UCLouvain

FACULTE DES SCIENCES

Faculté des sciences

XAI : A Comparative Study of Post-hoc and Intrinsic Explainability Methods for Categorical Tabular Data

Author: **Siméon Godfrin**
Supervisors: **Olivier Caelen, Corentin Vande Kerckhove, Robin Van Oirbeek**
Reader: **Marco Saerens**
Academic year 2025–2026
Master [120] en science des données, orientation statistique

Remerciements Je remercie la LSBA et tous ces professeurs pour la formation reçue.
Je remercie mes promoteurs : Monsieur Rovin Van Oirbeek pour le soutient et la planification de cette tâche; Monsieur Olivier Caelen pour la discipline quotidienne et Monsieur Corentin Vande Kerckhove pour l'accès au donnée.
Je remercie Monsieur Marco Saerens pour la découverte de cette branche du Machine learning qu'est la XAI.

Je remercie ma copine pour le soutient et Guillaume Haquin pour la relecture.

Contents

Chapter 1

Introduction

1.1 Contextualization

This Master's thesis focuses on eXplainable Artificial Intelligence (XAI) and aims to compare several explainability methods within the framework of a multi-class prediction task applied to an almost fully categorical dataset. The dataset consists of a survey on the habits of Quebec residents conducted in 2022. One of the primary objectives of collecting this dataset was to enable the development of a tool allowing individuals to position themselves on the political spectrum solely based on their lifestyle and demographic features (e.g., age, income). This represents a challenging and relatively uncommon case on which to test both the explainability methods and the comparison benchmark on a real-world dataset.

A central challenge of this task lies in defining criteria that allow for a meaningful comparison between fundamentally different explainability techniques. This difficulty arises from several factors, including diversity in explanation techniques, variability in explanation outputs, dependence on data type, and the lack of consensus on what constitutes a good explanation in the Machine Learning (ML) field. To address this challenge, a structured comparison framework is proposed in Section 5, largely inspired by the functionally grounded evaluation approaches described in [14] and [35]. Those scientific articles provide answers to the question of how to evaluate and compare XAI methods in a systematic way, based on the Ethics Guidelines for Trustworthy AI published by the European Union's High Level Expert Group (AI HLEG) [20].

This work begins by introducing XAI and the accuracy and interpretability tradeoff in section 1.2. Section 2 presents the datasets and the predictive model, a Neural Network (NN), used as a black-box model to be explained. Section 3 defines the theoretical foundation of the three post-hoc explainability techniques used — SHAP, LIME and Counterfactuals — while Section 4 focuses on the intrinsically interpretable model used — Optimal Sparse Decision Tree (OSDT). Those techniques were chosen because they suit the nature of the data and the task at hand. Moreover, the three post-hoc techniques are commonly used and I was interested in understanding them more deeply and to see how they compare to an intrinsically interpretable model and to each other's. The section 5 defines the characteristics providing the formal definition of a XAI's method desired properties, on which we are going to compare the 4 approaches. The section 6 discusses the findings and concludes the thesis.

1.2 Organization

1.2.1 Defintion of XAI

Following DARPA (Defense Advanced Research Projects Agency), one of the earliest actor of the field, XAI can be defined at aiming "to produce more explainable models, while maintaining a high level of learning performance (prediction accuracy) [...].". This field resurfaced in response to the prohiminent use of Neural Network (and DNN), the need to explain those models and the increasingly more complex datasets. This is still an ongoing and recent research field without a well established consensus about the taxonomy. The term “explainable” and “interpretable” are either used interchangeably or defined as two different concepts. In the following work, one will use them interchangeably but the term “interpretable/interpretability” will mostly be used following the current ruling taxonomy in the scientific and ML community. [5] Attempts have been done to establish a taxonomy but one finds it hard to bend to since most online papers do not follow it yet.

The interpretability and accuracy trade-off

One common idea in the ML world is that there exists a tradeoff between interpretability and accuracy [6, 5, 22] :

- On one hand, simple models— such as linear and logistic regression (LR), decision tree (DT) — are intrinsincaly interpretable, given their construction, but are deemed less accurate and less fit to model complex real-life problems.
- On the other hand, Deep Neural Networks (DNNs) models yield a higher accuracy “at the cost” of interpretability. (By increasing the model’s complexity).

