and 1 onto the real line (the inverse link function g^{-1} is, in that case, the logistic function). Finally, as explained in the previous section, the cluster-specific coefficients β_{jk}, for the N explanatory variables, depend on hyperpriors μ and σ, thus allowing an adaptive shrinkage to a mean common to all the observations.

3.3.2 Bayesian inference

Description

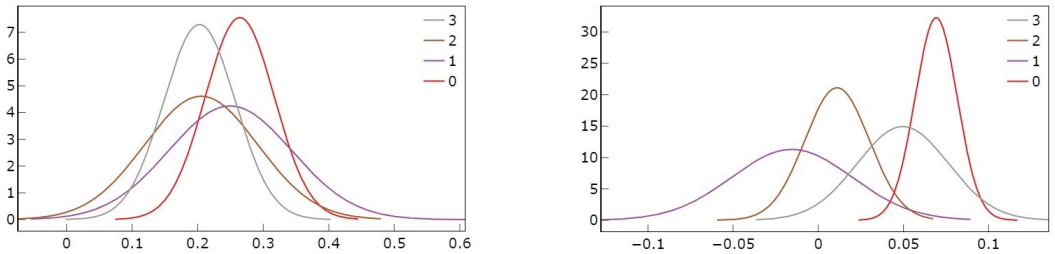
Bayesian Inference (BI) treats the parameters as random variable and associates a probability distribution to each parameter. Specifically, BI derives the posterior probability from the prior $p(\theta)$ and the likelihood $p(y \mid \theta)$:

$$p(\theta \mid y) \propto p(y \mid \theta)p(\theta) \tag{3.2}$$

with θ the set of parameters in the models.

Based on the knowledge of observed crash-count data, the parameters are updated according to eq. 3.2. However, the posterior distribution is usually not obtained in a closed-form distribution but by means of approximation techniques. In our work, we estimate the distribution with a sampling procedure based on a Markov Chain Monte Carlo (MCMC) algorithm. Despite the use of computationally expensive sampling algorithms, BI becomes more and more efficient thanks to the advances of probabilistic programming [117, 108].

Note that we could have used the frequentist alternative based on Maximum likelihood estimation (MLE). MLE use iterative algorithm, such as the Newton-Raphson



(a) log(AADT) (b) Percentage of heavy vehicles

Figure 3.11: Two examples of posterior distributions

method or gradient descent, to estimate coefficients β_j. MLE is closely connected to the Maximum A Posteriori (MAP) estimate, which is the mode of the posterior distribution after BI. Indeed, given a uniform prior, MLE and MAP estimates are identical. However, with MLE, each coefficient has a unique point estimate, thus the introduction of prior knowledge is not possible. Moreover, with BI, we also obtain uncertainty knowledge for each MAP estimate. For these reasons, we use BI to infer the parameters of our models.

Uncertainty knowledge

The analysis of the posterior distributions (see Fig. 3.11) allows one to measure the influence of each variable on the crash count. Due to the inherent partial pooling nature of the model, posterior distributions are dissimilar among clusters thus revealing various impacts of the same explanatory variable on the crash count. For instance, the posterior's mean related to the percentage of heavy vehicles in Fig. 3.11b is positive for cluster 0 and 3, but negative for cluster 1.

Moreover, we observe that the shapes of the posterior distributions are different from one cluster to another. This variability is linked to the size of the clusters: in general, the more samples, the more confidence in the estimates. For example, in Fig. 3.11a the posterior of the traffic related variable (log(AADT)) has a sharp probability distribution for cluster 0 and 3 but a flatter one for cluster 1 and 2. The latter highlights a greater uncertainty in estimating the coefficient associated with the traffic and calls for more vigilance when drawing conclusions for this risk factor.

3.3.3 Supervised learning of interactions with a polynomial network

Objectives

When first-order interactions between explanatory variables are integrated into a GLM, the relationship between a variable and the target may depend on the value of another variable. This can lessen the gap in predictive power between Bayesian inferred GLM and ML algorithms inherently able to capture complex nonlinear relationships. Moreover, these simple interactions are interpretable while the potentially highly entangled ones discovered by ML algorithms will often remain inaccessible to human understanding and increase the risk of overfitting.

Group Method of Data Handling algorithm

In our approach, important first-order interactions are discovered with a variant of the Group Method of Data Handling (GMDH) family of supervised algorithms [54]. This GMDH algorithm is a self-organized multi-layered structure of nodes (see Fig. 3.12). Each node generates its output z by applying a linear function with a covariation term to a pair of inputs (x_1, x_2) taken among either the nodes of the previous layer or the original explanatory variables:

$$z = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2$$

Let n be the number of nodes of the previous layer and m be the number of explanatory variables. To build the next layer, for each of the $\binom{n+m}{2}$ polynomials, the $a_0, ..., a_3$ parameters are set by minimizing through Ridge regression the least square error made by the polynomial when it approximates the target on a train dataset. Then, the fitted polynomials are evaluated on a validation dataset to select the top m constituting the new layer. When the score obtained on the validation dataset by the best node of the last layer added stops improving, the process terminates and the best node of the penultimate layer is the output of the network. Thus, this self-organized network discovers a polynomial that approximates the relationship observed on the training dataset between the explanatory variables and the target.

In our approach, we use this polynomial to select the most important first-order interactions between explanatory variables. If the coefficient of a term involving the product of two variables exceeds a given percentage δ of the magnitude of the biggest coefficient, δ being a hyper-parameter of our framework, we add to our Bayesian hierarchical model an interaction between these two variables.

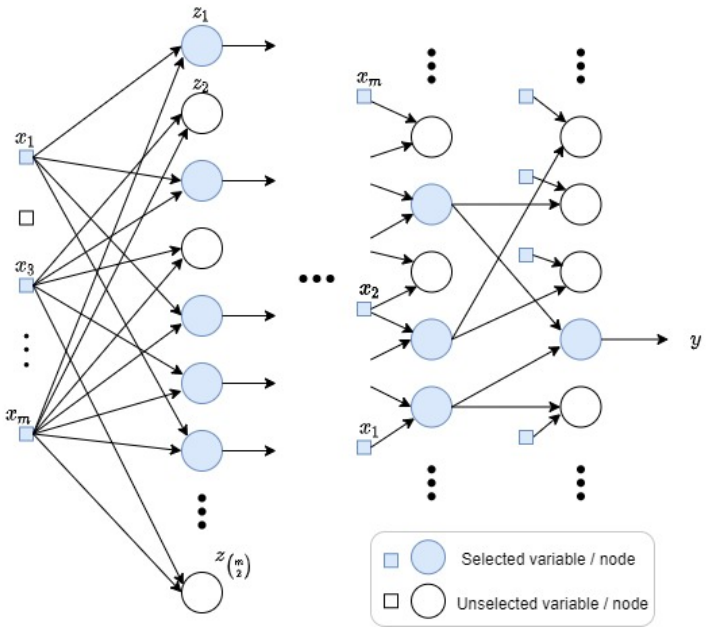


Figure 3.12: Structure of the GMDH model

Integration of interactions into the GLM

For a given observation i, let $\{\text{int}_{i1}, \text{int}_{i2}, ...\text{int}_{iM}\}$ be the set of first-order interactions selected by our GMDH-based methodology. To integrate them into the Bayesian hierarchical model, we modify the linear predictor η_{ik} (see section 3.3.1) such that:

$$\eta_{ik} = \beta_{0k} + \sum_{j=1}^{N} \beta_{jk} X_{ijk} + \sum_{m=1}^{M} \beta_{mk} \text{int}_{im}$$

Visualizing the effects of interactions

In our experiments, the GMDH polynomial highlights a major first-order interaction between speed limit and altitude. Triptych plots, introduced by Mc Elreath [83, p.234], are made to visualize such interactions. Thus, Fig. 3.13 depicts the bivariate relationship between annual average daily traffic (AADT) and predicted crash counts for cluster 1 from Fig. 3.5, depending on whether or not an interaction with the number of resting places is integrated into the Bayesian model. We observe that the slopes of the regression lines, constant and positive for both models, are steeper when considering an interaction. Thus, taking into account its interaction with the number of resting places, the positive influence of AADT on predicted crash counts gets more pronounced when the number of resting places increase.

(a) Without interaction effects

(b) With interaction effects

Figure 3.13: Triptych plots of predicted crash counts vs. annual average daily traffic. Note that explanatory variables are standardized (see section 4.2).

Implementation details

We trained the polynomial network with `GmdhPy`[5], an open source Python implementation of the GMDH algorithm. For fast computation time, we restrict the maximum number of layers to 5. The number of selected best neurons is equal to the number of original features, its default value. Once the model is trained, the underlying polynomial is obtained by recursive parsing of the network, from the final node to the input layer containing the explanatory variables. Major first-order interactions are identified when the absolute value of their coefficients are superior to a threshold value defined as $\delta \times max$, with max being the coefficient with the highest amplitude, and $\delta = 0.01$.

3.3.4 Model evaluation and validation

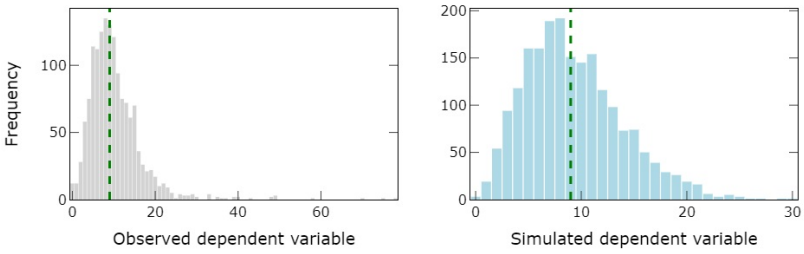
To use our model for point-estimate prediction, we must derive a Bayesian estimator from the posterior distributions. We use the mean of the posteriors which can be shown to minimize the mean squared error. In that way, we can compare our approach to black-box ML algorithms with standard quality metrics on a test dataset. Moreover, to check if the estimation of the posterior distributions converged well, we use the Posterior Predictive Check (PPC) graphical analysis method. Indeed, according to Gelman and Hill in [44, p. 158], PPC is a technique to "simulate replicated data under the fitted model and then compare these to the observed data". It allows one to look for systematic discrepancies between real and simulated data [43].

To illustrate this, we compare in Fig. 3.14, for two examples of clusters, the histogram of observed crashes with that of samples drawn from the posterior distribution of crash-counts. Dashed green lines indicate the means of, respectively, the observed and replicated data. The two distributions being similar, our model fits adequately the data. Thus, the integration of latent structure and interactions do not bring skewed prior knowledge that would disturb the inference.

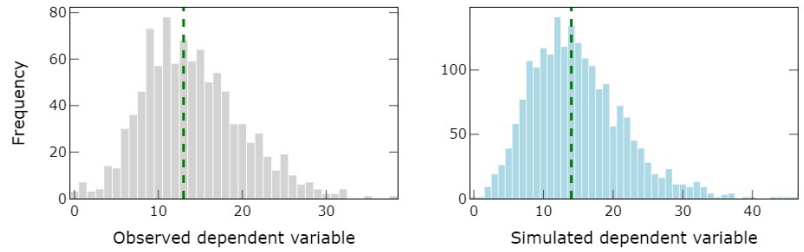
3.3.5 Conclusion

In this section, we have seen how a Bayesian learning of a hierarchical GLM can be enhanced by the introduction of a hierarchical structure and interactions identified by methods from the field of explainable artificial intelligence. In section 4.5, we report on results obtained on a 5-fold cross-validation, showing that these models improve the predictive performance of standard GLM (without multilevel formulation and interaction) and of the other simple interpretable models such as linear models or shallow decision trees.

[5] `https://github.com/kvoyager/GmdhPy`



(a) PPC on cluster 0



(b) PPC on cluster 2

Figure 3.14: Posterior Predictive Checks (PPC) on two clusters. Left column: observed data (cluster 0: 1491 samples; cluster 2: 965 samples). Right column: replicated data (2000 simulations)

However, this first proposal underlines several limitations. First, if one does not want the user to be solely responsible for deciding which functions of the original variables are to be considered, a strategy must be adopted to explore a combinatorial space of potential functional forms. Also, for the model to stay interpretable in presence of a large number of explanatory variables, the number of relevant coefficient shouldn't be too large or ones can suffer from the *information overload* phenomenom [3, 63, 99]. Thus, in our second proposal, we try to meet these challenges by computing a cluster-specific ranking of expansions of sparse regularized linear models ordered by increasing complexity thanks to symbolic regression.

3.4 Interpretable hierarchical symbolic regression

Throughout the previous sections, we described how a relevant hierarchical structure identified by analyzing the results of a post hoc explanation tool can enhance the predictive performance and interpretability of GLM by means of a multilevel formulation. In this section, we further exploit the discovery of this hierarchical structure by using symbolic regression to capture more complex but still interpretable relationships between the explanatory variables and the dependent variable.

First, we detail the related work of symbolic regression (SR) in section 3.4.1. Then, in section 3.4.2, we introduce SR with a focus on the technique we use to conduct the search in the space of mathematical expressions, *viz.*, a multi-objective optimization extension of the simulated annealing algorithm. Moreover, we present in section 3.4.3 how we successfully manage to build, through a partial pooling approach, a hierarchical symbolic regression to account for the structure previously discovered. Finally, in section 3.4.4, we explain how uncertainty knowledge is provided thanks to Bayesian Inference.

3.4.1 Related work

Symbolic regression consists in exploring a large space of functional forms to discover a predictive model with a good trade-off between accuracy and simplicity. Each element of this space is a parametric regression or classification model whose performance is measured (e.g., with cross-validation) on a given dataset after fitting its parameters. Both the parameters and the functional form of a predictive model are learned based on available data. A wide variety of approaches have been tried to effectively explore the space of functional forms.

Evolutionary algorithms and other optimization techniques

In genetic programming, population of a mathematical expressions evolves through selection, crossover and mutation to improve a fitness function [84, 11, 109, 46, 67]. Other SR are based on the metaheuristic algorithm of Pareto simulated annealing to discover a set of models which are optimal in terms of a balance of both accuracy and simplicity metrics [119]. Thanks to use of Meijer G-functions, SR can also be approached by algorithms based on gradient descent [6]. Bayesian processes, with algorithms based on the Markov Chain Monte Carlo strategy have been used as well to solve the SR problem [55].

Symbolic regression based on Machine Learning

Recent studies apply deep learning methods to symbolic regression in order to discover physical laws from experimental data. In [124], Udrescu *et al.* create a framework for symbolic regression based on neural network to discover hidden simplicity in the data (e.g., symmetry, separability) in order to decompose complex problems into simpler sub-problems. They apply their framework on 120 selected equations from classical mechanics, electromagnetism, quantum mechanics and obtain promising results on simulated data as they manage to discover more than 90% of the equations. Lately, this approach has been improved with the integration of an information complexity metric by means of Pareto optimization [123]. However, on a recent benchmark realised by La Cava *et al.* [67], this framework obtain poor results on real data and is even dominated by simple linear models.

Recently, Petersen *et al.* [97] use a hybrid approach that combines genetic algorithms and a recurrent neural network (RNN) trained by reinforcement learning to generate better symbolic models at each iteration. Finally, Valipour *et al.* [125] consider the problem as a sub task of language modelling and train a generative RNN model with reinforcement learning to produce symbolic equation skeletons whose constants are further adjusted by the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm [36].

Areas of application

SR finds applications in numerous domains such as physics [109], finance [23], climate modeling [118] or renewable energies [66]. So far, only few studies applied symbolic regression to safety analysis. Meier *et al.* [86] use prioritized grammar enumeration, a dynamic programming version of symbolic regression, to predict crash severity a few milliseconds before collision. Patelli *et al.* [94] design a GP-based symbolic regression to predict the traffic flow. To the best of our knowledge, symbolic regression has not been applied to long term crash predictions.

3.4.2 Symbolic regression with Pareto simulated annealing

General description

Most versions of symbolic regression (SR) discover an expansion of a linear model with the addition of non-linear effects by searching a space of functional forms. Section 3.4.1 gave an overview of the various methods that have been used to perform this search. Among them, we select simulated annealing, an effective metaheuristic known for its robustness in optimization problems involving a large search space [32, 26]. Thus, we represent the problem as a local search. Moreover, we adopt a multi-objective extension of the simulated annealing algorithm to perform the search while optimizing both the complexity and the accuracy of the models [119]. The search ends on a set of mutually non-dominated predictive models, the Pareto front.

Definition of a solution

The functional form of a model is extracted from a set of expression trees. Each expression tree is perfect, binary and consists of internal operator nodes and leaves. Leaves are either represented by a constant or an explanatory variable. Operator nodes can be unary (e.g., *cos*, *sin*, *tan*, *exp*, *ln*, *left*, *right*) or binary (e.g., $+, \times, /$) and have two children. For unary operators, we indicate with the subscripts " l " (for "left") and " r " (for "right") to which child the operation is applied. For instance, if the operator is ln_l, then the logarithm is applied to the left child. The *left* and *right* operators apply the identity function to the left and right child, respectively.

We extract the symbolic expression by a breadth-first traversal of the expression tree. In practice, as operators, constants and input variables are defined with `Sympy`, an open-source Python library for symbolic computation [87], the traversal returns a `Sympy` expression. Finally, the functional form $\mathscr{S}$ of a solution is obtained by the combination and algebraic simplification of the `Sympy` symbolic expressions of a set of expression trees (see Fig. 3.15).

Initialization of a first solution

Function `initialize` of algorithm 1 generates a first solution represented by a set $\mathscr{M}_{cur}$ of random expression trees, with $\mathscr{S}_{cur}$ the associated functional form. To this end, this function first creates balanced binary trees, each of the same depth. Then, for each tree, an inorder traversal associates an index to each node. Odd indices refer to internal nodes while even indices refer to the leaves (see Fig. 3.16a). In this way, we have an efficient means to search for a node in a tree and to know directly what type of node it is (see section Neighbourhood of a solution). At the same time, internal nodes are initialized with an operator chosen with equiprobability from the set of

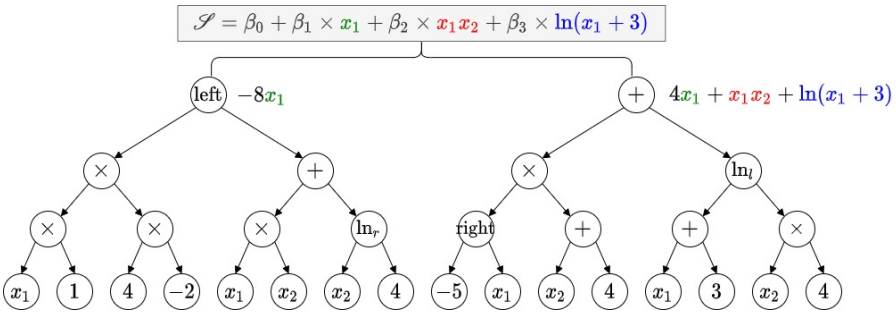


Figure 3.15: A functional form associated with a set of expression trees.

predefined operators introduced in section 3.4.2. Each leaf has a 50% probability of being initialized either to a constant or to one of the explanatory variables. In the latter case, each explanatory variable is equiprobable.