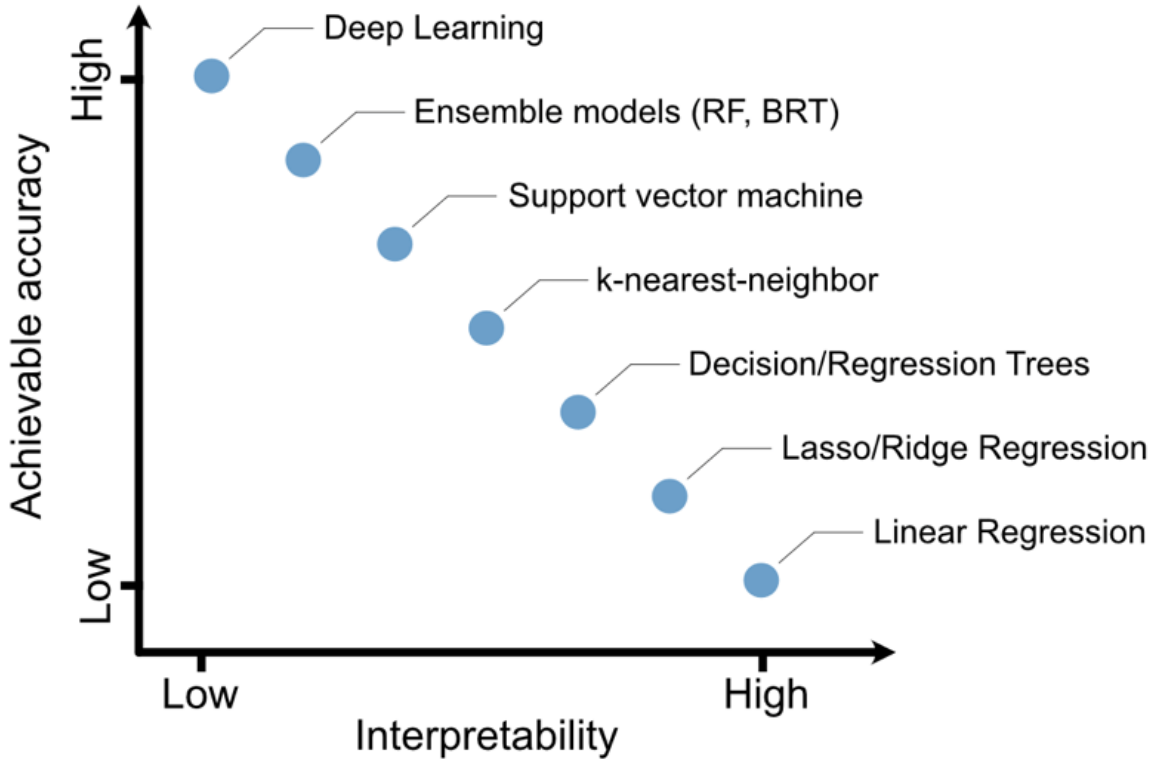


Figure 1.1: The accuracy-interpretability tradeoff : as the accuracy increases, the interpretability decreases. Source : [40]

The belief in this tradeoff partly emerged from older sources such as Leo Breiman, who introduced the Rashomon effect in statistics [13].

He asserted, more than 20 years ago, the superiority of accuracy over interpretability for statistical models.

This belief isn't simply an old idea, it is still present in the recent literature. Johansson et al.(2011) [28] write "Models exhibiting [high accuracy] are many times more complex and opaque, while interpretable models may lack the necessary accuracy." In their study of 16 biomedical classification tasks, black-box methods (like ensembles or SVMs) generally obtained higher accuracies than transparent models. This resonates with the intuition that a sufficiently complex model can fit intricate, non-linear relationships in data that a simpler model cannot capture. Those two examples might give a piece of explanation of the current inertia and tendency in the ML community ***to reach for complexity first***, motivated by the desire of maximizing accuracy in a world where data is increasingly complex.
Nonetheless, the belief into this tradeoff, or not, should not limit one to consider intrinsically interpretable models when the context allows it.

The existence of this tradeoff as general rule is debated, as it appears to be based more on intuition and heuristic than in solid empirical evidence. Most recent research tends to align with a circumstantial existence of the trade-off. Some qualify it as a "myth" [46], while others found that when taking the user's interpretation into account that "there are contexts where black-box model can be more explainable and less confusing". [11]

The performance gap between complex and simple models exists but is not always large. Johansson et al.(2011) [28] found that, although black-box models tended to be best, the accuracy difference was typically modest — suggesting the trade-off might be a gentle slope rather than a steep cliff.

In case of tabular data, the gap in performance between interpretable and black-box model tends to be lower.[22], [46] Neural networks, in this framework, generally find no decisive advantage. It has been known for a very long time that very simple models perform surprisingly well for tabular data [25]. And this finding resurfaces in recent research.
For example, research in finance and medicine have noted that simpler models (like logistic regression or decision trees) often come within a few percentage points of the accuracy of state-of-the-art algorithms. [46]
Goethals, Martens & Evgeniou (2022) [22] analysed 90 benchmark classification tabular datasets and finds that the "trade-off exists for most (69%) of the datasets, but that somewhat surprisingly for the majority of cases it is rather small while for only a few it [the trade-off] is very large". They found that the gap in accuracy between black box and white box models is correlated to dataset's characteristics such as how difficult it is to linearly separate it. The finding reinforces our belief in the superiority of black box models in capturing the non-linearities. It also opens the door to a better identification of the conditions as to *when* using which type of models.

In the case of *raw/unstructured data*, such as images, neural networks currently have an advantage over other approaches. "The difference in performance between comprehensible models compared to a black box ones such as DL is considered "unbridgeable." Goethals, Martens & Evgeniou (2022) [22]
Yet, even in these domains, hybrid approaches are emerging (e.g. prototype-based convolutional networks [15])

"These two data extremes show that in ML, the dichotomy between the accurate black box and the less-accurate interpretable model is false" [47], or at the very least, is much more nuanced. In certain contexts, i.e. tabular data, depending on the task at hand, we must not choose accuracy *or* interpretability — we can have *both*.
Therefore, in case of tabular data, a well-constructed transparent model *could* yield both an high accuracy and be intrinsically interpretable. [48] This idea is central, to the "Interpretable Machine Learning" ("IML" in short), which one will frame as sub-field within XAI.
Those findings serve to remind the ML practitioners to explore the space of simpler models before resorting to complex black-box models if the context suits it.

The Rashomon effect

In the context of tabular data, I have previously mentioned that interpretable models can match the performance of black-box models. In principle, if one searches the space of simpler models thoroughly, one may discover a model that performs comparably to its more complex counterpart.

This observation shifts the focus of model selection: if multiple models achieve near-identical accuracy, it becomes reasonable to prioritize other properties such as interpretability or fairness. In the direct continuity of this idea, the Rashomon set refers to the ML models achieving similar performance above a given threshold. [48] Research on the Rashomon set offers theoretical support for a practice often used heuristically in applied machine learning: selecting the simplest model that performs well. For a more formal treatment of the Rashomon set and its implications, see [50].

In conclusion, this trade-off is a central idea, a knot in the field, embodied by the Post-hoc vs Intrinsically interpretable models dichotomy. The belief in a strict trade-off between accuracy and interpretability may lead practitioners to assume a forced compromise. However, empirical studies [17, 25, 48] have shown that this trade-off is often circumstantial and depends on the specific data and task at hand. The Rashomon set offers a more flexible perspective: if many models perform equally well, at least one of them may be interpretable. This insight softens the perceived trade-off and aligns with what has been observed in practice.

1.2.2 The three axis

This work will revolve around the comparison of 4 explainability approaches — three post-hoc and one intrinsically interpretable model— examined across three dimensions inspired by [6] : i) Exploratory Data Analysis, ii) Model-Level analysis and iii) Comparisons of the approaches.
The comprehension and analysis of the data will be considered as an entire part of the explainability process since there is no understanding of the model if one does not understand the task at hand.
The three axis are the following :

- (I) Exploratory Data Analysis (EDA) : the comprehension and exploration of the raw data.
- (II) Models analysis : Definition of the post-hoc interpretability methods — SHAP, LIME, Counterfactuals— and of intrinsically interpretable model —Optimal Sparse Decision Tree (OSDT).
- (III) Comparison between the methods based on functionally-grounded evaluation, i.e. quantitative metrics independent of explainees.

Chapter 2

Exploratory Data Analysis, Preprocessing and Models

2.1 Description of the dataset

The dataset is composed of various categorical variables representing an individual across a wide spectrum through

- **Person's features** : Income, age, gender, language, ethnicity, education, dwelling, vehicle, sexual orientation, clothing type, city, place of birth.
- **Habits** : Sport, type of clothing store, means of transport, animal, film, music
- **Consumptions** : Alcohol, smoking habit, meat, coffee.
- **Activities** : Museums, Fishing, Hunting, Volunteering, Motorized Outdoor activities
- **Vote** : Intent of vote, prediction of the winning party, probability to vote.

The goal of collecting this informations was to enable a prediction, based on the life habits of a person, of the closest political party to its beliefs.
The variable concerning the vote intent of a person will be the response variable, and therefore central to our analysis, and all the others will be the explanatory variables.
The Quebecer (Quebecois) parties are the following :

- **CAQ (Coalition Avenir Québec)** : Center-right populist party emphasizing Quebec nationalism, pragmatic economic policies (fiscal responsibility, business-friendly), social conservatism, and skepticism of multiculturalism.
- **QS (Québec solidaire)** : Left-wing, sovereigntist party committed to social justice, environmentalism, feminism, participatory democracy and progressive politics; often more radical in social policy than the mainstream sovereignty parties.
- **PLQ (Parti libéral du Québec/Québec Liberal Party)** Federalist, centrist party favoring liberal democracy, bilingualism, and economic development; tends to support more business-oriented policy and less emphasis on sovereignty compared to PQ or QS.
- **PQ (Parti Québécois)** : Sovereigntist party historically leaning centre-left, combining Quebec nationalism, promotion of French language and culture, social democracy, and the goal of achieving political independence from Canada.

- **PCQ (Parti conservateur du Québec)** : More right-leaning, emphasizing individual liberty, reduced taxation, smaller government, and conservative values; some aspects of economic liberalism and skepticism toward big government programs.

The following graphs showcase 2.1, on the left, the distribution of the response variable, the vote intention, called 'op_intent', and on the right, the distribution of the people's prediction of the winning party, called 'people_predict'.
The plots show a distortion of the reality between which party is expected to win and the most popular party among the questioned individuals : on the left plot, the most popular party is *"QS"* while on the right plot, the party the people see win the election is *"CAQ"*.