Neighbourhood of a solution

Function `generate` of algorithm 2 generates a new solution $\mathscr{S}_{new}$ in the neighbourhood of the current solution $\mathscr{S}_{cur}$. It randomly selects an expression tree from $\mathscr{M}_{cur}$ and a node index from $\{0, ..., 2^T - 2\}$, T being the tree depth. Then, a recursive search finds the node with the selected index. When the node is an operator (*viz.*, its index is odd), it is replaced by a randomly selected operator. Likewise, when a leaf is selected (*viz.*, its index is even), it is replaced by a randomly selected constant or explanatory variable. Fig. 3.16 illustrates this process.

If unchecked, function `generate` could lead to ill-defined operators. For instance, a logarithm could be applied to a potentially negative domain. Therefore, function `integrityCheck` infers recursively the domain of each operator node and, based on rules from interval arithmetic, checks its validity (table 3.1 introduces some of these rules). With interval arithmetic, we have an efficient way to ensure that the functional form generated from the random process does not contain any undefined values [62]. For more details on the integrity check, we refer to [119].

Cost of a solution

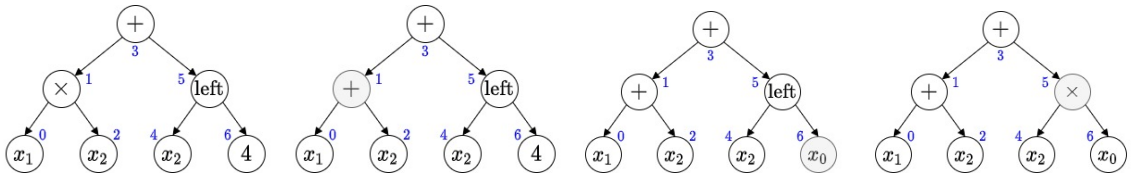
The cost of a solution is measured in terms of both the prediction error (see function `measurePerformance` of algorithm 2) and the complexity of the functional form (see function `measureComplexity` of algorithm 2).

Performance To obtain a robust estimate of the prediction error of $\mathscr{S}_{new}$, we compute the average RMSE, for a regression task, or $f1$-score, for a classification task, on

Algorithm 1 Symbolic regression with Pareto simulated annealing

Require: m: number of expression trees; $T_{min} = 0.0001$: initial temperature (heating phase) and minimum temperature (cooling phase); $\lambda_h = 1.15, \lambda_c = 0.85$: ratios between two adjacent temperatures in the heating phase and cooling phase, respectively; $s_h = s_c = 300$: number of iterations between two updates of temperature; $\gamma_c = 1.15$: ratio that controls the growth of s_c; max: maximum number of iterations during the cooling phase.

function SIMULATED ANNEALING($T_{min}, \lambda_h, s_h, \lambda_c, s_c, \gamma_c, max$)
  $T = T_{min}$ ▷ Annealing temperature
  $\zeta = \varnothing$ ▷ Pareto front
  $acc = 0, rej = 0$ ▷ Number of accepted and rejected solutions
  $\alpha = 0$ ▷ Acceptance rate
  $i = 0$
  $\mathscr{M}_{curr}, \mathscr{S}_{curr} \leftarrow$ `initialize` (m)
  while $\alpha \leq 0.9$ **do** ▷ Heating phase
    $i, \zeta, \mathscr{M}_{curr}, \mathscr{S}_{curr}, acc, rej \leftarrow$ `explore`$(T, i, \zeta, \mathscr{M}_{curr}, \mathscr{S}_{curr}, acc, rej)$
    if $i \bmod s_h = 0$ **then**
      $T \leftarrow T \times \lambda_h$, $\alpha \leftarrow acc/(acc + rej)$
      $acc \leftarrow 0, rej \leftarrow 0$
  $i = 0, \zeta = \varnothing$
  $\mathscr{M}_{curr}, \mathscr{S}_{curr} \leftarrow$ `initialize` (m)
  while $T > T_{min}$ and $i < max$ **do** ▷ Cooling phase
    $i, \zeta, \mathscr{M}_{curr}, \mathscr{S}_{curr}, acc, rej \leftarrow$ `explore`$(T, i, \zeta, \mathscr{M}_{curr}, \mathscr{S}_{curr}, acc, rej)$
    if $i \bmod s_c = 0$ **then**
      $T \leftarrow T \times \lambda_c$, $s_c \leftarrow s_c \times \gamma_c$
  return ζ



(a) $x_1x_2 + x_2$ (b) $x_1 + 2x_2$ (c) $x_1 + 2x_2$ (d) $x_1 + x_2 + x_0x_2$

Figure 3.16: A sequence of transformations applied to an expression tree. (a) An initial expression tree. Blue integers refer to node indices. (b) A transformation is applied to operator node 1, thus modifying the underlying functional form. (c) The transformation applied to leaf node 6 is muted due to its *left* operator parent. (d) Later in the process, node 6 can be reactivated when its parent is transformed.

Operation	Lower bound	Upper bound	Invalid if
$[a,b]+[c,d]$	$a+c$	$b+d$	
$[a,b]-[c,d]$	$a-d$	$b-c$	
$[a,b]\times[c,d]$	$\min\{ac,ad,bc,bd\}$	$\max\{ac,ad,bc,bd\}$	
$[a,b]/[c,d]$	$\min\{a/c,a/d,b/c,b/d\}$	$\max\{a/c,a/d,b/c,b/d\}$	$0\in[c,d]$
$\text{left}([a,b],[c,d])$	a	b	
$\ln_l([a,b],[c,d])$ [a]	$\ln(a)$	$\ln(b)$	$a\leq 0$

[a] $\ln_l$ designates a ln operator applied to the left child

Table 3.1: Rules for interval arithmetic, from [119, p.318]. We suppose that an operator node has two children. The left one is defined on $[a,b]$ and the right one on $[c,d]$.

Algorithm 2 Pseudo code of function `explore`

```
1: function explore(T, i, ζ, M_curr, S_curr, acc, rej)
2:     M_new ← generate(M_curr)
3:     if integrityCheck(M_new) then
4:         S_new ← simplify(M_new)
5:         if S_new ≠ S_curr then
6:             perf_new ← measurePerformance(S_new)
7:             compl_new ← measureComplexity (S_new)
8:             if accept(perf_new, compl_new, perf_curr, compl_curr, ζ, T) then
9:                 ζ, M_curr, S_curr, perf_curr, compl_curr ← update(
                       ζ, M_new, S_new, perf_new, compl_new)
10:                acc ← acc + 1
11:            else
12:                rej ← rej + 1
13:        else
14:            M_curr ← M_new
15:        i ← i + 1
16:    return i, ζ, M_curr, S_curr, acc, rej
```

the validation subsets of a 5-fold cross-validation process. On each training subset, the coefficients β_i of $\mathscr{S}_{new}$ are learned by solving an $l2$-regularized linear regression, for regression tasks, or an $l2$-regularized logistic regression, for classification tasks. In either case, the regularization parameter is determined on each training subset of the aforementioned 5-fold cross-validation either by an efficient generalized cross-validation [45] for ridge regression or by a nested cross-validation process for logistic regression. Once the estimate of the prediction error is obtained, the coefficients β_i are fitted one last time on the whole training dataset.

Complexity We improve the strategy introduced in [119] to propose a new measure of the complexity of a solution. We penalize both the collinearities and the number of terms present in the symbolic expression of the functional form. The complexity of a solution $\mathscr{S}$ composed of m terms is defined as:

$$\text{Complexity}(\mathscr{S}) = \sum_{i=1}^{m} \Big(1 + \max\big(\{|r_{ij}|; j \in \{1, 2, ..., m\} \setminus i\}\big)\Big) C_i \tag{3.3}$$

where r_{ij} is the Pearson's correlation coefficient, computed on the training dataset, between terms i and j, and C_i is the complexity of the term i. We use algebraic rules to compute the complexity of each term, some of which are presented in table 3.2. The complexity of a unary operator (e.g., the natural logarithm) is determined by approximating the operator, on its inferred domain, by a polynomial of increasing degree (at most 10) until the score of the fit, as measured on a validation set, is below a predefined threshold. The complexity of the unary operator is then defined as the degree of the best polynomial approximation. It should also be noted that, according to equation 3.3, the more terms a solution has, the more complex it is. We were able to confirm experimentally that the measured complexity represents well the complexity perceived by the safety experts.

Comparison of two solutions

The search ends with a set of Pareto optimal solutions that belong to the boundary beyond which neither the prediction error nor the complexity can be improved without deteriorating the other objective. This can be formally defined in terms of a dominance relation. Let U_1 be the prediction error and U_2 be the complexity metric.

$$\begin{aligned} &\mathscr{S}_a \text{ dominates } \mathscr{S}_b \\ \equiv \\ &\forall i \in \{1, 2\} \ : \ U_i(\mathscr{S}_a) \leq U_i(\mathscr{S}_b) \quad \text{and} \quad \exists j \in \{1, 2\} \text{ s.t. } U_j(\mathscr{S}_a) < U_j(\mathscr{S}_b) \end{aligned}$$

Thus, the search returns a set of non-dominated solutions called the Pareto front.

Term i	Complexity C_i	Example	Computed C_i
const	0	2	0
x	1	x_4	1
$f(x)^n$	$n \times C(f(x))$ [a]	x_2^2	2
$f(x) \times g(y)$	$C(f(x)) + C(g(y))$	$x_1 x_2^2$	3
$f(g(y))$	$C(f(x)) \times C(g(y))$	$\ln(3x_2^2)$	$C_{unary}(\ln) \times 2$ [b]

[a] $C(.)$ is the complexity of the inner function

[b] $C_{unary}(.)$ is the complexity of the unary operator

Table 3.2: Algebraic rules used to compute the complexity of each term, adapted from [119, p.320]

Exploration by Pareto simulated annealing

Simulated annealing (SA) is an iterative local search process used to solve optimization problems for which a simple hill-climbing approach would most often converge on a poor local optimum. At each iteration, SA generates randomly a solution $\mathscr{S}_{new}$ in the neighborhood of the current solution $\mathscr{S}_{cur}$. The probability P of accepting $\mathscr{S}_{new}$ as the new current solution is a function of both a temperature parameter T and the difference in cost ΔE between the two solutions.

$$P = e^{-\Delta E/T} \tag{3.4}$$

Annealing temperature T SA mimics the physical process of annealing in metallurgy where a material is first heated before being gradually cooled in order to reach an equilibrium state with increased ductility and hardness. SA follows a similar two-steps process.

The heating phase aims at discovering an initial temperature T_0 that favors exploration over exploitation in the beginning of the search. The heating process starts from a low temperature at which a deteriorating neighbour of the current solution is rarely accepted. Then, every s_h iterations, the temperature is increased according to a geometric series of ratio $\lambda_h > 1$. The process ends at a temperature T_0 at which at least 90% of the randomly generated neighbours are accepted.

During the cooling phase, the annealing temperature is progressively decreased, every s_c iterations, according to a geometric series of ratio $\lambda_c < 1$. High temperatures favor the exploration of the space of functional forms by preventing the process from converging too early on a local optimum (see Fig. 3.17). On the contrary, the more the temperature decreases, the less likely it is for a deteriorating neighbour to replace the current solution (see Fig. 3.17). The value of λ_c controls the speed at which the annealing temperature decreases. If λ_c is too small, the optimization may stay stuck

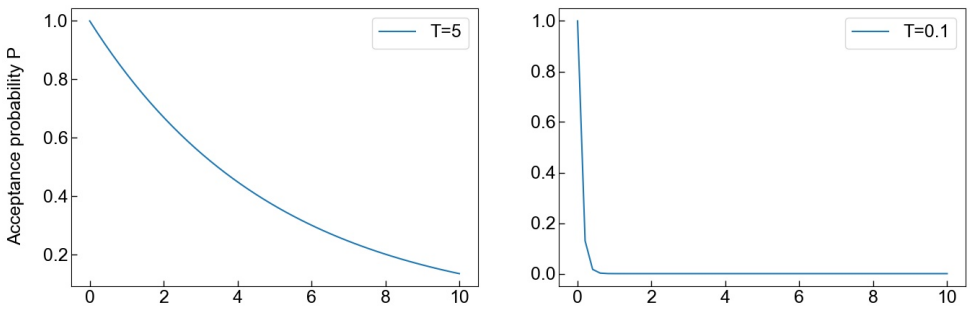


Figure 3.17: Effect of the annealing temperature on the acceptance probability.

too early in the neighborhood of a poor local optimum. Whereas, if λ_c is too close to 1, the optimization may take too long to reach a good optimum. Moreover, parameter s_c increases according to a geometric series of ratio $\gamma_c > 1$. Thus, more iterations are allocated to lower temperatures to favor the exploitation of promising functional forms. Finally, the search ends when either the temperature falls below a threshold or the number of iterations reaches a predefined maximum.

ΔE and the acceptance of a new solution For a single-objective optimization problem, ΔE is simply the difference of the objective function evaluated at two neighbouring solutions. For our multi-objective optimization problem, we use the dominance-based performance metric introduced above. When a new solution $\mathscr{S}_{new}$ dominates, or is as good as, $\mathscr{S}_{cur}$, it is accepted as the new solution (see function `accept` of algorithm 3). When $\mathscr{S}_{new}$ is less effective than $\mathscr{S}_{cur}$, it has a probability P defined by eq. 3.4 to be accepted. In that case, ΔE is defined as:

$$\Delta E(\mathscr{S}_{cur}, \mathscr{S}_{new}) = \frac{1}{|\tilde{\zeta}|}\left(|\tilde{\zeta}_{\mathscr{S}_{new}}| - |\tilde{\zeta}_{\mathscr{S}_{cur}}|\right) \tag{3.5}$$

with ζ the set of solutions that approximate the Pareto front, $|\tilde{\zeta}|$ the cardinality of $\zeta \cup \{\mathscr{S}_{cur}, \mathscr{S}_{new}\}$, and $|\tilde{\zeta}_{\mathscr{S}}|$ the number of solutions in $|\tilde{\zeta}|$ that dominate $\mathscr{S}$ (see Fig. 3.18). Moreover, to smooth the estimated acceptance probability distribution, new artificial points are added to the attainment surface to get an evenly spread attainment surface over the two dimensions of the Pareto front [115].

Updates of the Pareto front Finally, when $\mathscr{S}_{new}$ is accepted, the Pareto front ζ is updated (see function `update` of algorithm 3) by removing the solutions dominated by $\mathscr{S}_{new}$ and then adding $\mathscr{S}_{new}$ to ζ when it is not dominated by any other solution

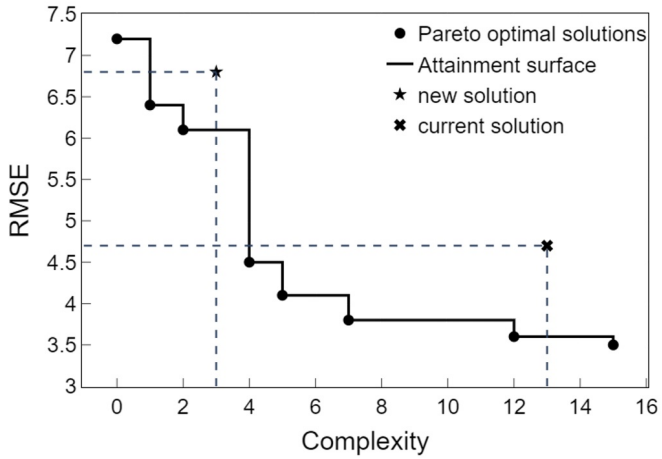


Figure 3.18: Example of an approximated Pareto front and its attainment surface, adapted from [119, p. 322]. From Eq. 3.5, $\Delta E(\mathscr{S}_{cur}, \mathscr{S}_{new}) = (2-4)/9 = -2/9$

in ζ. Thus, at the end of each iteration, ζ is the set of non-dominated solutions encountered during the search.

3.4.3 Partial pooling approach for symbolic regression

To handle correlations among groups of observations in the dataset, we developed a two-stage partial pooling approach to build hierarchical symbolic regressions. First, we start by running a symbolic regression on the whole dataset to automatically discover global models. Then, for each cluster in the hierarchical structure discovered in section 3.2, we run a new symbolic regression to extract cluster-specific functional forms that are merged afterwards to the functional form of a global model previously selected.

Automatic discovery of global models

In section 4.6, where we illustrate the dynamic interpretative process made possible by our framework, we emphasize the interest of being able to let the user choose a predictive model on the Pareto front. In that way, the end-user can precisely balance between the predictive performance and the simplicity of the model. However, in our proposed methodology, we also need a principled way to automatically select a model on the Pareto front. To do this, first, we consider the point Ω in the Pareto plan that, (i) on the performance axis, is at the level of the most efficient model encountered and, (ii) on the complexity axis, is at the level of the simplest model encountered. Then, we select the model $\mathscr{S}_{glob}$ on the Pareto front closest to Ω in the sense of the

Algorithm 3 Pseudocode of `accept` and `update` functions

function `accept`($\text{perf}_{new}, \text{compl}_{new}, \text{perf}_{curr}, \text{compl}_{curr}, \zeta, T$)
    $is\_accepted \leftarrow$ `False`
    if $\text{perf}_{new} \leq \text{perf}_{curr}$ and $\text{compl}_{new} \leq \text{compl}_{curr}$ **then**
        $is\_accepted \leftarrow$ `True` ▷ new solution dominates, or is as good as, the current one
    else
        compute P according to Eq. 3.4
        draw randomly j in $[0, 1]$
        if $P \geq j$ **then**
            $is\_accepted \leftarrow$ `True`
    return $is\_accepted$

function `update`($\zeta, \mathscr{M}_{new}, \mathscr{S}_{new}, \text{perf}_{new}, \text{compl}_{new}$)
    $is\_dominated =$ `False`
    for solution $\mathscr{S}$ in the Pareto front ζ **do**
        if $\mathscr{S}$ dominates $\mathscr{S}_{new}$ **then**
            $is\_dominated =$ `True`
    if $is\_dominated =$ `False` **then**
        remove solutions in ζ dominated by $\mathscr{S}_{new}$
        add $\mathscr{S}_{new}$ to ζ
    return $\zeta, \mathscr{M}_{new}, \mathscr{S}_{new}, \text{perf}_{new}, \text{compl}_{new}$

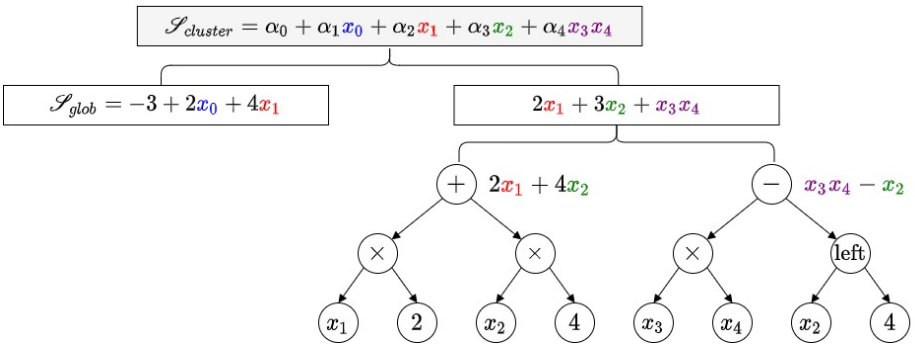


Figure 3.19: Extraction of a cluster-specific functional form from a set of expression trees and the fixed functional form of the global model

Euclidean distance. This model, located in the elbow of the Pareto front, is likely to offer a good trade-off between predictive performance and complexity. In the next stage of our approach, it is used as a starting point to build cluster-specific models.

Cluster-specific models

To discover cluster-specific phenomena, for each cluster discovered by the approach introduced in section 3.2, a modified version of the symbolic regression search is conducted. It consists in merging the functional form built from the expression trees with the fixed functional form of $\mathscr{S}_{glob}$ (see Fig. 3.19): common terms are grouped together and new terms are added to the formula. Hence, the marginal effects already represented by $\mathscr{S}_{glob}$ can be reduced or amplified and new cluster-specific effects can be discovered. It corresponds to a partial pooling approach where cluster-specific models can benefit from the effects already discovered by the global model.

In classification tasks, the dependent variable can be highly imbalanced on some clusters only. For imbalanced clusters, we apply jointly the synthetic minority over-sampling technique (SMOTE) [22] and edited nearest neighbor (ENN) [131]. The former generates samples from the minority class with interpolation. The later applies under-sampling to clean the noisy samples generated with SMOTE. Also, it should be noted that in the case of a homogeneous cluster, a cluster-specific model would not make sense and we propose to stay with the global model $\mathscr{S}_{glob}$.

3.4.4 Uncertainty estimation

Our approach results in global and cluster-specific expansions of linear models. Therefore, the marginal effects of the terms composing the models are readily interpretable. However, since the training set has been used to estimate the $l2$-regularization hyper-

parameters, there is no simple linear relationship between uncertainty in the parameters and uncertainty in the target.

Bootstrap and asymptotic statistics

Bootstrap techniques could estimate the uncertainty in the parameters. Still, a standard bootstrap approach is not appropriate since the bias introduced by the penalty term would not be correctly estimated. Double bootstrap techniques have been proposed [128, 82] to take into account an estimation of the bias. Nonetheless, they are computationally expensive ($O(n^3)$ where n is the number of samples). Also, asymptotic statistics have been derived to measure the uncertainty in the parameters under a fixed setting of the regularization parameter [35]. They wouldn't be appropriate in our case since we estimate the regularization parameter by leave-one-out cross-validation.

Bayesian interpretation of Ridge regression

In our work, we make use of a well-known equivalence [85] between the ridge regression regularization parameter and the parameters of a Gaussian prior for the Bayesian formulation of linear regression. It can be shown that the variance of the zero-centered Gaussian prior τ^2 must be defined as:

$$\tau^2 \equiv \frac{\sigma^2}{\lambda}$$

where λ is the ridge regularization parameter and σ^2 is the variance of the likelihood that can be estimated by measuring the variance of the target on the training dataset.

More specifically, in the frequentist approach, the coefficient estimate $\hat{\beta}_{Ridge}$ of Ridge regressions is solved from:

$$\hat{\beta}_{Ridge} = \arg\min[(Y - \mathbf{X}\beta)^\top (Y - \mathbf{X}\beta) + \lambda \beta^\top \beta] \tag{3.6}$$

In the Bayesian paradigm, no single coefficient estimate is found but a posterior distribution of β is inferred from the data. From the Bayes theorem, this posterior distribution can be written as:

$$p(\beta|Y, \mathbf{X}) \propto p(\beta) \cdot p(Y|\mathbf{X}, \beta) \tag{3.7}$$

where $Y \,|\, \mathbf{X}, \beta \sim \mathcal{N}(\mathbf{X}\beta, \sigma^2 I)$ with $\sigma > 0$ is the Bayesian formulation of linear regression. If we suppose zero-centered gaussian priors with variance τ^2, then Eq. 3.7 becomes:

$$\begin{aligned} p(\beta|Y, \mathbf{X}) &\propto \exp\Big[-\frac{1}{2}(\beta - 0)^\top \frac{1}{\tau^2} I (\beta - 0)\Big] \cdot \exp\Big[-\frac{1}{2}(Y - \mathbf{X}\beta)^\top \frac{1}{\sigma^2}(Y - \mathbf{X}\beta)\Big] \\ &= \exp\Big[-\frac{1}{2\sigma^2}(Y - \mathbf{X}\beta)^\top (Y - \mathbf{X}\beta) - \frac{1}{2\tau^2}\beta^\top \beta\Big] \end{aligned}$$

From this expression, we can compute the Maximum A Posteriori estimate $\hat{\beta}_{\text{MAP}}$:

$$\begin{aligned}\hat{\beta}_{\text{MAP}} &= \arg\max\ \exp\Big[-\frac{1}{2\sigma^2}(Y-\mathbf{X}\beta)^\top(Y-\mathbf{X}\beta)-\frac{1}{2\tau^2}\beta^\top\beta\Big]\\ &= \arg\min\ \frac{1}{\sigma^2}(Y-\mathbf{X}\beta)^\top(Y-\mathbf{X}\beta)+\frac{1}{\tau^2}\beta^\top\beta\\ &= \arg\min\ (Y-\mathbf{X}\beta)^\top(Y-\mathbf{X}\beta)+\frac{\sigma^2}{\tau^2}\beta^\top\beta\end{aligned}$$

which is equivalent to the Ridge regression estimate found in Eq. 3.6 when $\lambda \equiv \frac{\sigma^2}{\tau^2}$.

Thus, for each Pareto optimal solution, we start from the discovered functional form and the value of the ridge regularization hyper-parameter λ to infer again the coefficients, but this time, using Bayesian inference with the above prior. The resulting posterior distributions give an estimate of the parameters' uncertainty.

3.4.5 Implementation details

We implement our model in `Python`. In order to converge towards interpretable models, we restrict the operators available to the symbolic regression to $\{left, right, ln\}$ for the unary ones, and $\{\times, +, -\}$ for the binary ones. For the algebraic simplification of the expression trees by, e.g., grouping common terms together (see function `simplify` in algorithm 2), we use a module[6] from the `Sympy` library ([87]).

To fit the coefficients of a newly discovered functional form, we use the `scikit-learn`[7] [96] implementations of ridge regression (in the case of a regression task) and $l2$-regularized logistic regression (in the case of a classification task). The optimal coefficients of the linear models are computed with regularized least square algorithms. Indeed, with the introduction of a weight decay, better generalization performances can usually be achieved and the models are less prone to the negative effects of multicollinearities. We optimize the $l2$-regularization parameter by either an efficient form of leave-one-out cross-validation (*viz.*, generalized cross validation) for regression tasks or by five-folds cross-validation for classification tasks. However, for even moderately large classification datasets (e.g., *Adult* in table 4.2), performing a k-fold cross-validation at each iteration of the symbolic regression can take a long time. In that case, we perform the search using the `scikit-learn` implementation of a ridge classifier to benefit from the efficient computation of the generalized cross validation. Only when a model is added to the Pareto front, do we fit its parameters by $l2$-regularized logistic regression with k-fold cross-validation to optimize the regularization parameter. Indeed, we observed experimentally that an $l2$-regularized

[6] `https://docs.sympy.org/latest/modules/simplify/simplify.html`

[7] `https://scikit-learn.org/stable/`

logistic regression model tends to perform slightly better than a ridge classifier. By adopting this strategy, we reduce the computation time by a factor of 6 on the *Adult* dataset, while the final Pareto front remains nearly identical.

For imbalanced classification tasks, we use the `imbalanced-learn` implementation[8] of the SMOTE-ENN resampling technique [22, 131].

As explained in section 3.4.4, in order to endow our final models with uncertainty estimates, we use Bayesian inference to compute the posteriors for the coefficients of each functional form on the Pareto front. We apply gaussian priors corresponding to the already known optimal value of the regularization hyper-parameter (see section 3.4.4). We rely on the `pymc3`[9] library with the *No U-Turn Sampler* ([51]) to run simultaneously two Markov chains for 3000 iterations, with a burn-in period of 1000 iterations.

[8] `https://imbalanced-learn.org/stable/references/generated/imblearn.combine.SMOTEENN.html`

[9] `https://docs.pymc.io/en/v3/`

Chapter 4

Experiments

In the thesis, we design intrinsically interpretable predictive models to improve high-stakes decision-making processes. In our first contribution, we propose a methodology that combines Bayesian learning of hierarchical GLM with automatic detection of a hierarchical structure and interactions through methods borrowed from the field of explainable artificial intelligence (XAI). In our second contribution, we develop a novel methodology to exploit even better the hierarchical structure by combining symbolic regression and multi-objective optimization to obtain models that are sparser, more efficient while capturing more relevant interactions.

To validate that our approaches improve the predictive capacities of interpretable models and get close to the black-box models, our experiments are carried out on the highway dataset and on more than ten public datasets covering different tasks from various domains.

In this chapter, we begin by presenting in section 4.1 the different datasets on which the experiments are carried out and we introduce the different data processing in section 4.2. Then, we introduce in section 4.3 the evaluation metrics for regressions and classifications. In section 4.4, we describe all the parametric and non-parametric models that are used for comparison purposes. Then, we share the results in section 4.5. Finally, in section 4.6, we illustrate the benefits of our approach by introducing, on a realistic case study, an application we designed for highway safety experts

4.1 Description of the datasets

4.1.1 French highway dataset

To address the problem of crash prediction, we generate a tabular dataset from the different available data sources. Most of the data come from the *APRR*'s relational databases, and the remaining data, in particular topographical surveys of the network infrastructure, was sent to us by field experts in the form of structured flat files. To generate the dataset, we first divide the highway network into segments of equal length, then for each segment, we calculate the different variables on a given time period. In this section, we describe in more detail the process that leads to the generation of this dataset, further referred to as the *French Highway* dataset.

Highway network meshing

In the first place, the network must be divided into segments in order to be able to identify afterwards more or less crash-prone configurations on the network and to understand the associated influential explanatory variables. For this, we rely on spatial surveys of reference points of each highway belonging to the network. Each highway is divided into one-way segments of equal length. In this study, we select a length equal to 10 kilometers, in order to be able to capture the variations of the explanatory variables on the vast network while having a crash count rarely equal to zero. Of course, at the ends of the network, some segments will have a length less than the predefined one. Thus, we decide not to consider them in the analysis.

Calculation of variables

Considering the data available to us, we chose to restrict the analysis to the APRR network only. The study therefore covers 1,894 km of network, since the 429 km of the AREA subsidiary's highway network (see Fig. 1.1) have been withdrawn. The data covers eleven years, from January 1, 2008 to December 31, 2018. In our work, we focus on predicting the annual crash count. The available temporal explanatory variables are therefore also aggregated to this time scale. The spatial variables remain fixed over all the years of the study. In this section, we describe how we calculate the different variables, starting with the accident count. Descriptive statistics of the computed variables are given in table 4.1.

Crash count The accident data comes from police reports or descriptive sheets filled out by APRR staff present at the crash site. The information transmitted are very detailed: location, date and time, number of individuals involved, type of vehicle, etc. About 20% of accidents have no time indication because the protagonists did not

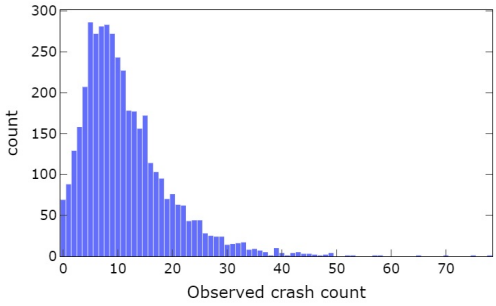


Figure 4.1: Distribution of the observed crash counts

wait for the arrival of help, police or group staff before leaving. However, they are identified by the patrols on the same day and are therefore also taken into account.

Then, accident data are stored in the company's large databases. It is from these databases that we are able to retrieve all crash-related information needed for the study. For each highway segment, the accidents are aggregated over each year of the study. The number of crashes per segment varies greatly (see Figure 4.1 for the distribution of observed crash counts). In table 4.1, we observe that the variance of the crash count exceeds its mean. This phenomenon, called over-dispersion, is often witnessed with crash data [75].

Traffic related variables Two traffic-related variables are considered in the study: the annual average daily traffic (AADT) and the percentage of heavy vehicles. Two data sources are available to estimate these variables. One, coming from the 797 counting loops integrated into the road pavement, represents the number of vehicles and the percentage of heavy vehicles having passed through the sensors during a fixed period of time (6 min). The other comes from transactions at toll gates, taking into account the class of vehicle (e.g., light vehicle, two-axled truck, etc.). While the former often has noisy data (e.g., outliers, long absence of data), mainly due to sensor failures, the latter provides a very reliable estimate of the AADT and percentage of heavy vehicles on highway segments. We therefore select toll data as a way to estimate the yearly traffic related variables on the network. We use the weighted mean to evaluate the traffic on our segments and to be able to capture traffic variations inside the segments:

$$\overline{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{4.1}$$

with w_i being the distance for which the variable has the value x_i.

Name	type	Min	Max	Mean	Std.
Dependent variable					
Crash count	discrete	0	78	11.43	8.06
Explanatory variables					
Temporal					
AADT	continuous	659	37644	12828	6791
Percentage of heavy vehicles	continuous	1	35.1	17.39	5.80
Spatial					
Speed limit	continuous	104	130	128.37	4.91
Number of interchanges	discrete	0	7	0.2	0.68
Number of resting places	discrete	0	3	0.63	0.57
Right shoulder width	continuous	0.6	3	2.93	0.25
Altitude	continuous	71.97	615.8	250.83	105.71
Presence of ramps	binary	0	1	0.24	0.43
Presence of tunnels	binary	0	1	0.02	0.13
Presence of tollgates	binary	0	1	0.43	0.5
Presence of bridges	binary	0	1	0.18	0.39
Number of instances in the dataset: 4152					

Table 4.1: Description of the *French Highway* dataset

Spatial variables A list of spatial variables is given in table 4.1. For the variables representing specific elements such as interchanges, tunnels, etc., we received from field experts a spatial statement of their coordinates. Variables that have very small variation on the network (*viz.*, ramps, tunnels, toll barrier, structures) are binarized to indicate whether an element is present on a roadway segment or not. The others (*viz.*, interchanges, resting places) are discrete variables referring to as a count of the element of interest on the roadway segment. The remaining variables (*viz.*, right shoulder width, altitude and speed limit), representing continuous elements on the network, are calculated from the weighted mean defined in eq. 4.1.

4.1.2 Other datasets

As said before, we extend the study to 13 other datasets covering regression and classification tasks from various domains. The size of these datasets varies greatly, both in terms of volumes (from 546 to 116640 instances) and dimensionalities (from 7 to 117 features). A general description of the datasets is given in table 4.2.

Dataset	#Instances	#Features	dependent variable	Pos[a]
Regression				
French Highway	4152	11	crash count	-
Insurance [69]	1338	7	health insurance costs	-
Airbnb [5]	48895	12	housing prices	-
House (218_house_8L)*	22784	8	-	-
Puma (225_puma8NH)*	8192	8	-	-
Satellite (294_satellite_image)*	6435	36	-	-
Wind (503_wind)*	6574	14	-	-
Breast tumor (1201_BNG_breastTumor)*	116640	9	-	-
Music (4544_GeographicalOriginalofMusic)*	1059	117	-	-
Wine†	4898	12	white wine quality	-
Toxicity†	546	9	aquatic toxicity	-
Gas†	36733	11	gas emission	-
Classification				
Breastcancer†	569	30	malignant tumor	0.06
Adult†	48842	14	earnings $> \$50K$/year	0.27

[a] percentage of positive cases

* Datasets taken from `https://epistasislab.github.io/pmlb/index.html`

† Datasets taken from `https://archive.ics.uci.edu/ml/index.php`

Table 4.2: Datasets

4.2 Data preparation

For all datasets, we do not apply any process of dimensionality reduction such as feature selection or principal component analysis. The original data are transformed in different ways depending on the ML model. Models that introduce regularization terms have standardized data as input. Standardization makes the values of each continuous explanatory variable have zero-mean and a unit-variance:

$$x' = \frac{x - \mu}{\sigma}$$

where μ is the mean value of the variable, and σ its standard deviation.

For neural networks, continuous explanatory variables are re-scaled in the $[0, 1]$ interval:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Moreover, categorical explanatory variables are one-hot encoded. More precisely, if n is the number of different categorical values, then one-hot encoding transforms the variable into $n - 1$ binary explanatory variables describing the values of the original explanatory variable.

		Predicted class	
		Positive	Negative
Actual class	Positive	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True negative (TN)

Table 4.3: Confusion matrix

4.3 Performance metrics

4.3.1 Regression

To measure the performance of predictive models on regression tasks, we compute the root mean square error and the mean absolute deviation. Given n the number of observations, y_i the target and $\hat{y}_i$ the predicted value:

Root mean square error is a measure of how spread out are the prediction errors, in the context of numerical predictions. This metric is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

Mean Absolute Deviation is a measure of variability that indicates the average distance between the predictions and the average value of the observed variable:

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - \bar{y}|$$

with $\bar{y}$ being the mean value of y.

4.3.2 Classification

To evaluate our models on binary classification tasks, we use metrics derived from the confusion matrix presented in table 4.3. Each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class. However, many machine learning and statistical models predict a probability. To know the associated predicted class, a classification threshold is used, most of the time equal to 0.5. Probability values greater than this threshold are mapped to one class, and the remaining are mapped to another.

From the confusion matrix in table 4.3, we can compute:

accuracy is the proportion of correct results that a classifier achieved:

$$acc = \frac{TP + TN}{TP + TN + FP + FN}$$

This very intuitive metric loses interest when data are imbalanced, as it can give an over-optimistic measure of the model's predictive performance. For instance, on the *Breastcancer* dataset where the percentage of negative cases (*viz.*, *the tumor is benign*) is 0.94 (see table 4.2), a model that has learned to detect only benign tumors will obtain a high accuracy. However, measuring the model's ability to predict malignancies can be equally useful, depending on the context. Thus, other metrics must be considered.

precision is the number of correctly predicted positive observations over the total number of predicted positive observations:

$$pre = \frac{TP}{TP + FP}$$

recall is the number of correctly predicted positive observations over the total number of observations in the actual class:

$$rec = \frac{TP}{TP + FN}$$

$f1$**-score** is the harmonic mean of the precision and recall:

$$f1 = 2 * \frac{pre * rec}{pre + rec}$$

Both the false positives and false negatives are considered which makes this metric very useful, especially when the data are imbalanced.

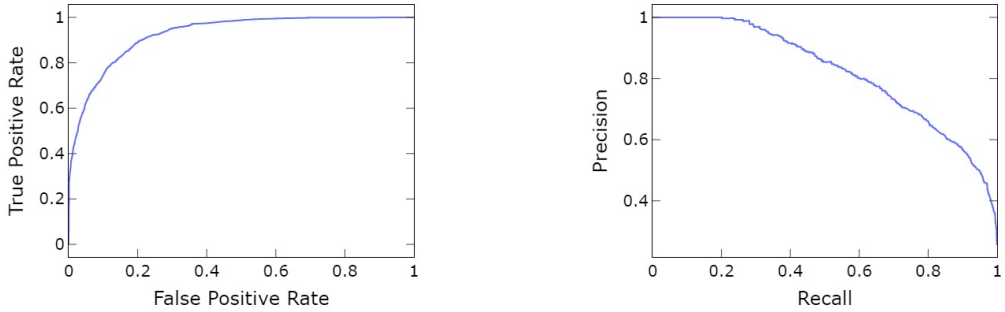
Receiving Operator Characteristic (ROC) curve is designed by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold values (see Fig. 4.2a). TPR indicates how many positive predicted outcomes occur among all positive samples:

$$TPR = \frac{TP}{TP + FN}$$

FPR defines how many incorrect positive outcomes occur among all negative samples:

$$FPR = \frac{FP}{FP + TN}$$

ROC curve is appropriate when the observations are well balanced between each class. However, it can give an overly optimistic view of the model's performance if datasets are very imbalanced.