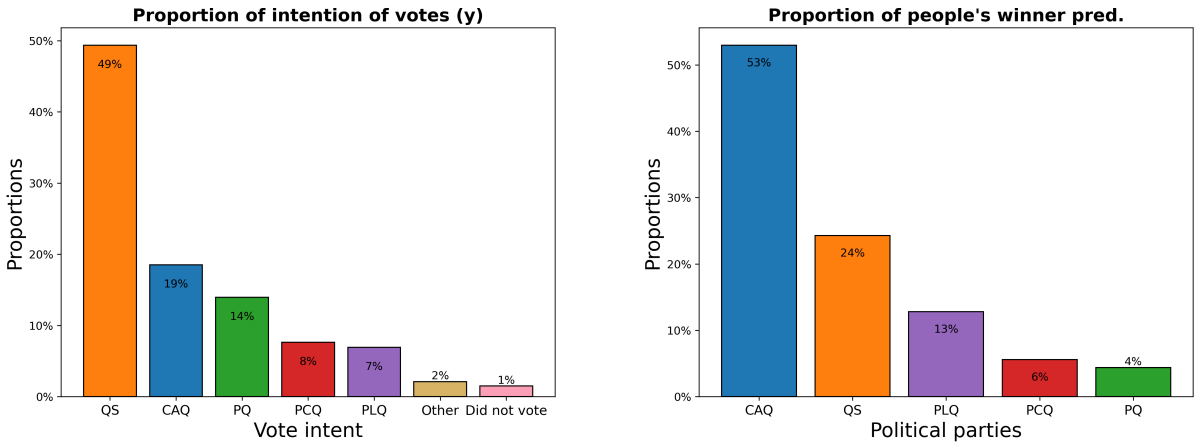


Figure 2.1: Comparison of the predictions and intentions of voters

Furthermore, this is a generalized trend as the next plot showcases 2.2. It represents the proportion of people predicting a winning party given their vote intentions. For instance, the first set of bars represents the proportion of people stating that *"CAQ"* will win given their vote intention, highlighted by their colors.
The graph showcase that among all voters, a majority of them believed that *"CAQ"* will win the elections.

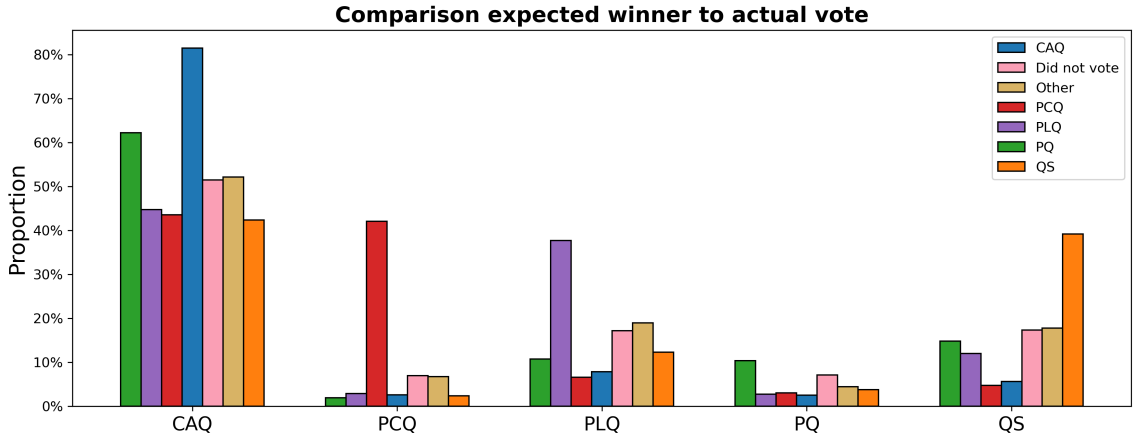


Figure 2.2: Expectation of winners

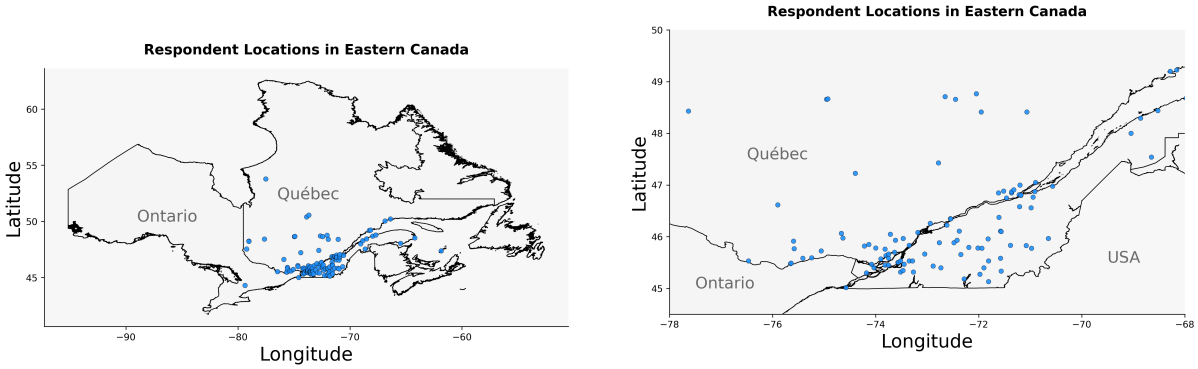
The table 2.1 shows the number behind the bars in the plot. The columns represent the people's predictions of the winner and the rows represent their intention of vote at the time of the poll. The rows sum up to 1.