(a) (b)

Figure 4.2: Examples of (a) ROC curve and (b) precision-recall curve obtained on the *FrenchHighway* dataset

Precision-recall curve illustrates the trade-off between precision and recall for various threshold values (see Fig. 4.2b). This curve is tailored for problems where classes are very imbalanced, but also when the positive class is more interesting than the negative one.

The two aforementioned parametric curves are not performance measures by themselves but representations of how well classifiers can discriminate between the two classes for different threshold settings. Thus, to obtain a measure of performance across all possible classification thresholds, we compute the area under the curve. These metrics are further referred to as $AUROC$ and $AUPR$ for the ROC curve and the precision-recall curve, respectively.

4.4 Models used for comparison

4.4.1 Parametric interpretable models

Among the most simple and interpretable models, we select the `scikit-learn` implementations of ordinary least square regression (OLS), logistic regression (LR) and *decision trees* of depth no more than 5 (to preserve interpretability). We also consider a standard Bayesian inferred GLM ($B\text{-}GLM$) implemented with the `Pymc3` library. This model does not include the prior knowledge of a hierarchical structure nor interactions between variables. A variant with interactions discovered by the polynomial network presented in section 3.3.3 is also considered and is further referred to as *B-GLM-int*. On the *French Highway* dataset, we also train a generalized linear mixed

model (*GLMM*) implemented with the `statsmodels`[1] library.

Moreover, we include two variants of Generalized Additive Models (GAM). For the first one, *GAM-splines*, based on a spline basis, we use the `PyGAM`[2] implementation. For the second one, explainable boosting machine (*EBM*), based on gradient boosting with bagging, we use the implementation provided by the `InterpretML` framework [92].

We also compare our approach to genetic programming based symbolic regressions with the reference implementations of the `gplearn`[3] package (*SR-GP*) and GP-GOMEA[4] (*SR-Gomea*) [129], the latter being known to perform well on many real world datasets [67]. For both implementations, the set of operators is restricted to the one we use in our approach (*viz.*, $\{+, -, \times, ln\}$). We also consider *SR-Gomea-op*, the same model with a less restricted set of operators (*viz.*, $\{+, -, \times, ln, cos, sin, \sqrt{}\}$), the same as the one used by [67] in their recent survey. Note that GP-GOMEA does not currently have an implementation for classification problems.

For all the aforementioned interpretable models, we apply a no pooling approach that accounts for the clusters discovered by our *Hierarchical structure* module (see section 3.2). This approach fits a separate model for each cluster and considers that no similarities exist between them.

4.4.2 Non-parametric black-box models

We select three highly flexible black-box models: (i) the `scikit-learn` implementation of Support Vector Machines (SVM) and (ii) Multilayer Perceptrons (MLP), and (iii) the `XGBoost` ([24]) gradient tree boosting library[5]

4.4.3 Models from our proposal

We consider the first proposal of ours namely, the Bayesian hierarchical GLM (*BH-GLM-int*). Recall that the latter is based on the bayesian inference of a linear hierarchical model with data-driven discovery of objective priors in the form of i) a hierarchical structure (chapter 3.2) and ii) strong first-order interactions obtained through the analysis of the structure of a trained self-adaptive polynomial network (chapter 3.3). We also include the variant without interactions between variables (*BH-GLM*). For each dataset, the associated parameters and priors are given in table 4.4.

[1] `https://www.statsmodels.org/stable/mixed_glm.html`
[2] `https://pygam.readthedocs.io/en/latest/`
[3] `https://gplearn.readthedocs.io/en/stable/`
[4] `https://github.com/marcovirgolin/GP-GOMEA`
[5] `https://xgboost.readthedocs.io/en/latest/python/python_intro.html`

Dataset	Link[a]	#clusters	1st level	2nd level	3rd level
Regression					
FrenchHighway	Log	4	$Y_{ik} \sim \text{NB}(\mu_{ik}, \alpha)^b$, $\alpha \sim G(a = 0.5, b = 2.5)^c$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(5)$
Insurance	Id[d]	2	$Y_{ik} \sim \mathcal{N}(\mu, 1000)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 100)$, $\sigma \sim \mathcal{H}(5)$
Airbnb	Id	4	$Y_{ik} \sim \mathcal{N}(\mu, 5)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
House	Id	4	$Y_{ik} \sim \mathcal{N}(\mu, 1000)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 100)$, $\sigma \sim \mathcal{H}(5)$
Puma	Id	3	$Y_{ik} \sim \mathcal{N}(\mu, 10)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
Satellite	Id	3	$Y_{ik} \sim \mathcal{N}(\mu, 20)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 50)$, $\sigma \sim \mathcal{H}(5)$
Wind	Id	3	$Y_{ik} \sim \mathcal{N}(\mu, 10)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
Breast tumor	Id	2	$Y_{ik} \sim \mathcal{N}(\mu, 20)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 50)$, $\sigma \sim \mathcal{H}(5)$
Music	Id	3	$Y_{ik} \sim \mathcal{N}(\mu, 10)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
Wine	Id	3	$Y_{ik} \sim \mathcal{N}(\mu, 10)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
Toxicity	Id	2	$Y_{ik} \sim \mathcal{N}(\mu, 5)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 10)$, $\sigma \sim \mathcal{H}(2.5)$
Gas	Id	2	$Y_{ik} \sim \mathcal{N}(\mu, 20)$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 50)$, $\sigma \sim \mathcal{H}(5)$
Classification					
Breastcancer	Logit	2	$Y_{ik} \sim \mathcal{B}(p = \frac{\exp(\mu_{ik})}{1+\exp(\mu_{ik})})^e$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 100)$, $\sigma \sim \mathcal{H}(5)$
Adult	Logit	4	$Y_{ik} \sim \mathcal{B}(p = \frac{\exp(\mu_{ik})}{1+\exp(\mu_{ik})})$	$\beta_j \sim \mathcal{N}(\mu, \sigma)$	$\mu \sim \mathcal{N}(0, 100)$, $\sigma \sim \mathcal{H}(5)$

[a]Link function; [b]Negative Binomial distribution; [c]Gamma distribution; [d]Identity function; [e]Bernoulli distribution

Table 4.4: Parameters and priors for the Bayesian hierarchical models

Moreover, we consider several variants of our second proposal. *SR-trad* and *SR-max* use only the global model of section 3.4.3 while *HSR-trad* and *HSR-max* use the cluster-specific models of section 3.4.3 (prefix "H" stands for *hierarchical*). We also train our hierarchical symbolic regression on clusters discovered with a hierarchical agglomerating clustering applied to the training data, including the dependent variable. These models are further referred to as *HSR-naive-trad* and *HSR-naive-max*. Finally, *SR-NP-trad* and *SR-NP-max* are cluster-specific models learned with a no pooling approach, meaning that they do not include the knowledge of the global model. The suffixes *trad* and *max* are used to distinguish models selected near the elbow of the Pareto front, that should have a good trade-off (whence *trad*) between complexity and predictive performance, from models of maximum complexity (whence *max*).

4.4.4 Hyper-parameters tuning

For fair comparisons, the hyper-parameters of the models presented in section 4.4.1 and section 4.4.2 are optimized by cross-validation with grid-search. For each model, the grid of hyper-parameters' values are given in table 4.5.

Moreover, we conducted a grid-search for the SR's hyper-parameters (*viz.*, number of expression trees, depth of an expression tree, parameters controlling the annealing temperature in simulated annealing) on two datasets, *Insurance* and *Adult*. We obtained the following results:

Expression tree depth Deeper trees lead to more complex models, mainly due to the possibility of deep compositions of functions. Motivated by finding a good com-

Model	Hyper-parameters	Values
LR	C	$\{0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000\}$
	solver	{newton-cg, lbfgs, liblinear}
	tol	$\{0.0000001, 0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1\}$
	class_weight	{balanced, None}
GAM-splines	lam	$\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
EBM	max_bins	$\{8, 16, 32, 64, 128, 256, 512, 1024\}$
	min_samples_leaf	$\{1, 2, 5, 10, 20\}$
SR-GP	population_size	$\{500, 1000, 1500\}$
	generations	$\{20, 50, 100\}$
SR-Gomea	initmaxtreeheight	$\{4, 6\}$
	popsize	$\{500, 1000\}$
XGBoost	learning_rate	$\{0.0001, 0.001, 0.01, 0.1\}$
	max_depth	$\{2, 3, 5, 10, 15\}$
	min_child_weight	$\{1, 3, 5, 7\}$
	gamma	$\{0, 0.5, 1, 1.5, 2, 5\}$
	col_sample_by_tree	$\{0.3, 0.4, 0.5, 0.7, 1\}$
SVM	C	$\{0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000\}$
	kernel	{linear, poly, rbf}
	tol	$\{0.0000001, 0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1\}$
MLP	hidden_layer_sizes	$\{(16, 16), (16, 8), (8, 8)\}$
	alpha	$\{0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1\}$
	activation	{tanh, relu}
	learning_rate_init	$\{0.0001, 0.001, 0.01, 0.1\}$

Table 4.5: Hyper-parameters tuning

promise between the complexity of the final models and their predictive performance, we restrict the tree depth to 4. In this way, each expression tree, being a perfect binary tree, has 8 leaves.

Number of expression trees There is no noticeable improvement in predictive performance when the number of expression trees exceeds two-thirds of the number of features after data preparation (*viz.*, one hot encoding).

Parameters of the simulated annealing The grid-search is also conducted on the parameters of the simulated annealing. For the heating phase, we find that $\lambda_h = 1.15$ is a good value for reaching quickly a suitable starting temperature T_0. For the cooling phase, the ratio $\lambda_c = 0.85$ is a good balance between algorithmic efficiency and not falling into local optimums too quickly. The parameter γ_c, which controls the growth of s_c (*viz.*, the number of iterations between two updates of temperature) is set to 1.15.

4.5 Results

We apply a 5-fold cross-validation to measure the predictive performances of the models. Results obtained on regression datasets are reported in table 4.7 and table 4.8, the latter for cluster-specific interpretable models trained with a no pooling approach. Results for classification datasets are shown in table 4.9.

4.5.1 Hierarchical structure module

For each dataset, the optimal number of clusters computed in the *Hierarchical structure* module is given in table 4.6. Moreover, to validate the ability of this module to associate an unknown sample to a cluster, a train-test split approach is applied on each training subset of the 5-fold cross-validation. For each training subset, the decision tree classifier is trained, on 80% of the data, to predict, based on the explanatory variables, the cluster to which a new observation belongs. A $f1$-score is computed on the remaining 20% of each training subset. The decision tree classifier is highly accurate on all datasets (see table 4.6).

4.5.2 Regression datasets

BH-GLM and *BH-GLM-int* obtain better RMSE and MAD than the fully interpretable models (*viz.*, *OLS*, *decision tree*, *B-GLM*, *B-GLM-int*) on most datasets. Moreover, on the *French Highway* dataset, *BH-GLM* obtains a 6.64% decrease (6.77%

Dataset	#**clusters**	**f1 (std)**
Regression		
French Highway	4	0.995 (0.003)
Insurance	2	1.0 (0.0)
House	4	0.908 (0.026)
Puma	3	0.975 (0.013)
Satellite	3	0.964 (0.003)
Wind	3	0.944 (0.022)
Breast tumor	2	0.995 (0.004)
Music	3	0.96 (0.029)
Wine	3	0.957 (0.012)
Toxicity	2	0.95 (0.039)
Gas	2	0.99 (0.003)
Airbnb	4	0.985 (0.005)
Classification		
Breastcancer	2	0.961 (0.028)
Adult	4	0.995 (0.002)

Table 4.6: For each dataset: number of clusters selected and performance of the prediction to associate a new observation to its cluster

for *BH-GLM-int*) in RMSE when compared to *B-GLM*, the standard Bayesian model. As expected, we observe the superior predictive performance of modern tree-based algorithms (*viz.*, *XGBoost* and *EBM*). However, the black-box models, *SVM* and *MLP*, do not appear to have the expected high predictive performances as *BH-GLM* and *BH-GLM-int* obtain similar if not better RMSE and MAD on most datasets.

SR-GP, the symbolic regression based on genetic programming, obtains poor results and is even dominated by the fully interpretable models on all datasets. The more recent approach *SR-Gomea* obtains better predictive performance than *SR-GP* but is still dominated by *HSR-trad* and *HSR-max*. *SR-Gomea-op* does not highlight significant predictive gains compared to *SR-Gomea* on most datasets. This validates that restricting the operators makes it possible to obtain interpretable functional forms with more than satisfactory predictive performance on real world datasets.

HSR-trad, the model that, according to our second proposal, should offer a good trade-off between performance and complexity, obtains better RMSE and MAD than *BH-GLM* and *BH-GLM-int* on most datasets. As expected, *HSR-max*, the most complex model resulting from our approach, performs better than *HSR-trad*, except for the *Insurance* dataset where they obtain similar predictive performance. Moreover, the fact that these models have performance metrics with low standard deviations testifies to their robustness. Indeed, they are likely to discover similar solutions on similar datasets.

HSR-trad and *HSR-max*, the cluster-specific models, often show a clear improvement when compared to the global models *SR-trad* and *SR-max*. The partial pooling approach has a clear interest given that *HSR-trad* and *HSR-max* outperform *SR-NP-trad* and *SR-NP-max*, their no pooling variants. Also, our approach to discover a hierarchical structure is robust, efficient and obtain better results on all datasets compared to the approach that considers more naive clusters. Indeed, on all datasets, *HSR-trad* and *HSR-max* are substantially better than *HSR-naive-trad* and *HSR-naive-max*. Moreover, incorporating the data-driven discovery of a hierarchical structure not only provides better predictive performances, it also offers better interpretability by capturing cluster-specific phenomena (see section 4.6).

HSR-max and *GAM-splines* have similar performances on all datasets but the *Insurance* dataset where *HSR-max* discovers a significant interaction between the body mass index and being a smoker. However, the no pooling variant of *GAM-splines* is slightly better than *HSR-max* on the *Insurance* dataset. Finally, *EBM* performs well on all datasets. It obtains the best performances on the *Insurance* and *Gas* datasets and is similar, if not better, to *XGBoost* on the *French Highway*, *Airbnb*, and *Wine* datasets.

	French Highway (8 [a])		*Insurance* (6)		*Airbnb* (12)		*Puma* (6)	
	RMSE (std) [b]	MAD (std)	RMSE (std)	MAD (std)	RMSE (std)	MAD (std)	RMSE (std)	MAD (std)
Local	5.052 (0.250)	3.677 (0.120)	-	-	-	-	-	-
GLMM	**5.044 (0.287)**	**3.676 (0.140)**	-	-	-	-	-	-
OLS	6.213 (0.496)	4.458 (0.216)	6077 (287)	4203 (144)	0.50 (0.011)	0.360 (0.004)	4.471 (0.068)	3.649 (0.026)
Decision tree	6.134 (0.422)	4.405 (0.196)	4739 (324)	2738 (125)	0.490 (0.011)	0.352 (0.004)	3.688 (0.05)	2.881 (0.011)
GAM-splines	5.80 (0.344)	4.245 (0.183)	6021 (299)	4229 (158)	0.464 (0.011)	0.330 (0.004)	4.236 (0.065)	3.492 (0.03)
EBM	5.363 (0.233)	3.968 (0.134)	**4533 (339)**	**2528 (131)**	0.450 (0.013)	0.321 (0.005)	3.283 (0.049)	2.553 (0.037)
SR-GP	8.06 (0.730)	5.602 (0.408)	5168 (360)	2611 (95)	0.563 (0.026)	0.411 (0.014)	4.504 (0.086)	3.593 (0.045)
SR-Gomea	6.309 (0.303)	4.539 (0.128)	4885 (251)	2931 (113)	0.502 (0.010)	0.363 (0.004)	3.362 (0.033)	2.626 (0.057)
SR-Gomea-op	6.297 (0.267)	4.543 (0.155)	4815 (250)	2852 (160)	0.514 (0.01)	0.371 (0.004)	3.238 (0.059)	2.488 (0.044)
B-GLM	6.468 (0.531)	4.58 (0.241)	6080 (266)	4228 (124)	0.498 (0.011)	0.360 (0.004)	4.464 (0.068)	3.648 (0.026)
B-GLM-int	6.356 (0.513)	4.474 (0.235)	5151 (293)	2950 (140)	0.497 (0.01)	0.360 (0.004)	4.282 (0.065)	3.537 (0.026)
XGBoost	5.571 (0.377)	4.084 (0.175)	4667 (346)	2673 (193)	**0.440 (0.011)**	**0.312 (0.003)**	3.257 (0.056)	2.545 (0.028)
SVM	6.310 (0.645)	4.382 (0.268)	4953 (272)	2968 (147)	0.503 (0.011)	0.356 (0.004)	4.493 (0.089)	3.612 (0.037)
MLP	6.140 (0.462)	4.392 (0.212)	4867 (347)	2916 (205)	0.463 (0.013)	0.330 (0.008)	**3.170 (0.052)**	**2.445 (0.029)**
BH-GLM	6.102 (0.441)	4.380 (0.187)	4926 (300)	2930 (130)	0.476 (0.019)	0.354 (0.01)	3.873 (0.133)	3.057 (0.091)
BH-GLM-int	6.004 (0.46)	4.315 (0.191)	4925 (301)	2929 (131)	0.474 (0.02)	0.352 (0.011)	3.871 (0.13)	3.056 (0.088)
SR-trad	6.258 (0.509)	4.488 (0.208)	5219 (330)	3176 (209)	0.510 (0.018)	0.373 (0.011)	3.961 (0.031)	3.162 (0.026)
SR-max	6.186 (0.469)	4.44 (0.224)	4889 (293)	3095 (366)	0.507 (0.011)	0.369 (0.008)	3.528 (0.042)	2.791 (0.041)
HSR-naive-trad	6.472 (0.462)	4.644 (0.261)	6429 (307)	3304 (237)	0.584 (0.067)	0.410 (0.031)	4.156 (0.184)	3.064 (0.162)
HSR-naive-max	6.386 (0.473)	4.582 (0.252)	6328 (307)	3170 (243)	0.545 (0.018)	0.401 (0.026)	4.062 (0.078)	3.002 (0.09)
HSR-trad	5.921 (0.54)	4.250 (0.226)	4840 (308)	2930 (153)	0.475 (0.012)	0.343 (0.011)	3.30 (0.084)	2.547 (0.084)
HSR-max	5.80 (0.507)	4.210 (0.282)	4844 (304)	2933 (149)	0.470 (0.011)	0.340 (0.011)	3.277 (0.082)	2.532 (0.087)

	Satellite (24)		*Wind* (9)		*Breast tumor* (6)		*Music* (78)	
OLS	1.213 (0.009)	1.02 (0.011)	3.289 (0.104)	2.521 (0.077)	10.023 (0.036)	7.88 (0.027)	**0.465 (0.039)**	0.35 (0.03)
Decision tree	1.061 (0.022)	0.621 (0.017)	3.839 (0.112)	2.980 (0.103)	9.844 (0.039)	7.632 (0.03)	0.705 (0.066)	0.497 (0.053)
GAM-splines	0.90 (0.009)	0.624 (0.007)	3.082 (0.084)	2.366 (0.057)	9.663 (0.047)	7.537 (0.034)	0.898 (0.101)	0.678 (0.069)
EBM	0.851 (0.035)	0.575 (0.021)	3.140 (0.089)	2.398 (0.061)	9.519 (0.049)	7.401 (0.04)	0.60 (0.036)	0.441 (0.035)
SR-GP	1.628 (0.438)	1.055 (0.095)	3.858 (0.216)	2.979 (0.191)	10.441 (0.277)	8.151 (0.23)	0.71 (0.154)	0.51 (0.102)
SR-Gomea	1.164 (0.028)	0.897 (0.036)	3.306 (0.128)	2.545 (0.099)	9.988 (0.056)	7.795 (0.044)	0.523 (0.081)	0.379 (0.056)
SR-Gomea-op	1.102 (0.032)	0.826 (0.029)	3.296 (0.101)	2.534 (0.077)	9.973 (0.049)	7.788 (0.025)	0.499 (0.036)	0.369 (0.036)
B-GLM	1.213 (0.008)	1.019 (0.012)	3.308 (0.098)	2.527 (0.074)	10.018 (0.033)	7.776 (0.03)	0.473 (0.033)	0.353 (0.03)
B-GLM-int	1.117 (0.044)	0.905 (0.069)	3.295 (0.098)	2.517 (0.073)	9.995 (0.034)	7.791 (0.026)	0.469 (0.045)	0.358 (0.031)
XGBoost	**0.667 (0.033)**	**0.35 (0.016)**	3.084 (0.075)	**2.365 (0.050)**	**9.435 (0.048)**	**7.288 (0.041)**	0.507 (0.046)	0.367 (0.037)
SVM	1.261 (0.028)	1.003 (0.022)	3.307 (0.102)	2.521 (0.073)	10.045 (0.036)	7.86 (0.029)	0.472 (0.042)	**0.348 (0.03)**
MLP	0.789 (0.047)	0.448 (0.03)	**3.076 (0.087)**	2.367 (0.062)	9.67 (0.04)	7.519 (0.023)	0.498 (0.042)	0.36 (0.04)
BH-GLM	0.986 (0.036)	0.598 (0.034)	3.297 (0.101)	2.538 (0.079)	9.76 (0.035)	7.611 (0.031)	0.472 (0.035)	0.356 (0.027)
BH-GLM-int	0.952 (0.029)	0.553 (0.028)	3.291 (0.106)	2.531 (0.084)	9.751 (0.036)	7.598 (0.033)	0.467 (0.037)	0.353 (0.027)
SR-trad	1.175 (0.05)	0.948 (0.059)	3.342 (0.104)	2.568 (0.080)	10.096 (0.064)	7.917 (0.051)	0.543 (0.081)	0.401 (0.049)
SR-max	1.018 (0.04)	0.784 (0.035)	3.176 (0.081)	2.445 (0.059)	10.03 (0.079)	7.864 (0.06)	0.476 (0.045)	0.359 (0.037)
HSR-naive-trad	1.012 (0.064)	0.645 (0.068)	3.547 (0.041)	2.731 (0.050)	11.932 (0.16)	9.289 (0.241)	0.524 (0.045)	0.372 (0.02)
HSR-naive-max	0.991 (0.034)	0.631 (0.031)	3.562 (0.068)	2.711 (0.046)	11.915 (0.108)	9.267 (0.185)	0.507 (0.056)	0.372 (0.039)
HSR-trad	0.95 (0.034)	0.554 (0.032)	3.205 (0.074)	2.457 (0.054)	9.727 (0.056)	7.595 (0.06)	0.497 (0.05)	0.366 (0.031)
HSR-max	0.934 (0.056)	0.529 (0.062)	3.198 (0.074)	2.455 (0.054)	9.662 (0.061)	7.531 (0.049)	0.471 (0.053)	0.351 (0.027)

	House (5)		*Wine* (8)		*Toxicity (*6)		*Gas* (7)	
OLS	41563 (1270)	24354 (202)	0.754 (0.02)	0.586 (0.012)	1.256 (0.097)	0.949 (0.072)	8.112 (0.133)	5.796 (0.081)
Decision tree	35752 (1199)	20035 (401)	0.753 (0.015)	0.596 (0.013)	1.394 (0.12)	1.059 (0.086)	7.705 (0.127)	5.638 (0.068)
GAM-splines	33460 (1405)	18634 (212)	0.728 (0.029)	0.565 (0.015)	1.245 (0.106)	0.92 (0.08)	5.993 (0.10)	4.187 (0.038)
EBM	31062 (1192)	16921 (133)	0.689 (0.019)	0.537 (0.01)	1.20 (0.095)	0.899 (0.06)	**5.476 (0.072)**	**3.763 (0.038)**
SR-GP	56615 (21299)	24687 (2204)	0.857 (0.068)	0.664 (0.055)	1.457 (0.264)	1.087 (0.152)	10.75 (1.107)	8.165 (0.775)
SR-Gomea	36750 (1693)	20768 (976)	0.742 (0.022)	0.582 (0.015)	1.343 (0.189)	0.978 (0.088)	8.703 (0.144)	6.583 (0.176)
SR-Gomea-op	36865 (1434)	20916 (788)	0.739 (0.021)	0.579 (0.015)	1.267 (0.116)	0.956 (0.086)	8.742 (0.278)	6.764 (0.227)
B-GLM	42104 (1406)	22632 (220)	0.753 (0.02)	0.586 (0.012)	1.237 (0.099)	0.934 (0.073)	8.112 (0.134)	5.797 (0.081)
B-GLM-int	39811 (1375)	20872 (335)	0.751 (0.019)	0.583 (0.012)	1.246 (0.089)	0.941 (0.056)	8.112 (0.137)	5.797 (0.081)
XGBoost	**29630 (1237)**	**15733 (120)**	**0.68 (0.014)**	**0.531 (0.006)**	**1.157 (0.129)**	**0.872 (0.092)**	5.705 (0.190)	4.001 (0.126)
SVM	44879 (1676)	21201 (228)	0.748 (0.018)	0.582 (0.01)	1.286 (0.195)	0.959 (0.132)	6.954 (0.372)	4.875 (0.453)
MLP	36004 (851)	20174 (396)	0.757 (0.071)	0.587 (0.051)	1.280 (0.153)	0.973 (0.112)	6.023 (0.239)	4.223 (0.134)
BH-GLM	35212 (1661)	19444 (708)	0.741 (0.017)	0.576 (0.013)	1.235 (0.095)	0.933 (0.069)	6.87 (0.282)	4.768 (0.159)
BH-GLM-int	35039 (1820)	19079 (443)	0.737 (0.017)	0.574 (0.014)	1.237 (0.098)	0.934 (0.073)	6.87 (0.282)	4.769 (0.159)
SR-trad	37412 (2150)	21067 (524)	0.744 (0.024)	0.584 (0.015)	1.253 (0.089)	0.953 (0.078)	8.153 (0.686)	6.131 (0.515)
SR-max	34716 (2049)	18976 (491)	0.731 (0.019)	0.571 (0.016)	1.233 (0.106)	0.935 (0.071)	7.395 (0.553)	5.467 (0.565)
HSR-naive-trad	37518 (1563)	19592 (433)	0.735 (0.021)	0.575 (0.011)	1.363 (0.112)	0.989 (0.08)	7.46 (0.708)	5.557 (0.607)
HSR-naive-max	37640 (1741)	19294 (472)	0.725 (0.019)	0.566 (0.009)	1.266 (0.071)	0.944 (0.057)	6.621 (0.15)	4.762 (0.152)
HSR-trad	33542 (1127)	17964 (258)	0.724 (0.019)	0.566 (0.013)	1.243 (0.097)	0.933 (0.047)	7.181 (0.618)	5.32 (0.776)
HSR-max	33102 (833)	17403 (155)	0.712 (0.014)	0.559 (0.011)	1.214 (0.065)	0.921 (0.038)	6.421 (0.150)	4.662 (0.153)

[a] number of trees for *SR-** and *HSR-** models

[b] averages and standard deviations of performance metrics obtained on 5-fold cross-validation

Table 4.7: Results obtained on 12 regression datasets

	French Highway		*Insurance*		*Airbnb*		*Puma*	
	RMSE (std)	MAD (std)	RMSE (std)	MAD (std)	RMSE (std)	MAD (std)	RMSE (std)	MAD (std)
OLS	6.05 (0.45)	4.323 (0.182)	4971 (293)	2975 (150)	0.484 (0.01)	0.346 (0.003)	3.867 (0.127)	3.039 (0.088)
GAM-splines	5.747 (0.467)	4.022 (0.168)	4913 (346)	2775 (132)	0.751 (0.649)	0.329 (0.01)	3.523 (0.076)	2.752 (0.048)
EBM	5.18 (0.275)	3.819 (0.113)	4512 (354)	2479 (134)	0.444 (0.012)	0.315 (0.003)	3.320 (0.057)	2.556 (0.039)
SR-GP	6.165 (0.555)	4.347 (0.196)	5795 (691)	2518 (195)	0.512 (0.015)	0.363 (0.005)	3.975 (0.178)	2.973 (0.115)
SR-Gomea	5.998 (0.257)	4.357 (0.131)	4872 (252)	2931 (139)	0.478 (0.011)	0.342 (0.004)	3.273 (0.078)	2.512 (0.043)
SR-Gomea-op	5.959 (0.143)	4.268 (0.105)	4625 (307)	2660 (135)	0.477 (0.009)	0.342 (0.002)	3.217 (0.072)	2.447 (0.049)
B-GLM	6.03 (0.453)	4.33 (0.190)	4879 (298)	2910 (149)	0.484 (0.01)	0.345 (0.003)	3.845 (0.112)	3.027 (0.078)
B-GLM-int	6.02 (0.47)	4.328 (0.191)	4872 (299)	2906 (142)	0.484 (0.01)	0.346 (0.003)	3.547 (0.092)	2.764 (0.043)
SR-NP-naive-trad	6.446 (0.42)	4.641 (0.249)	6751 (521)	3595 (365)	0.552 (0.017)	0.407 (0.014)	4.338 (0.229)	3.22 (0.179)
SR-NP-naive-max	6.377 (0.454)	4.572 (0.239)	6550 (365)	3357 (280)	0.554 (0.036)	0.414 (0.035)	4.266 (0.18)	3.157 (0.137)
SR-NP-trad	6.006 (0.473)	4.338 (0.184)	4842 (290)	2942 (137)	0.489 (0.019)	0.351 (0.006)	3.536 (0.087)	2.754 (0.071)
SR-NP-max	6.043 (0.617)	4.399 (0.30)	4844 (318)	2940 (167)	0.483 (0.011)	0.346 (0.005)	3.386 (0.063)	2.653 (0.048)

	Satellite		*Wind*		*Breast tumor*		*Music*	
OLS	1.101 (0.053)	0.602 (0.05)	3.291 (0.094)	2.532 (0.074)	9.8 (0.123)	7.663 (0.117)	1.028 (0.458)	0.519 (0.116)
GAM-splines	0.899 (0.042)	0.477 (0.026)	3.108 (0.087)	2.387 (0.063)	9.62 (0.067)	7.488 (0.063)	0.827 (0.031)	0.614 (0.024)
EBM	0.874 (0.03)	0.44 (0.017)	3.200 (0.055)	2.453 (0.050)	9.493 (0.057)	7.367 (0.054)	0.613 (0.058)	0.429 (0.042)
SR-GP	1.209 (0.096)	0.659 (0.055)	3.678 (0.099)	2.837 (0.081)	10.242 (0.286)	7.994 (0.249)	0.762 (0.076)	0.545 (0.061)
SR-Gomea	1.025 (0.052)	0.594 (0.027)	3.283 (0.071)	2.514 (0.047)	9.798 (0.098)	7.636 (0.082)	0.624 (0.079)	0.428 (0.048)
SR-Gomea-op	0.991 (0.058)	0.553 (0.018)	3.287 (0.091)	2.516 (0.074)	9.798 (0.103)	7.658 (0.096)	0.584 (0.06)	0.409 (0.043)
B-GLM	0.988 (0.044)	0.598 (0.047)	3.293 (0.094)	2.535 (0.074)	9.8 (0.077)	7.641 (0.088)	0.652 (0.092)	0.431 (0.061)
B-GLM-int	0.986 (0.036)	0.598 (0.035)	3.287 (0.099)	2.529 (0.079)	9.798 (0.064)	7.62 (0.059)	0.721 (0.065)	0.508 (0.058)
SR-NP-naive-trad	1.025 (0.04)	0.673 (0.07)	3.675 (0.139)	2.822 (0.105)	12.0 (0.161)	9.365 (0.254)	0.549 (0.05)	0.408 (0.026)
SR-NP-naive-max	1.035 (0.148)	0.664 (0.125)	3.608 (0.079)	2.734 (0.058)	11.953 (0.149)	9.311 (0.228)	0.576 (0.102)	0.388 (0.03)
SR-NP-trad	0.975 (0.036)	0.574 (0.036)	3.312 (0.073)	2.539 (0.050)	9.865 (0.141)	7.711 (0.126)	0.550 (0.038)	0.392 (0.019)
SR-NP-max	0.953 (0.069)	0.539 (0.038)	3.278 (0.063)	2.449 (0.038)	9.8 (0.117)	7.662 (0.109)	0.564 (0.059)	0.381 (0.034)

	House		*Wine*		*Toxicity*		*Gas*	
OLS	35162 (1651)	19186 (475)	0.736 (0.017)	0.573 (0.014)	1.264 (0.109)	0.952 (0.073)	7.169 (0.281)	5.06 (0.158)
GAM-splines	32633 (1111)	17535 (316)	0.729 (0.019)	0.566 (0.012)	1.293 (0.168)	0.957 (0.097)	5.671 (0.168)	3.85 (0.069)
EBM	31298 (944)	16426 (209)	0.683 (0.019)	0.527 (0.015)	1.207 (0.114)	0.890 (0.077)	5.423 (0.119)	3.721 (0.060)
SR-GP	49985 (7436)	22503 (1690)	0.807 (0.021)	0.627 (0.01)	1.395 (0.27)	0.99 (0.126)	9.966 (1.797)	6.843 (0.161)
SR-Gomea	33156 (863)	18045 (403)	0.729 (0.017)	0.574 (0.013)	1.238 (0.066)	0.943 (0.049)	7.614 (0.205)	5.607 (0.173)
SR-Gomea-op	33091 (943)	17950 (292)	0.734 (0.016)	0.577 (0.011)	1.273 (0.117)	0.959 (0.092)	7.747 (0.338)	5.78 (0.293)
B-GLM	41837 (1433)	20963 (341)	0.738 (0.019)	0.574 (0.014)	1.278 (0.119)	0.931 (0.081)	6.864 (0.276)	4.765 (0.157)
B-GLM-int	40835 (1380)	20610 (285)	0.739 (0.021)	0.575 (0.017)	1.293 (0.129)	0.959 (0.084)	6.47 (0.152)	4.454 (0.071)
SR-NP-naive-trad	38810 (1892)	20545 (684)	0.734 (0.027)	0.573 (0.013)	1.345 (0.173)	0.97 (0.072)	7.873 (1.55)	6.022 (1.733)
SR-NP-naive-max	38260 (1531)	19466 (315)	0.723 (0.016)	0.568 (0.009)	1.262 (0.096)	0.932 (0.073)	7.105 (0.906)	5.223 (0.827)
SR-NP-trad	33922 (1158)	18147 (247)	0.73 (0.019)	0.574 (0.013)	1.326 (0.125)	0.989 (0.094)	7.549 (0.515)	5.420 (0.439)
SR-NP-max	34021 (1251)	17841 (358)	0.724 (0.018)	0.569 (0.013)	1.347 (0.199)	0.975 (0.069)	7.296 (0.728)	5.34 (0.749)

Table 4.8: Results obtained by cluster-specific interpretable models

4.5.3 Classification datasets

For the *breast cancer* dataset, the search for a hierarchical structure (see section 3.2) identifies two homogeneous clusters and there is no need to learn cluster-specific models. Thus, for this dataset, we consider only the global models *SR-trad* and *SR-max*. Still for this same dataset, all models perform very well. Nonetheless, *SR-max* is as, or even more, efficient than *LR* and black-box models. Also, we observe that *BH-GLM* and *BH-GLM-int*, the Bayesian models that account for the hierarchical structure in the data, perform better than the standard Bayesian models *B-GLM*. Embedding major first-order interaction also improves the predictive performance.

XGBoost and *EBM* achieve the best results on the *adult* dataset. *HSR-trad* obtains similar results to *LR*, thus suggesting that sparse models can have similar predictive performances than models that consider all explanatory variables, the latter being prone to the *information overload* phenomenon. *HSR-max* obtains similar, if not better, results than black-box models. Its performance is close to the one of *GAM-splines*. As observed previously for regression datasets, *HSR-trad* and *HSR-max* still perform better than *BH-GLM* and *BH-GLM-int*. Also, they are better than the global models (*viz.*, *SR-trad* and *SR-max*), thus confirming the benefits of incorporating a hierarchical structure in case of classification problems.

4.5.4 Discussion

Confirming previous studies [76, 19], we observe that EBM, as a variant of GAM, is very efficient on both regression and classification tasks. Moreover, this model meets many of the expected criteria for interpretability enumerated in [10]. However, it also has limitations that can make it unsuitable for road safety analysis. First, different optimization strategies adopted to learn an EBM model, can lead to different interpretations of its predictions [20]. However, for road safety analysis, trust in the identification of the main risk factors is required by experts when they elaborate remedial actions. Moreover, for satisfactory interpretability, it helps if a GAM has a small number of components and if each component function is relatively smooth. However, EBM, due to their reliance on boosted trees, can hardly maintain these constraints [105]. With our approach, field experts are more likely to be confident in models with cluster-specific behaviors and stable functional forms that highlight a selection of relevant factors and their interactions.