Table 2.1

Voting intention	**CAQ**	**PCQ**	**PLQ**	**PQ**	**QS**
CAQ	81.5 %	2.55%	7.87%	2.48%	5.59%
Did not vote	51.45%	6.98%	17.15%	7.12%	17.3 %
Other	52.12%	6.72%	18.92%	4.45%	17.79%
PCQ	43.55%	42.11%	6.62%	3.01%	4.71%
PLQ	44.71%	2.91%	37.66%	2.72%	11.99%
PQ	62.2 %	1.88%	10.72%	10.38%	14.83%
QS	42.34%	2.38%	12.3 %	3.78%	39.2 %

It appears that a big portion of voters sees their party as the winning one. This knowledge created friction with the choice of taking or not the variable "people_predict", i.e. people's winning party predictions, which seemed, at first sight, correlated with my response variable "op_intent", i.e. the vote intent. The Cramer's V association between the two is actually 0.313, which means a moderate association between the two variables.

These results are normal once you take the geographic location of the poll into account : most interviewed people come from the small area where the *"CAQ"* is not favorite. This area is nonetheless the most densely populated. One can visualize those areas on the following figures. In Québec, most provinces end up with *"CAQ"* as the most popular party, except for the area around Montreal, where the majority of people were interviewed. [42]



(a) Zoomed-in view of Montreal area

Figure 2.3: Locations of voters

Here are a few general features of the dataset :

- 53 % of men, 44% of female and 2 % of other gender.
- 43% of the people live in detached houses, 23% lives in an apartment and 11% in condominium
- 91 % of the people practice physical activity.
- 84 % of the people never smoked and only 0.7% are daily smokers which is ten times less than the Canadian average of 8.2% of daily smokers.
- 12 % does not drink alcohol.
- 91 % of people are from white ethnicity.
- 49 % have 34 years or less, 36% have between 35 and 54 years old.

Figure 2.4 shows that the majority of respondents hold at least a bachelor's degree, with nearly 40% of the sample reporting this as their highest education level. This is followed by college graduates and master's degree holders. In Québec's education system, "college" typically refers to post-secondary programs, which precede university. Primary and secondary education are underrepresented in the sample, with few respondents lacking post-secondary training. This suggests the survey population is relatively well-educated, which may influence model predictions and generalizability. But this result is in line with the geographical location of the survey : a city with a high density of universities and colleges.