Furthermore, for the *French Highway* dataset, as already observed in [126], the best known strategy to estimate the number of crash counts is to average, for each highway network segment, the number of accidents that occurred in previous years (c.f., the *Local* model in table 4.7). We also observe that the predictive performances of *GLMM* are equivalent to those of the *Local* model. When looking at the model in

	Breast cancer (20)					
	acc (std)	AUPR (std)	AUROC (std)	f1 (std)	pre (std)	rec (std)
LR	**0.980 (0.012)**	**0.995 (0.006)**	0.995 (0.007)	**0.974 (0.014)**	**0.981 (0.026)**	0.968 (0.026)
Decision tree	0.940 (0.004)	0.898 (0.076)	0.921 (0.016)	0.919 (0.002)	0.923 (0.010)	0.915 (0.010)
GAM-splines	0.965 (0.019)	0.991 (0.008)	0.992 (0.008)	0.953 (0.022)	0.962 (0.012)	0.945 (0.035)
EBM	0.963 (0.016)	0.989 (0.011)	0.992 (0.009)	0.950 (0.021)	0.965 (0.015)	0.935 (0.036)
SR-GP	0.946 (0.033)	0.975 (0.021)	0.979 (0.022)	0.925 (0.044)	0.944 (0.073)	0.909 (0.028)
B-GLM	0.965 (0.013)	0.988 (0.009)	0.989 (0.01)	0.956 (0.015)	0.938 (0.022)	**0.976 (0.023)**
B-GLM-int	0.971 (0.020)	0.983 (0.007)	0.983 (0.009)	0.949 (0.024)	0.945 (0.024)	0.955 (0.040)
XGBoost	0.965 (0.020)	0.988 (0.011)	0.989 (0.011)	0.952 (0.026)	0.966 (0.022)	0.940 (0.038)
SVM	0.978 (0.005)	**0.995 (0.006)**	**0.995 (0.005)**	0.971 (0.006)	0.981 (0.027)	0.963 (0.026)
MLP	0.961 (0.023)	0.992 (0.006)	0.993 (0.007)	0.948 (0.031)	0.965 (0.052)	0.936 (0.064)
BH-GLM	0.965 (0.016)	0.985 (0.005)	0.987 (0.006)	0.951 (0.019)	0.964 (0.018)	0.938 (0.035)
BH-GLM-int	0.965 (0.016)	0.987 (0.006)	0.990 (0.006)	0.952 (0.022)	0.966 (0.026)	0.938 (0.041)
HSR-naive-trad	0.939 (0.023)	0.943 (0.044)	0.941 (0.049)	0.911 (0.036)	0.939 (0.058)	0.887 (0.064)
HSR-naive-max	0.941 (0.028)	0.943 (0.045)	0.941 (0.049)	0.915 (0.043)	0.94 (0.059)	0.895 (0.072)
SR-trad	0.944 (0.030)	0.979 (0.011)	0.984 (0.011)	0.926 (0.035)	0.924 (0.059)	0.931 (0.033)
SR-max	0.972 (0.022)	0.994 (0.006)	0.995 (0.007)	0.963 (0.030)	0.972 (0.029)	0.955 (0.056)
	Adult (38)					
LR	0.850 (0.004)	0.762 (0.010)	0.905 (0.002)	0.657 (0.010)	0.733 (0.014)	0.595 (0.016)
Decision tree	0.846 (0.004)	0.749 (0.006)	0.867 (0.003)	0.614 (0.009)	0.768 (0.011)	0.512 (0.009)
GAM-splines	0.860 (0.001)	0.795 (0.009)	0.915 (0.002)	0.680 (0.010)	0.754 (0.011)	0.621 (0.019)
EBM	0.869 (0.003)	0.819 (0.010)	0.924 (0.004)	0.702 (0.010)	0.777 (0.012)	0.640 (0.018)
SR-GP	0.806 (0.011)	0.663 (0.009)	0.847 (0.007)	0.399 (0.065)	**0.790 (0.030)**	0.270 (0.057)
B-GLM	0.85 (0.005)	0.766 (0.009)	0.905 (0.003)	0.659 (0.011)	0.732 (0.014)	0.599 (0.017)
B-GLM-int	0.845 (0.004)	0.735 (0.009)	0.870 (0.009)	0.633 (0.011)	0.701 (0.013)	0.578 (0.017)
XGBoost	**0.872 (0.002)**	**0.827 (0.006)**	**0.927 (0.002)**	**0.705 (0.005)**	0.786 (0.012)	0.639 (0.014)
SVM	0.850 (0.006)	0.756 (0.010)	0.890 (0.004)	0.642 (0.022)	0.747 (0.015)	0.563 (0.033)
MLP	0.851 (0.002)	0.770 (0.011)	0.908 (0.003)	0.675 (0.013)	0.713 (0.028)	**0.644 (0.044)**
BH-GLM	0.838 (0.011)	0.738 (0.016)	0.875 (0.013)	0.635 (0.015)	0.705 (0.014)	0.578 (0.020)
BH-GLM-int	0.839 (0.012)	0.740 (0.018)	0.878 (0.016)	0.638 (0.020)	0.708 (0.017)	0.580 (0.021)
SR-trad	0.825 (0.009)	0.683 (0.034)	0.852 (0.014)	0.586 (0.054)	0.703 (0.018)	0.501 (0.067)
SR-max	0.842 (0.005)	0.734 (0.009)	0.890 (0.007)	0.633 (0.018)	0.714 (0.009)	0.569 (0.023)
HSR-naive-trad	0.731 (0.076)	0.644 (0.075)	0.832 (0.03)	0.581 (0.021)	0.506 (0.129)	0.773 (0.195)
HSR-naive-max	0.822 (0.014)	0.684 (0.041)	0.862 (0.011)	0.611 (0.034)	0.659 (0.067)	0.589 (0.122)
HSR-trad	0.849 (0.006)	0.760 (0.021)	0.895 (0.011)	0.655 (0.020)	0.730 (0.018)	0.594 (0.022)
HSR-max	0.857 (0.005)	0.778 (0.009)	0.902 (0.007)	0.670 (0.017)	0.739 (0.004)	0.611 (0.026)

Table 4.9: Results on the classification datasets

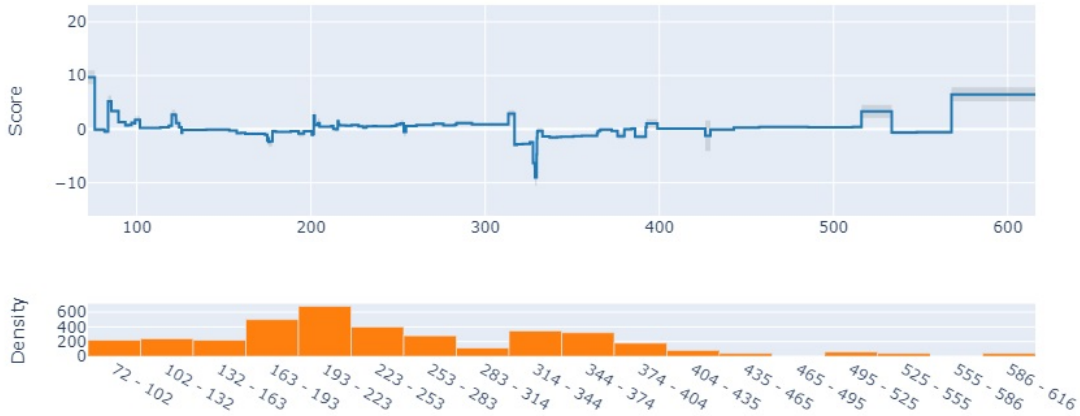


Figure 4.3: Global explanation plot provided by the InterpretML framework for an *EBM* model on the altitude variable on the *French Highway* dataset

more detail, we understand that the introduced random effects adjust the predictions of fixed effects such that they match the averages of the historical crashes observed on the segments. However, these models (*viz.*, *GLMM* and *Local*) do not offer much insight about the associations between crash counts and risk factors. We observed that flexible models, such as EBM, are able to approach in performance these local models by discovering quasi-identifiers of road segments. For example, an EBM discovers a complex nonlinear relationship between the altitude and the number of accidents, see Fig. 4.3. Accidents appear more likely for the lowest altitudes. However, this phenomenon should not be interpreted as a potential risk factor linked to the altitude. In fact, the model is using the altitude as a proxy variable to identify a group of nearby road segments. Therefore, in that particular context, EBM, despite its good predictive performance, does not always provide relevant information to field experts. It can even, at times, mislead them.

Although plots like the one of Fig. 4.3 make it possible to identify these potentially misleading models' behaviors, the EBM model does not provide alternative associations between the explanatory variables and the target. With our approach, the risk of misinterpretation is reduced thanks to the successive models on the Pareto front: from less complex, which capture only overall effects, to most complex, which are flexible enough to focus on hazardous configurations specific to a few roadway segments. Through such a dynamic interpretative process, field experts can use the model best suited to meet their needs. In the next section, we illustrate this process with an experimental study applied to the *French Highway* dataset.

4.6 Dynamic interpretative process

4.6.1 Introduction

A glass-box interpretable crash prediction model able to highlight the marginal effects of the potential risk factors is a valuable tool to help design effective highway safety policies. Indeed, in such a context, the predictive model must limit as much as possible the risks of misinterpretation. In this section, we show how, on a realistic use case validated by field experts, the framework we propose responds to this challenge by making possible an efficient dynamic interpretative process that leads to selecting a predictive model well suited to the task at hand.

Each functional block of our framework (see figure 3.2) plays an important role in the methodology we propose. The *hierarchical structure* module brings out a relevant partition of the highway network. This partitioning of the measured crash counts, with their associated explanatory variables, not only improves the performance of the models (see section 4.5) but also refines the analysis of marginal effects by capturing cluster-specific phenomena. Moreover, based on the multi-objective optimization leading to *global* and *cluster-specific models*, safety experts can explore a list of optimal models that offers a variety of alternatives along the performance/complexity trade-off axis. They will often start with the simpler ones that capture only the global effects, and move towards the more complex ones that can represent more localized phenomena. To support this methodology, we developed, in dialogue with field experts, a graphical user interface (see Fig 4.4).

In the following description of a case study, we suppose that the partitioning of the highway network and the training of the global and cluster-specific models have already been done on ten years of data, from January 1st, 2008 to December 31th, 2017. Data from 2018 is used to validate that, based on out-of-sample predictions, the framework provides useful information to safety experts.

4.6.2 Use case

Partitioning the highway network

From the *hierarchical structure* module of section 3.2, safety experts identified four relevant clusters (see Fig. 4.5). In table 4.10, we share some descriptive statistics of explanatory variables and crash count for each cluster. We observe that clusters are heterogeneous on most of the explanatory variables and on the crash count. For instance, cluster 1 is representative of hazardous segments with high traffic, whereas cluster 3 is made of less accidental secondary segments that connect the main highways.

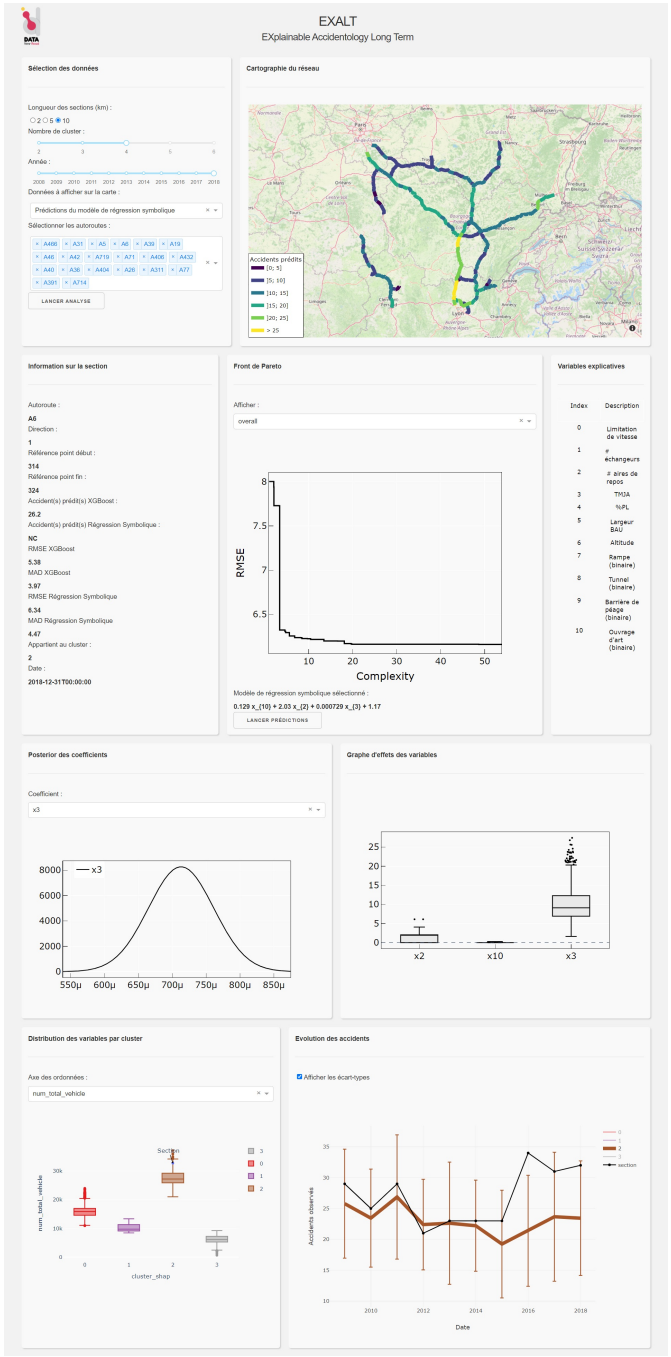


Figure 4.4: Screenshot of the web application

Cluster	0 (1491)				1 (485)				2 (965)				3 (823)			
	mean	std.	min	max	mean	std.	min	max	mean	std.	min	max	mean	std.	min	max
Dependent variable																
Crash count observed	10.27	6.92	0	78	22.83	9.48	1	65	14.03	5.91	0	38	5.90	3.59	0	21
Explanatory variables																
Speed limit	128.8	4.08	110	130	124.95	8.41	104	130	129.07	3.51	110	130	128.94	3.62	108.4	130
#Interchanges	0.05	0.32	0	3 0.37	0.94	0	7	0.10	0.43	0	2	0.45	0.98	0	4	
#Resting places	0.56	0.54	0	2	0.73	0.49	0	2	0.82	0.63	0	3	0.48	0.55	0	2
AADT	10000.97	1465.42	7685	14658	26606.63	3513.14	20949	37113	15920.91	1721.15	13382	20945	5627.28	1225.75	659	8438
%age heavy vehicles	18.22	6.73	2.08	30.3	17.13	4.54	5.2	31.4	16.84	5.35	1	35.06	16.28	4.67	1	34.13
Right shoulder width	2.91	0.27	1.15	3	2.97	0.06	2.68	3	2.89	0.34	1.03	3	2.97	0.07	2.6	3
Altitude	274.53	104.72	79.76	615.78	210.17	70.51	71.97	427.78	286.83	96.58	84.03	521.02	188.75	100.25	79.76	549.63
Presence of ramps	0.30	0.46	0	1	0.17	0.37	0	1	0.2	0.4	0	1	0.21	0.41	0	1
Presence of tunnels	0.04	0.2	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Presence of tollgates	0.43	0.49	0	1	0.55	0.5	0	1	0.45	0.5	0	1	0.36	0.48	0	1
Presence of bridges	0.17	0.38	0	1	0.17	0.38	0	1	0.17	0.37	0	1	0.24	0.43	0	1

Table 4.10: Descriptive statistics of the dependent variable and explanatory variables for each cluster on the *FrenchHighway* dataset

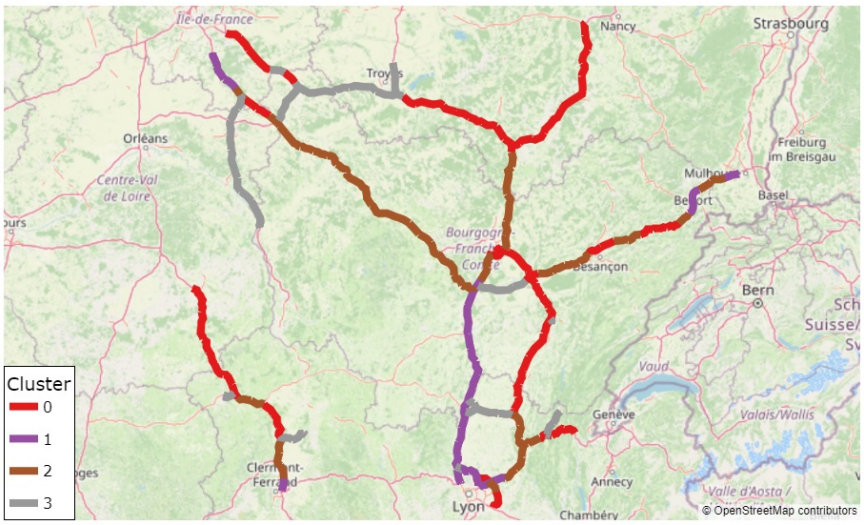


Figure 4.5: Clusters identified for 2018

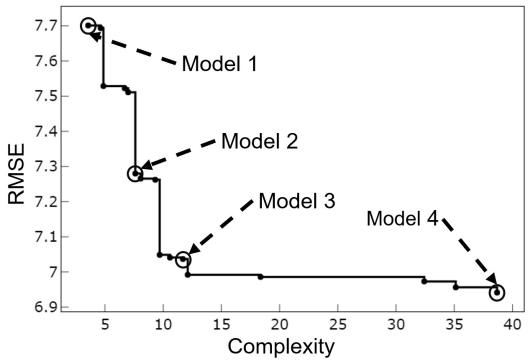
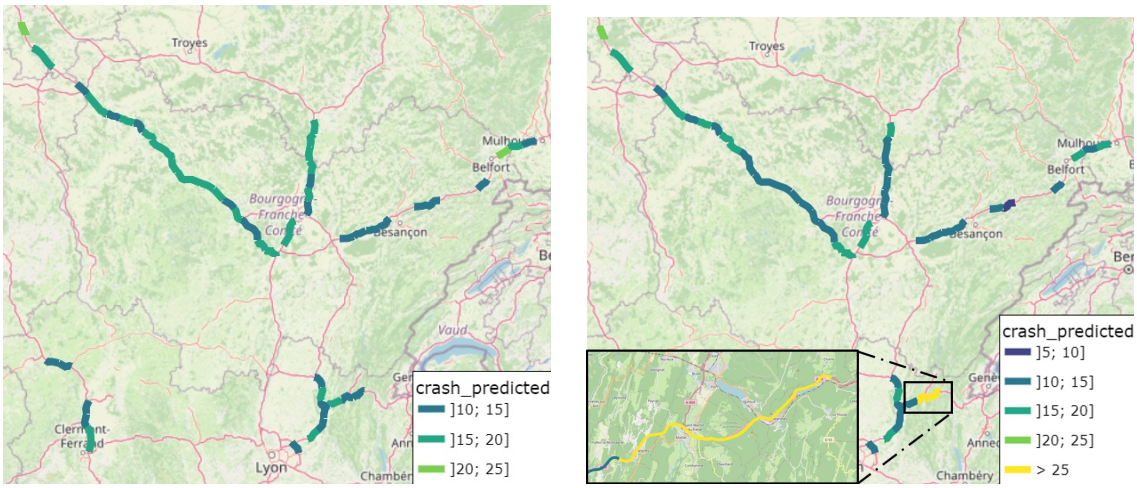


Figure 4.6: Pareto front for cluster 0 specific models

From global models to cluster-specific models

After selecting a cluster of interest, safety experts can navigate within the series of cluster-specific models that make up the Pareto front (see Fig. 4.6), from the least complex one (*viz.*, model 1) to the most complex one (*viz.*, model 4). For illustrative purposes, we focus on the moderately hazardous cluster 0, composed mainly of rural and mountainous segments. Model 1 corresponds to the functional form of the global model (section 3.4.3) whose coefficients are inferred based on cluster 0 data.

$$\text{model 1: } \hat{y} = 5.95 + 0.000394x_3 + 0.312x_{10} + 2.7x_2$$



(a) Model 1 (b) Model 2

Figure 4.7: Crash count predictions for 2018

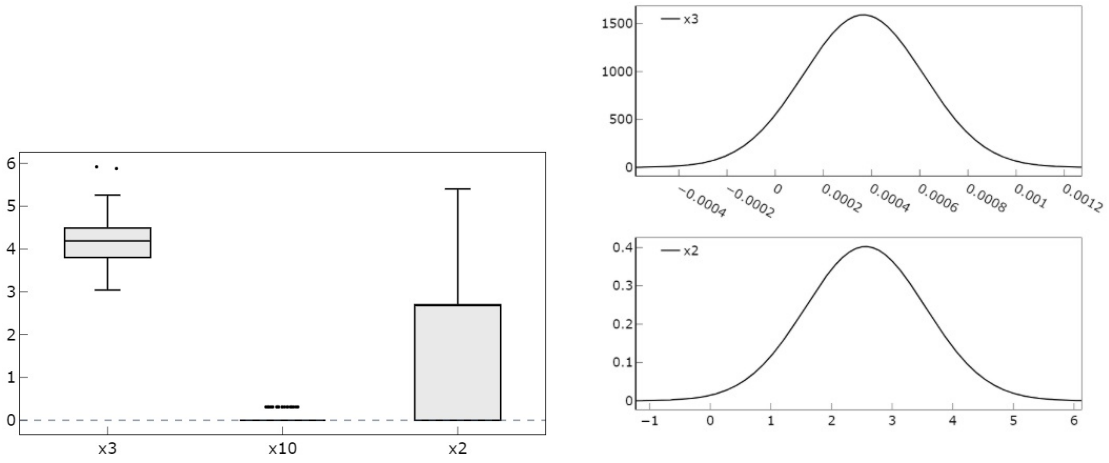
with $\hat{y}$ being the predicted crash count, x_3 the average annual daily traffic, x_{10} the presence of bridges (binary) and x_2 the number of rest areas. From the effects plots of Fig. 4.8a, it appears that, for model 1, the amount of traffic and the number of rest areas have the more prominent marginal effects. Moreover, from the posterior distributions of Fig. 4.8b, one can see that these risk factors have narrow credible intervals.

Model 1 captures only the global risk factors. When considering models of increasing complexity, more specific effects will appear. For instance, model 2 (see Fig. 4.6) is defined as:

$$\text{model 2: } \hat{y} = 3.34 + 0.000555x_3 + 2.34x_2 + 1.43x_1 + 0.055x_{10} + 0.117x_0x_8$$

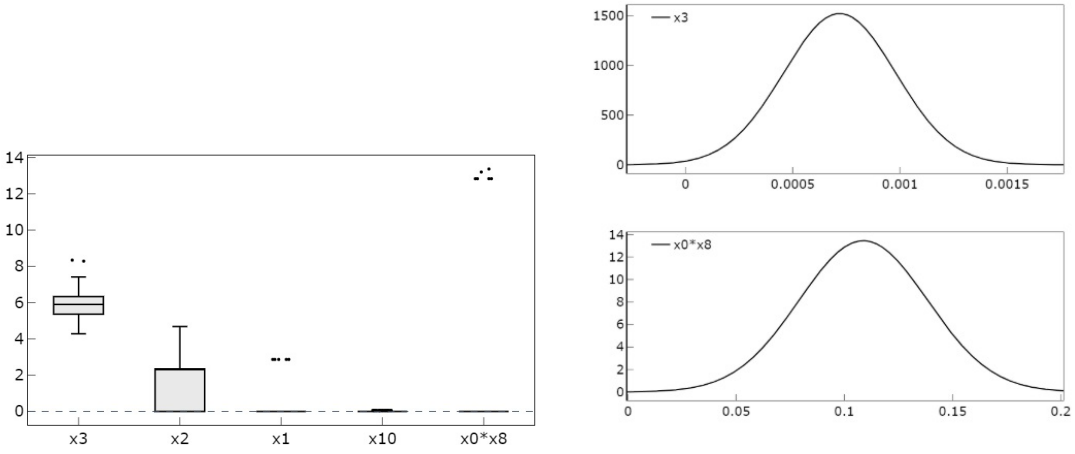
where the additional variables x_0, x_1 and x_8 are, respectively, the speed limit, the number of interchanges and the presence of tunnels. Out-of-sample predictions from models 1 and 2 differ locally (see Fig 4.7a and Fig 4.7b). In particular, in mountainous areas, segments considered as moderately hazardous by the first model, are now associated with a high risk of accidents due to the discovery of a first-order interaction between the presence of tunnel and the speed limit. Safety experts, by combining prior knowledge of the network with the observed transition from model 1 to model 2, are confident that this interaction is one of the main reasons why a large number of accidents have occurred on these segments during the ten years covered by the training data. This discovery may support a proposal for reducing the authorized speed limit on these specific segments.

Also, more refined models are sometimes able to better identify less hazardous



(a) (b)

Figure 4.8: Effects plots and posteriors of the most influencing variables for model 1. Effects plots are obtained by computing for all observations the effect of a variable j on the crash count, defined by $\text{effect}_j^{(i)} = \beta_j x_j^{(i)}$, where β_j is the coefficient estimate of the j-th variable of the model and $x_j^{(i)}$ is the value of variable j on the i-th observation [91].



(a) (b)

Figure 4.9: Effects plots and posteriors plots of the most influencing variables for model 2

configurations. For instance, a segment close to Belfort was identified as highly hazardous by model 1 (between 20 and 25 predicted crashes) while model 2 considers it as moderately hazardous (between 15 and 20 predicted crashes). As remedial or preventive actions may involve significant costs, predicting less hazardous configurations is therefore beneficial because actions will be implemented at first on more prominent hazardous segments. So, thanks to this more complex model, field experts optimise the order of safety policies and funding shall be better allocated.

Nonetheless, one of the major difficulties for the interpretation of more complex models is related to the introduction of collinearities and interactions between continuous variables. To illustrate this, consider model 3 from Fig. 4.6:

$$\text{Model 3: } \hat{y} = -33.3 - 0.00489x_{10} + 1.57x_2 - 9.37 \cdot 10^{-6}x_3x_6 + 0.00286x_3 + 0.15x_6$$

Model 3 is characterized by an interaction between the averaged altitude x_6 and the traffic x_3. By focusing on this interaction, model 3 better fits training data than model 2 but its effects plots (see Fig 4.11a) are arguably more difficult to interpret. However, our framework only produces differentiable closed-form expressions for which it is always possible to compute the partial analytical derivatives (PD) w.r.t. variables of interest, to quantify explicitly their partial effects (i.e., a measure of the conditional effect of a variable on the target) [7]. In this sense, we can understand how a unit change in an explanatory variable affects the crash count when other variables are held constant. For instance, the partial derivatives for the trafic x_3 and altitude x_6 are respectively:

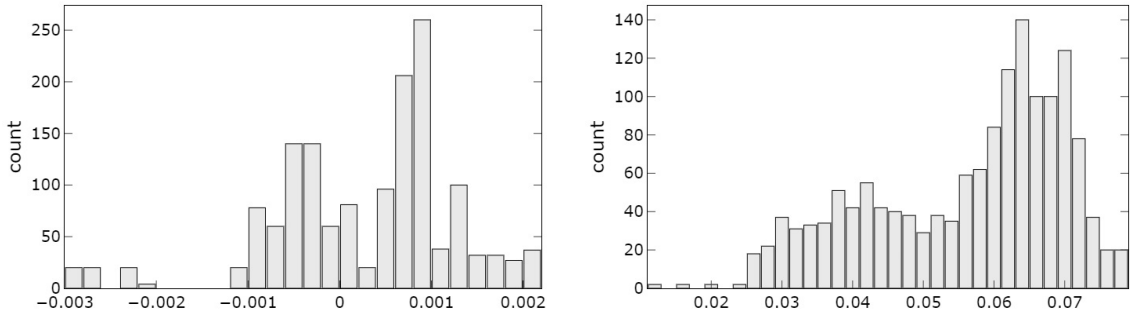
$$\text{PD}(x_3) = \frac{\delta\hat{y}}{\delta x_3} = 0.00286 - 9.37 \cdot 10^{-6}x_6, \quad \text{PD}(x_6) = \frac{\delta\hat{y}}{\delta x_6} = 0.15 - 9.37 \cdot 10^{-6}x_3$$

Histograms of the pointwise partial derivatives can be useful interpretative tools (see Fig. 4.10). Although the partial effects of the traffic x_3 are mostly positive, a few are negative for segments of high altitudes and above average traffic (see Table 4.11). Thus, these variations in partial derivatives emphasize that the relation between the crash count and x_3 is more complex than the linear dependency proposed by model 1 and model 2. By introducing this novel interaction, model 3 manages to capture more variability in the dependent variable than the previous models.

Finally, model 4 is much more complex:

$$\begin{aligned}\text{Model 4: } \hat{y} = & -34.7 + 8.5 \cdot 10^{-6}x_1^2 - 0.000671x_1x_3 - 8.5 \cdot 10^{-6}x_1x_6 + 13.5x_1 \\ & + 0.0113x_{10} + 1.92x_2 - 8.5 \cdot 10^{-6}x_3x_6 + 0.000679x_3x_8 + 0.00269x_3 \\ & - 0.0276x_4 + 1.61x_5 + 0.133x_6 + 0.12x_7 - 0.276x_8 + 0.0472x_9\end{aligned}$$

As we can see from Fig. 4.11b, the introduction of new correlated terms improves slightly the fit. To capture extra variability in the dependent variable, model 4



(a) PD(x_3) (b) PD(x_6)

Figure 4.10: Histograms of partial derivatives for (a) the traffic x_3 and (b) the altitude x_6, for model 3.

	x_3		x_6	
	overall	PD(x_3) < -0.0015	overall	PD(x_3) < -0.0015
count	1491	64	1491	64
mean	10001	12206	274	579
std	1465	997	105	32
min	7685	10877	79	521
max	14658	14658	616	616

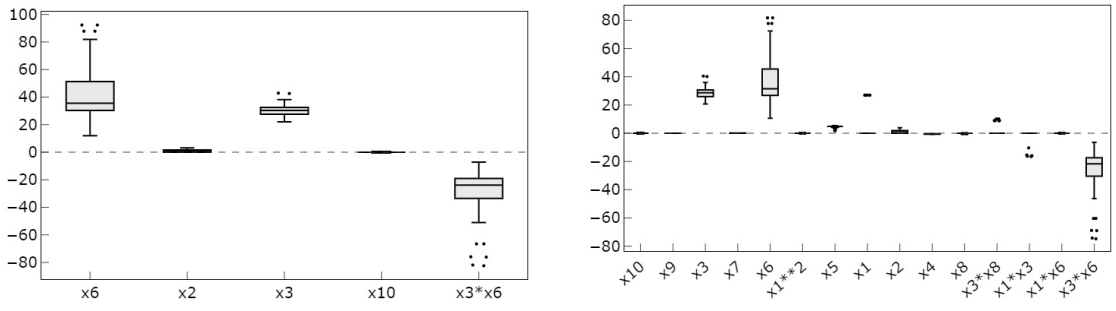
Table 4.11: Description of explanatory variables for the overall cluster-specific data and for samples where partial derivatives w.r.t. x_3 are the lowest.

introduces highly correlated combinations of terms. From model 3 to model 4, a sharp increase in complexity for a small gain in performance should alert the user to the risk of no longer understanding the inner workings of model 4: time must be spent at studying the various partial effects before deciding if the model can still be trusted.

Specificities and benefits of the ranking-by-complexity approach

Thanks to our complexity metric (see Eq. 3.3), model 3 does not dominate model 2 even though they have the same number of terms and their terms have equal complexities. If we had not penalize collinearities, then model 2, which is of high interest to field experts, would not have been included in the Pareto front. In this sense, the ranking-by-complexity favors a progressive analysis of numerous instructive models.

Among Fig. 4.6 models, some can attain similar predictive performances while bringing out different effects of the explanatory variables. This can be understood from the point of view of the Rashomon effect [16] which characterizes problems



(a) Model 3 (b) Model 4

Figure 4.11: Effects plots

where many accurate-but-different models exist to describe the same data [110]. As discussed by [104], we argue that the availability of multiple efficient predictive models is useful since field experts may have more flexibility in choosing a model that they find interpretable. Moreover, we help them in this process as our definition of the complexity warns them when models are likely to be difficult to understand.

Finally, the dynamic interpretative process can be a useful tool to construct new handmade predictive models, based on the knowledge learnt by analyzing the Pareto optimal models. For instance, we have seen that the interactions introduced in model 2 and model 3 are both valuable. The user could consider building a new model with both of them.

Towards causality

A central question remains: among these different models, how can be distinguished the trustworthy ones from the ones based on spurious associations due to inductive bias? As illustrated above, one way is to rely on the diligence of the user equipped with expert knowledge and effective tools. This could also be partially automated when prior knowledge of the conditional independences between variables is formalized, e.g., as a causal graph [95]. Such approaches are beyond the scope of our current work. However, our framework fosters a dynamic interpretative process that, combined with a clear quantification of uncertainty, is a useful tool to identify variables of interest and to understand how they interact. Therefore, we can surmise that our framework could facilitate the development of causal models.

Chapter 5

Conclusion and perspectives

5.1 Conclusion

Predictive models are being used increasingly to make high stake decisions. For many applications, there is a need for both accuracy and interpretability. For instance, in highway safety analysis, we argue that a preference should be given to predictive models that are both accurate and glass-box interpretable in order to increase the confidence of safety experts in the identification of hazardous segments.

Motivated by these requirements, in our first contribution, we introduced a framework to build efficient and interpretable Bayesian hierarchical models for regression or classification tasks. We proposed a data-driven discovery of objective priors in the form of a hierarchical structure and strong first-order interactions between explanatory variables. We start with a trained and usually efficient, albeit opaque, ML algorithm in order to compute for each observation the Shapley values of the explanatory variables. Then, a partition of the instances, related to how the ML algorithm predicts the target, emerges from the hierarchical agglomerative clustering of the observations described by the Shapley values. Furthermore, we analyze the structure of a trained self-adaptive polynomial network to discover important first-order interactions. This prior knowledge is then embedded into the definition of a Bayesian hierarchical GLM with nonlinear functional form.

In our second contribution, we proposed to exploit even better the discovery of the hierarchical structure by using symbolic regression to discover sparse interpretable models with rich interactions. We combined a multi-objective simulated-annealing-based symbolic regression and a partial pooling approach to discover models that capture global effects and cluster-specific effects. More specifically, we start by computing a Pareto front of global predictive models. We select among these models the one offering a good trade-off between its predictive performance and its complexity.

Afterwards, for each cluster of the previously discovered hierarchical structure, the global model is used as the starting seed for a new multi-objective symbolic regression. Finally, the best models, i.e. the ones appearing on the Pareto fronts, are re-estimated through Bayesian inference in order to associate uncertainty estimates to their coefficients.

On fourteen datasets, covering both regression and classification tasks, our two proposals outperform most interpretable models. On some datasets, we achieve performance comparable to that of non-parametric black-box models. Furthermore, we presented a case study based on the highway network dataset to validate the new dynamic interpretative process made possible by our second proposal. As our approach discovers transparent and parsimonious symbolic models, safety experts can be more confident in their understanding of the relations between the explanatory variables and the dependent variable. Moreover, thanks to Bayesian inference, the risk factors are associated with measures of uncertainty. In addition, the use of Pareto optimization allows field experts to build a multi-scale view of the risk factors, from the most general to the most specific.

5.2 Perspectives

First, we plan to improve the dynamic interpretative process by generating human-readable explanations of the relationships between Pareto optimal models, such as: *model X is an extension of model Y and improves the predictive performance on training data by* 3% *thanks to the introduction of a new interaction between variables* x_A *and* x_B. In this way, field experts will have a direct understanding of the Pareto front.

Then, we are going to test several divisions of the road network into segments. Currently, we generate the dataset for crash prediction, we first divide the network with segments of equal length and then compute the variables for each segment (see section 4.1.1). With this procedure, we can capture the heterogeneity of the explanatory and dependent variables along the network. However, other types of meshes can be used. For instance, we can use the segments identified by the SURE approach (see section 1.1). Also, Deublein *et al.* [27] divide the network into homogeneous segments where explanatory variables are constant, assuming that the dependent variable is weighted by an exposure term that depends on the length of the section and traffic. Thus, future work will analyze the effects of different types of meshes. Also, the user interface will allow to choose the type of mesh. Thus, field experts will be more aware of the effects of this important prior.

Moreover, our framework relies on a specific approach to discover a hierarchical structure. Even though we validated its robustness on numerous datasets, promis-

ing next steps involve analyzing other methods that are compatible with ours. For instance, in [101, 100], the authors propose an efficient ensemble feature importance method where multiple feature importance approaches are applied to a set of ML models and their crisp importance values are combined to produce a final importance for each feature. Thus, we will constitute a benchmark of feature importance methods [10] and evaluate them based on their efficiency, scalability, and on the quality of computed clusters.

Lastly, we will also extend the framework for near real-time crash risk assessment. New spatio-temporal explanatory variables, such as weather-related variables (e.g. rainfall, snow, wind, etc.) or the pavement quality, will be introduced in the analysis. In this context, since remedial actions will probably affect humans' lives even more directly, having both efficient and interpretable models will be all the more important to assist safety experts in their work.

Bibliography

[1] Hassan T Abdelwahab and Mohamed A Abdel-Aty. Development of artificial neural network models to predict driver injury severity in traffic accidents at signalized intersections. *Transportation Research Record*, 1746(1):6–13, 2001.

[2] Abdiansah Abdiansah and Retantyo Wardoyo. Time complexity analysis of support vector machines (svm) in libsvm. *International journal computer and application*, 128(3):28–34, 2015.

[3] Russell L Ackoff. Management misinformation systems. *Management science*, 14(4):B–147, 1967.

[4] Mohamed Ahmed, Helai Huang, Mohamed Abdel-Aty, and Bernardo Guevara. Exploring a bayesian hierarchical approach for developing safety performance functions for a mountainous freeway. *Accident Analysis & Prevention*, 43(4):1581–1589, 2011.

[5] Airbnb. New York city housing prices, 2019. Last accessed: 10 March 2022. URL: `https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data`.

[6] Ahmed M Alaa and Mihaela van der Schaar. Demystifying black-box models with symbolic metamodels. *Advances in Neural Information Processing Systems*, 32:11304–11314, 2019.

[7] Guilherme Seidyo Imai Aldeia and Fabrício Olivetti de França. Measuring feature importance of symbolic regression models using partial effects. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 750–758, 2021.

[8] American Association of State Highway Transportation Professionnals (AASHTO). The highway safety manual, 2010. Last accessed: 07 April 2022. URL: `http://www.highwaysafetymanual.org/`.

[9] APRR. Cartography of the highway network, 2022. Last accessed: 07 April 2022. URL: `https://voyage.aprr.fr/nos-autoroutes`.

[10] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[11] Douglas Adriano Augusto and Helio JC Barbosa. Symbolic regression via genetic programming. In *Proceedings. Vol. 1. Sixth Brazilian Symposium on Neural Networks*, pages 173–178. IEEE, 2000.

[12] Pranjal Awasthi and Reza Zadeh. Supervised clustering. *Advances in neural information processing systems*, 23, 2010.