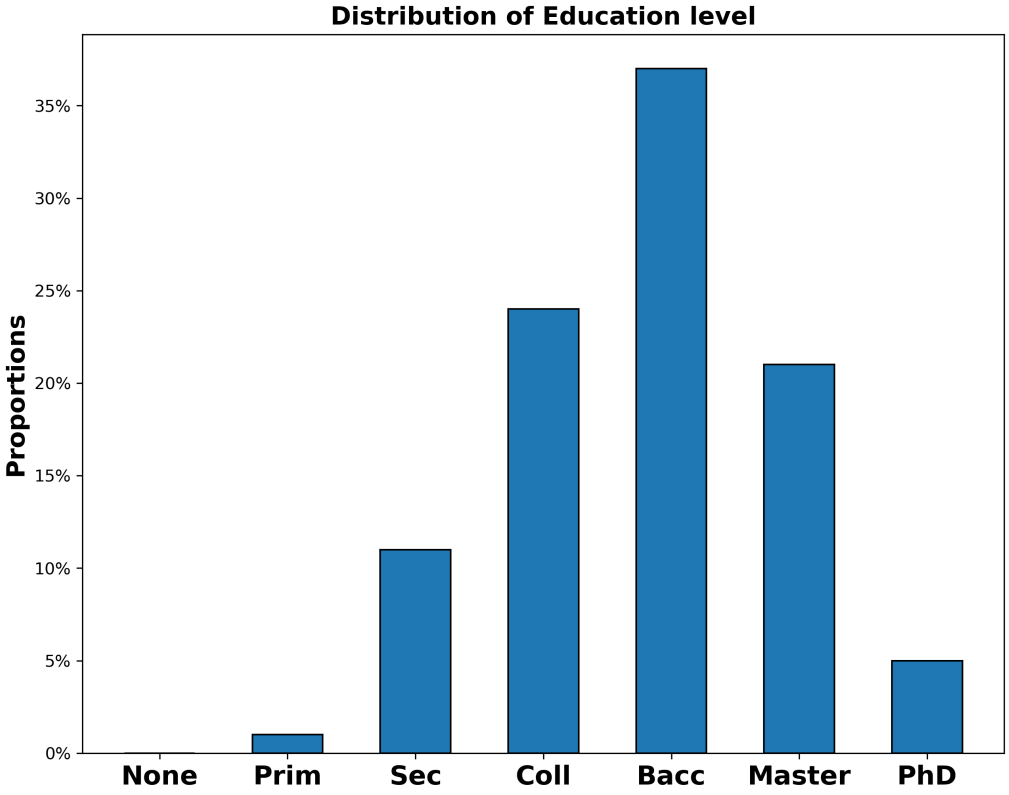


Figure 2.4: Proportion of the education levels

The distribution of the annual income level by household is shown in the figure 2.5 below. The middle class income are the most represented. The plot is not a power-law or heavily skewed distribution — which is somewhat unexpected for income, which is usually skews right. It shows a rather balanced distribution among the different income levels, where high income levels are over-represented compared to what is expected in the standard population.

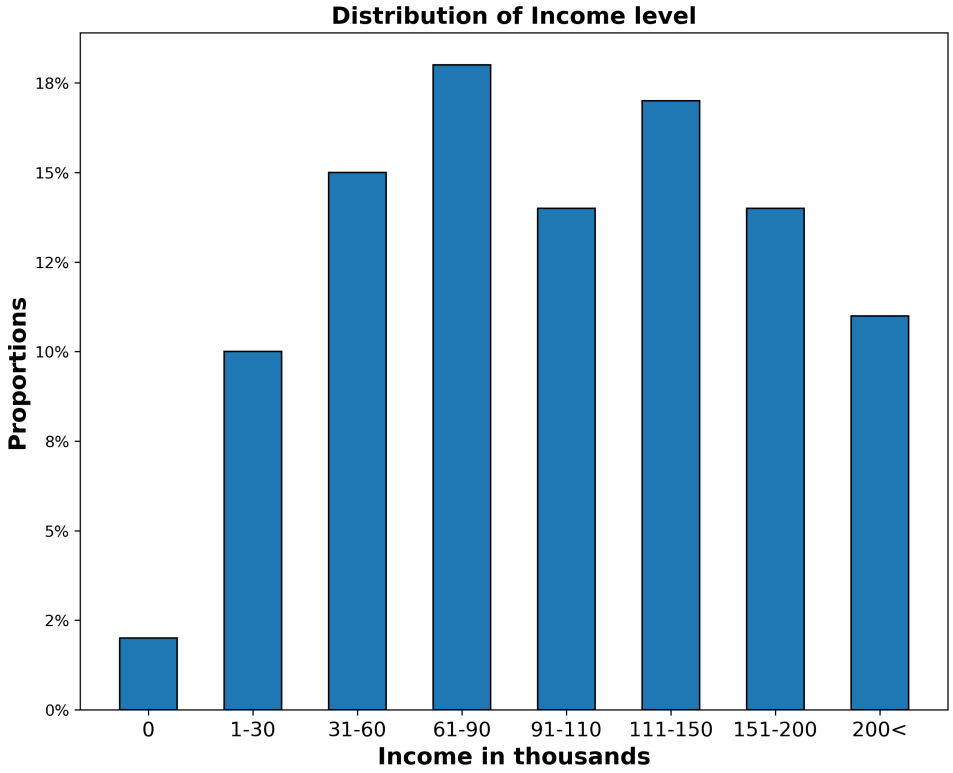


Figure 2.5: Proportion of the income levels

The distribution of the vote among different variable levels is surprisingly uniform. The figure 2.6 shows this effect upon the education levels. This finding appears to also be true for the income levels, the genders and the age group.
On the abscissa is vote intention and on the ordinate is the proportion that voted for this party given its education level.

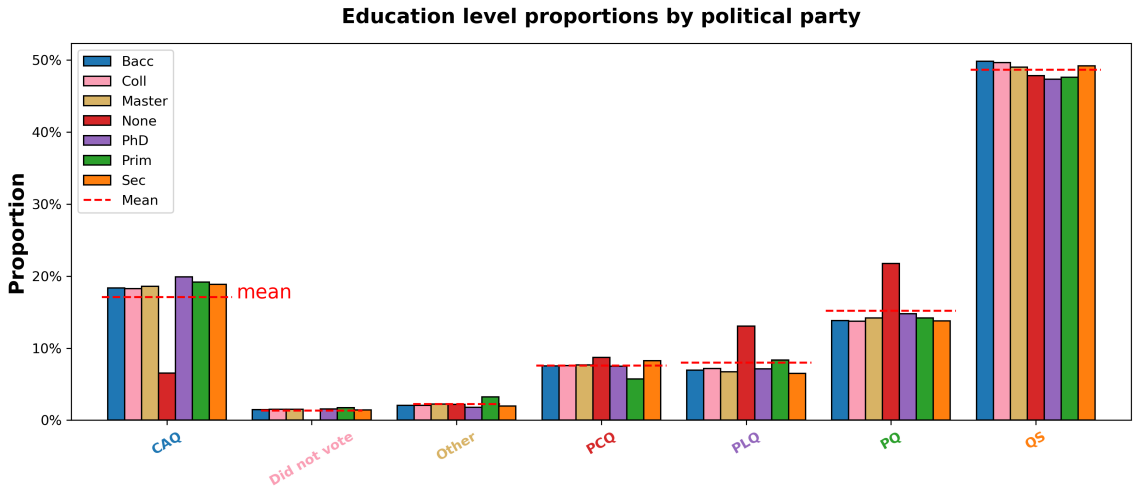


Figure 2.6: Prop. of vote intention by education levels

2.1.1 Summary of all variables

Variable	Description
day	Day of survey response.
cons_coffee	Usual place where the respondent gets coffee (e.g., Tim Hortons, Starbucks).
ses_income	Household income before taxes, categorized in income brackets.
ses_dwelling	Type of dwelling (e.g., apartment, detached house, condo).
ses_educ	Highest completed level of education (e.g., high school, college, university).
app_swag	Clothing style preference (e.g., chic, casual, punk).
music	Preferred genre of music consumption (e.g., rock, electro, etc.).
film	Preferred genre of film (e.g., drama, comedy, crime, etc.).
ses_ethn	Self-reported ethnicity (e.g., White, Black, Asian, Indigenous).
act_transport	Main means of transportation (e.g., car, public transit, walk).
vehicule	Type of car most frequently used (e.g., SUV, sedan, luxury).
cons_Smoke	Frequency of smoking cigarettes.
cons_meat	Frequency of meat consumption.
cons_brand	Preferred shopping type for clothes (e.g., chain stores, thrift stores).
animal	Presence and type of pets (e.g., cat, dog, none).
sport	Physical activity practiced most often (e.g., gym, yoga, walking).
alcohol	Usual type of alcoholic beverage consumed (e.g., wine, beer, none).
age	Age group of the respondent (e.g., under 34, 34–55, 55+).
lang	Main language spoken at home (French, English, or other).
people_predict	Prediction of the winning political party according to the respondent.
op_intent	Political party the respondent would vote for in a provincial election.
gender	Self-identified gender of the respondent (male, female, other).
sex_ori	Sexual orientation (heterosexual, bisexual, gay/lesbian, other).
VisitsMuseumsGaleries	Frequency of visiting museums or art galleries.
Fishing	Frequency of going fishing.
Hunting	Frequency of going hunting.
Motorized Outdoor	Frequency of doing motorized outdoor activities (e.g., snowmobiling).
Volunteering	Frequency of volunteering or community involvement.
voting_probability	Estimated probability of voting in an election.
pays_qc	Whether the respondent lives in Quebec.
immigrant	Whether the respondent was born outside Canada.
lat	Latitude coordinate of the respondent's location.
long	Longitude coordinate of the respondent's location.

Table 2.2: Description of variables used in the analysis.

To finalize this analysis, one inspected the correlations. As a measure of the strength of the associations between variables, one chooses to use Cramer's V. It is based on the Pearson's chi-squared statistics and gives a value between 0 and $+1$ to measure the association. It does not give any indication about its *direction*.
The figure 2.7 shows the result for our variables having the most associations. One can see that within the dataset, there is only weak (value between 0 and 0.3) and one moderate association, between *"people_predict"*, the party the person sees as winning the elections, and *"op_intent"*, the vote intentions of the individual, our variable of interest. This association was expected since people tend to see their own party as the winning one, as underlined by Figure 2.2.