[13] Nathaniel Beck, Gary King, and Langche Zeng. Improving quantitative studies of international conflict: A conjecture. *American Political science review*, 94(1):21–35, 2000.

[14] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889*, 2021.

[15] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[16] Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001.

[17] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.

[18] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006.

[19] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[20] Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How interpretable and trustworthy are gams? In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 95–105, 2021.

[21] Li-Yen Chang. Analysis of freeway accident frequencies: negative binomial regression versus artificial neural network. *Safety science*, 43(8):541–557, 2005.

[22] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[23] Shu-Heng Chen. *Genetic algorithms and genetic programming in computational finance*. Springer Science & Business Media, 2012.

[24] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[25] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[26] Daniel Delahaye, Supatcha Chaimatanan, and Marcel Mongeau. Simulated annealing: From basics to applications. In *Handbook of metaheuristics*, pages 1–35. Springer, 2019.

[27] Markus Deublein, Matthias Schubert, Bryan T Adey, and Borja García de Soto. A bayesian network model to predict accidents on swiss highways. *Infrastructure Asset Management*, 2(4):145–158, 2015.

[28] Markus Deublein, Matthias Schubert, Bryan T Adey, Jochen Köhler, and Michael H Faber. Prediction of road accidents: A bayesian hierarchical approach. *Accident Analysis & Prevention*, 51:274–291, 2013.

[29] Ni Dong, Helai Huang, and Liang Zheng. Support vector machine in crash prediction at the level of traffic analysis zones: assessing the spatial proximity effects. *Accident Analysis & Prevention*, 82:192–198, 2015.

[30] Harris Drucker, Christopher J Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9, 1996.

[31] Karim El-Basyouny and Tarek Sayed. Comparison of two negative binomial regression techniques in developing accident prediction models. *Transportation Research Record*, 1950(1):9–16, 2006.

[32] Yavuz Eren, İbrahim B. Küçükdemiral, and İlker Üstoğlu. Chapter 2 - introduction to optimization. In Ozan Erdinç, editor, *Optimization in Renewable Energy Systems*, pages 27–74. Butterworth-Heinemann, 2017.

[33] Ludwig Fahrmeir and Stefan Lang. Bayesian inference for generalized additive mixed models based on markov random field priors. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 50(2):201–220, 2001.

[34] Lee Fawcett, Neil Thorpe, Joseph Matthews, and Karsten Kremer. A novel bayesian hierarchical model for road safety hotspot prediction. *Accident Analysis & Prevention*, 99:262–271, 2017.

[35] Luis Firinguetti and Gladys Bobadilla. Asymptotic confidence intervals in ridge regression based on the edgeworth expansion. *Statistical Papers*, 52(2):287–307, 2011.

[36] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[37] Scott Fortmann-Roe. Understanding the bias-variance tradeoff, 2012. Last accessed: 20 April 2022. URL: `http://scott.fortmann-roe.com/docs/BiasVariance.html`.

[38] French Road Safety Observatory. Road safety annual report, 2020. Last accessed: 10 March 2022. URL: `https://www.onisr.securite-routiere.gouv.fr/etat-de-l-insecurite-routiere/bilans-annuels-de-la-securite-routiere/bilan-2019-de-la-securite-routiere`.

[39] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[40] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

[41] Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of lime. In *International Conference on Artificial Intelligence and Statistics*, pages 1287–1296. PMLR, 2020.

[42] Andrew Gelman. Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper). *Bayesian analysis*, 1(3):515–534, 2006.

[43] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[44] Andrew Gelman and Jennifer Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2006.

[45] Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[46] Maryam Amir Haeri, Mohammad Mehdi Ebadzadeh, and Gianluigi Folino. Statistical genetic programming for symbolic regression. *Applied Soft Computing*, 60:447–469, 2017.

[47] Jens Hainmueller and Chad Hazlett. Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach. *Political Analysis*, 22(2):143–168, 2014.

[48] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[49] Trevor J Hastie and Robert J Tibshirani. *Generalized additive models*. Routledge, 2017.

[50] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[51] Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

[52] Giles Hooker and Lucas Mentch. Please stop permuting features: An explanation and alternatives. *arXiv preprint arXiv:1905.03151*, 2019.

[53] Helai Huang, Qiang Zeng, Xin Pei, SC Wong, and Pengpeng Xu. Predicting crash frequency using an optimised radial basis function neural network model. *Transportmetrica A: transport science*, 12(4):330–345, 2016.

[54] Alexey Grigorevich Ivakhnenko. Polynomial theory of complex systems. *IEEE transactions on Systems, Man, and Cybernetics*, (4):364–378, 1971.

[55] Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. Bayesian symbolic regression. *arXiv preprint arXiv:1910.08892*, 2019.

[56] Per Johansson. Speed limitation and motorway casualties: a time series count data regression approach. *Accident Analysis & Prevention*, 28(1):73–87, 1996.

[57] Andrew P Jones and Stig H Jørgensen. The use of multilevel models for the prediction of road accident outcomes. *Accident Analysis & Prevention*, 35(1):59–69, 2003.

[58] Bryan Jones, Lester Janssen, and Fred Mannering. Analysis of the frequency and duration of freeway accidents in seattle. *Accident Analysis & Prevention*, 23(4):239–255, 1991.

[59] Sarath C Joshua and Nicholas J Garber. Estimating truck accident rate and involvements using linear and poisson regression models. *Transportation planning and Technology*, 15(1):41–58, 1990.

[60] Surya Mattu Julia Angwin, Jeff Larson and Lauren Kirchner. Machine bias, 2016. Last accessed: 29 April 2022. URL: `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`.

[61] Kaggle. The state of ml and data science, 2021. Last accessed: 25 March 2022. URL: `https://www.kaggle.com/kaggle-survey-2021`.

[62] Maarten Keijzer. Improving symbolic regression with interval arithmetic and linear scaling. In *European Conference on Genetic Programming*, pages 70–82. Springer, 2003.

[63] Kevin Lane Keller and Richard Staelin. Effects of quality and quantity of information on decision effectiveness. *Journal of consumer research*, 14(2):200–213, 1987.

[64] Young-Ju Kim and Chong Gu. Smoothing spline gaussian regression: more scalable computation via efficient approximation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(2):337–356, 2004.

[65] Young-Jun Kweon and Kara M Kockelman. Safety effects of speed limit changes: Use of panel models, including speed, use, and design variables. *Transportation Research Record*, 1908(1):148–158, 2005.

[66] William La Cava, Kourosh Danai, Lee Spector, Paul Fleming, Alan Wright, and Matthew Lackner. Automatic identification of wind turbine models using evolutionary multiobjective optimization. *Renewable Energy*, 87:892–902, 2016.

[67] William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabrício Olivetti de França, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H Moore. Contemporary symbolic regression methods and their relative performance. *arXiv preprint arXiv:2107.14351*, 2021.

[68] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 131–138, 2019.

[69] Brett Lantz. Medical cost personal dataset, 2013. Last accessed: 10 March 2022. URL: `https://github.com/stedy/Machine-Learning-with-R-datasets`.

[70] Xiugang Li, Dominique Lord, Yunlong Zhang, and Yuanchang Xie. Predicting motor vehicle crashes using support vector machine models. *Accident Analysis & Prevention*, 40(4):1611–1618, 2008.

[71] Zhibin Li, Pan Liu, Wei Wang, and Chengcheng Xu. Using support vector machine models for crash injury severity analysis. *Accident Analysis & Prevention*, 45:478–486, 2012.

[72] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[73] Dominique Lord and Pei-Fen Kuo. Examining the effects of site selection criteria for evaluating the effectiveness of traffic safety countermeasures. *Accident Analysis & Prevention*, 47:52–63, 2012.

[74] Dominique Lord, Abdelaziz Manar, and Anna Vizioli. Modeling crash-flow-density and crash-flow-v/c ratio relationships for rural and urban freeway segments. *Accident Analysis & Prevention*, 37(1):185–199, 2005.

[75] Dominique Lord and Fred Mannering. The statistical analysis of crash-frequency data: A review and assessment of methodological alternatives. *Transportation research part A: policy and practice*, 44(5):291–305, 2010.

[76] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158, 2012.

[77] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

[78] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

[79] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[80] Peter McCullagh and John A Nelder. *Generalized linear models*. Routledge, 2019.

[81] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[82] BD McCullough and HD Vinod. Implementing the double bootstrap. *Computational Economics*, 12(1):79–95, 1998.

[83] Richard McElreath. *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC, 2020.

[84] Ben McKay, Mark J Willis, and Geoffrey W Barton. Using a tree structured genetic algorithm to perform symbolic regression. In *First international conference on genetic algorithms in engineering systems: innovations and applications*, pages 487–492. IET, 1995.

[85] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre GR Day, Clint Richardson, Charles K Fisher, and David J Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics reports*, 810:1–124, 2019.

[86] Andreas Meier, Mark Gonter, and Rudolf Kruse. Symbolic regression for precrash accident severity prediction. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 133–144. Springer, 2014.

[87] Aaron Meurer, Christopher P Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K Moore, Sartaj Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, 2017.

[88] Shaw-Pin Miaou. The relationship between truck accidents and geometric design of road sections: Poisson versus negative binomial regressions. *Accident Analysis & Prevention*, 26(4):471–482, 1994.

[89] Shaw-Pin Miaou and Dominique Lord. Modeling traffic crash-flow relationships for intersections: dispersion parameter, functional form, and bayes versus empirical bayes methods. *Transportation Research Record*, 1840(1):31–40, 2003.

[90] Adriana-Simona Mihaita, Zheyuan Liu, Chen Cai, and Marian-Andrei Rizoiu. Arterial incident duration prediction using a bi-level framework of extreme gradient-tree boosting. *arXiv preprint arXiv:1905.12254*, 2019.

[91] Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022. URL: `https://christophm.github.io/interpretable-ml-book`.

[92] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

[93] Amir Bahador Parsa, Ali Movahedi, Homa Taghipour, Sybil Derrible, and Abolfazl Kouros Mohammadian. Toward safer highways, application of xgboost and shap for real-time accident detection and feature analysis. *Accident Analysis & Prevention*, 136:105405, 2020.

[94] Alina Patelli, Victoria Lush, Aniko Ekart, and Elisabeth Ilie-Zudor. Traffic modelling and prediction via symbolic regression on road sensor data. *arXiv preprint arXiv:2002.06095*, 2020.

[95] Judea Pearl. *Causality*. Cambridge university press, 2009.

[96] 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*, 12:2825–2830, 2011.

[97] Brenden K Petersen, Mikel Landajuela Larma, T Nathan Mundhenk, Claudio P Santiago, Soo K Kim, and Joanne T Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*, 2019.

[98] Oskar Pfungst. *Clever Hans:(the horse of Mr. Von Osten.) a contribution to experimental animal and human psychology*. Holt, Rinehart and Winston, 1911.

[99] Forough Poursabzi-Sangdeh, Daniel G Goldstein, Jake M Hofman, Jennifer Wortman Wortman Vaughan, and Hanna Wallach. Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–52, 2021.

[100] Divish Rengasamy, Jimiama M Mase, Mercedes Torres Torres, Benjamin Rothwell, David A Winkler, and Grazziela P Figueredo. Mechanistic interpretation

of machine learning inference: A fuzzy feature importance fusion approach. *arXiv preprint arXiv:2110.11713*, 2021.

[101] Divish Rengasamy, Benjamin C Rothwell, and Grazziela P Figueredo. Towards a more reliable interpretation of machine learning outputs for safety-critical systems using feature importance fusion. *Applied Sciences*, 11(24):11854, 2021.

[102] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[103] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[104] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[105] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–85, 2022.

[106] Cynthia Rudin, Caroline Wang, and Beau Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv preprint arXiv:1811.00731*, 2018.

[107] Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392, 2009.

[108] John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55, 2016.

[109] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[110] Lesia Semenova, Cynthia Rudin, and Ronald Parr. A study in rashomon curves and volumes: A new perspective on generalization and model simplicity in machine learning. *arXiv preprint arXiv:1908.01755*, 2019.

[111] Venkataraman N Shankar, Richard B Albin, John C Milton, and Fred L Mannering. Evaluating median crossover likelihoods with clustered accident counts:

An empirical inquiry using the random effects negative binomial model. *Transportation Research Record*, 1635(1):44–48, 1998.

[112] LS Shapley. Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, page 343, 1953.

[113] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.

[114] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

[115] Kevin I Smith, Richard M Everson, and Jonathan E Fieldsend. Dominance measures for multi-objective simulated annealing. In *Proceedings of the 2004 congress on evolutionary computation (IEEE Cat. No. 04TH8753)*, volume 1, pages 23–30. IEEE, 2004.

[116] Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.

[117] Stan development team. Stan: A c++ library for probability and sampling, version 2.5, 2015. Last accessed: 02 May 2022. URL: `https://mc-stan.org/`.

[118] Karolina Stanislawska, Krzysztof Krawiec, and Zbigniew W Kundzewicz. Modeling global temperature changes with genetic programming. *Computers & Mathematics with Applications*, 64(12):3717–3728, 2012.

[119] Erwin Stinstra, Gijs Rennen, and Geert Teeuwen. Metamodeling by symbolic regression and pareto simulated annealing. *Structural and Multidisciplinary Optimization*, 35(4):315–326, 2008.

[120] Sarah Tan, Rich Caruana, Giles Hooker, Paul Koch, and Albert Gordo. Learning global additive explanations for neural nets using model distillation. 2018.

[121] Robert L Thorndike. Who belongs in the family. In *Psychometrika*. Citeseer, 1953.

[122] Gerhard Tutz and Harald Binder. Generalized additive modeling with implicit variable selection by likelihood-based boosting. *Biometrics*, 62(4):961–971, 2006.

[123] Silviu-Marian Udrescu, Andrew Tan, Jiahai Feng, Orisvaldo Neto, Tailin Wu, and Max Tegmark. Ai feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity. *arXiv preprint arXiv:2006.10782*, 2020.

[124] Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020.

[125] Mojtaba Valipour, Bowen You, Maysum Panju, and Ali Ghodsi. Symbolicgpt: A generative transformer model for symbolic regression. *arXiv preprint arXiv:2106.14131*, 2021.

[126] Thomas Veran, Pierre-Edouard Portier, and François Fouquet. Crash prediction for a french highway network with an xai-informed bayesian hierarchical model. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1256–1265. IEEE, 2020.

[127] Thomas Veran, Pierre-Edouard Portier, and François Fouquet. Interpretable hierarchical symbolic regression for safety-critical systems with an application to highway crash prediction. *Engineering Applications of Artificial Intelligence*, 117:105534, 2023. URL: `https://www.sciencedirect.com/science/article/pii/S0952197622005243`, `doi:https://doi.org/10.1016/j.engappai.2022.105534`.

[128] Hrishikesh D Vinod. Double bootstrap for shrinkage estimators. *Journal of Econometrics*, 68(2):287–302, 1995.

[129] Marco Virgolin, Tanja Alderliesten, Cees Witteveen, and Peter AN Bosman. Improving model-based genetic programming for symbolic regression of small expressions. *Evolutionary computation*, 29(2):211–237, 2021.

[130] Christian Walck et al. Hand-book on statistical distributions for experimentalists. *University of Stockholm*, 10:96–01, 2007.

[131] Dennis L Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3):408–421, 1972.

[132] World Health Organization. Road traffic injuries, 2021. Last accessed: 10 March 2022. URL: `https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries`.

[133] Yuanchang Xie, Dominique Lord, and Yunlong Zhang. Predicting motor vehicle collisions using bayesian neural network models: An empirical analysis. *Accident Analysis & Prevention*, 39(5):922–933, 2007.

FOLIO ADMINISTRATIF

THESE DE L'INSA LYON, MEMBRE DE L'UNIVERSITE DE LYON

NOM : VERAN DATE de SOUTENANCE : 04/11/2022

Prénoms : Thomas, Marcel, Simon, Nicolas

TITRE : Efficient and interpretable crash prediction models : an application to a French highway network

NATURE : Doctorat Numéro d'ordre : 2022ISAL0096

Ecole doctorale : INFOMATHS

Spécialité : Informatique

RESUME :

Dans le monde entier, les accidents de la route ont des impacts sociaux et financiers importants. Pour réduire leur fréquence et leur gravité, les modèles de prédiction d'accidents (CPM) sont utilisés pour identifier les segments de route dangereux et fournir des indices exploitables sur les facteurs de risque associés. Les CPM sont soit des modèles statistiques paramétriques, en particulier des modèles linéaires généralisés (GLM), soit des modèles d'apprentissage automatique avec un nombre important de paramètres sans estimation d'incertitude associée (e.g., ensemble d'arbres de décision, machine à vecteurs de support…). Les modèles paramétriques simples ont tendance à être plus interprétables mais moins performants que les modèles non paramétriques très flexibles qui fonctionnent comme des boîtes noires. Lorsqu'ils réfléchissent à des décisions à fort enjeu, comme dans le contexte de la sécurité routière, les experts métier s'attendent à ce que les modèles prédictifs soient à la fois performants et interprétables. En effet, les modèles doivent les aider à concevoir et à déployer des actions de sécurité préventives ou correctives.
Dans ces travaux, nous contribuons à améliorer les performances prédictives des modèles paramétriques tout en conservant leur interprétabilité. En premier lieu, une structure hiérarchique bien choisie peut gérer les corrélations entre groupes d'observations et améliorer significativement la qualité des prédictions des modèles et leur interprétation. Nous proposons de l'apprendre en exploitant le résultat d'une méthode d'interprétabilité post-hoc (viz., SHAP) appliquée à un modèle boîte noire flexible (viz., XGBoost). Puis, nous introduisons deux approches algorithmiques (viz., un réseau de neurones polynomial, et une extension de la régression symbolique multi-objectif) pour découvrir des transformations non linéaires des variables explicatives. Ces dernières, tout en restant simple (e.g. interactions de premier ordre, logarithme), permettent aux modèles de capturer une plus grande partie de la variabilité dans la variable dépendante. De plus, avec la régression symbolique multi-objectif, nous calculons un classement spécifique à chaque cluster des expansions de modèles linéaires régularisés ordonnés par complexité croissante. Cela facilite un processus d'interprétation dynamique qui permet de découvrir des modèles prédictifs efficaces, efficients et interprétables.
Des expériences ont été menées sur un jeu de données de sécurité routière et sur plus de dix jeux de données publics couvrant des problèmes de classification et de régression variés. Les résultats obtenus sont prometteurs étant donné que nos deux contributions surpassent les modèles interprétables traditionnels et se rapprochent des meilleurs modèles non paramétriques boîtes noires. Enfin, nous illustrons les bénéfices de notre approche en présentant, sur une étude réelle de cas, une application que nous avons conçue pour les experts de la sécurité routière.

MOTS-CLÉS : Machine Learning, model interpretability, SHAP, Bayesian hierarchical modeling, symbolic regression, multi-objective optimization

Laboratoire (s) de recherche : LIRIS

Directeur de thèse : Prof. Dr. Jean-Marc PETIT

Président de jury : Prof. Dr. Florence SEDES

Composition du jury : Prof. Dr. Pierre GANCARSKI, Prof. Dr. Gabriele GIANINI, Prof. Dr. Florence SEDES, Prof. Dr. Julien JACQUES, Prof. Dr. Jean-Marc PETIT, Dr. Pierre-Edouard PORTIER