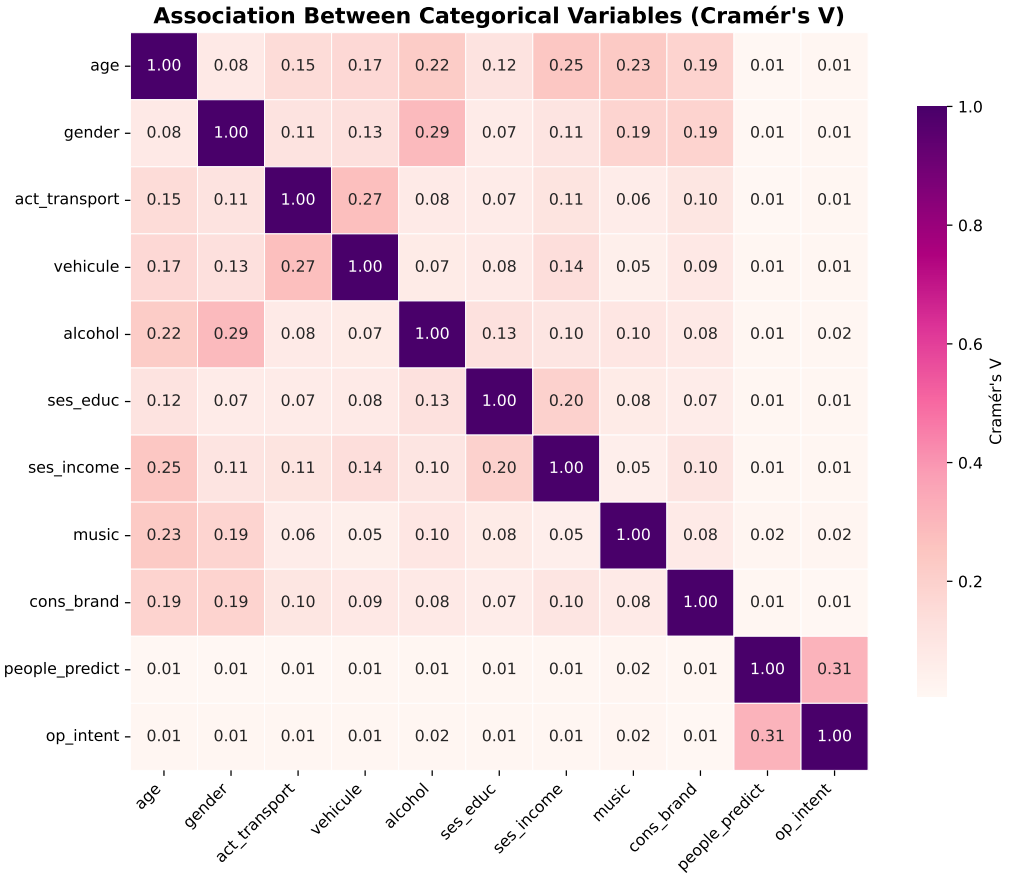


Figure 2.7: Cramer's V associations.

2.2 Preprocessing

2.2.1 Missing data

The initial dataset contained a substantial number of missing values across both input and output variables. To ensure data quality and prevent downstream issues in model training, I performed a series of steps to either impute or remove NaNs depending on their context, prevalence, and relevance. Missing outputs (e.g. 'op_intent') were excluded, while missing inputs were handled more selectively.
For instance, several variables related to health and well-being had extremely high proportions of missing values (over 72%). To avoid introducing bias through imputation or excessive row deletion, I chose to exclude these variables entirely.

2.2.2 Imputation model

The variable 'voting_probability', which represents the self-assigned probability of voting, had missing entries for some respondents (initially for 38% of them). Rather than discarding these rows and lose a significant chunk of my data, I applied a multivariate imputation strategy using 'IterativeImputer' with median initialization (from the package 'sklearn').
Iterative imputer trains a regression model based on the subset where my futurly imputed variable is not missing. Then, it predicts the missing values based on the learned weights and my features.
This approach preserved overall sample size while modeling missingness in a principled way.
The imputation model slightly under-represent rare occurring probability as you can see in the following plot. The **above plot** is a zoomed-in version of the **below plot** to better visualize the differences in distribution.

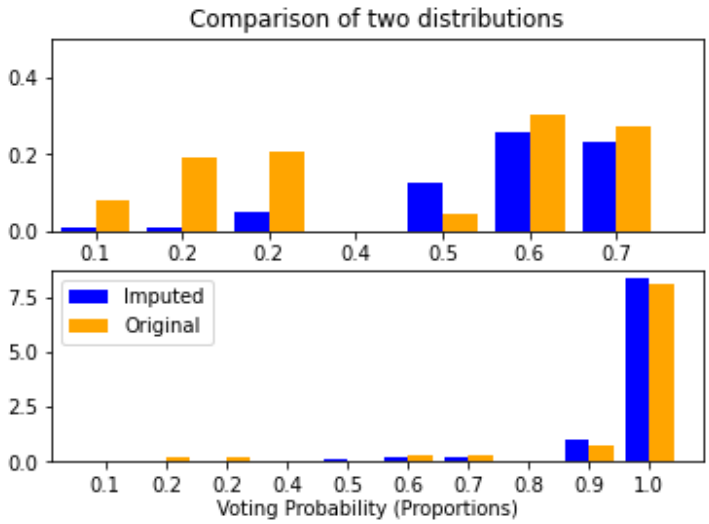


Figure 2.8: Comparison of the imputed and original distribution.

Although KNNImputer can handle non-linear patterns and skewed distributions, it was not used here due to the categorical nature of most features. The default Euclidean distance metric assumes numerical continuity and magnitude, which is not meaningful for categorical variables. Alternative distance measures like Hamming distance or the value difference metric (VDM) could better represent categorical similarity, but are not directly available in the default KNNImputer implementation in scikit-learn. I decided to not spend time on it.
That is why I opted for IterativeImputer, which models each feature as a function of others in a multivariate framework. This approach is more flexible and can better handle mixed-type datasets.
In addition to imputing missing values, a binary indicator variable ('imp_ind') was introduced to explicitly flag

whether a value was originally missing. This allows the predictive model to account for potential systematic effects associated with missingness.
Since the imputation procedure yields continuous values, the imputed probabilities were constrained to the valid interval $[0, 1]$, and discretized to one decimal to remain consistent with the original variable scale.

2.2.3 Geographical information

The dataset included postal codes and city names, which are discrete spatial features with high cardinality. To encode geographical information more efficiently and enable spatial reasoning, I mapped each location to its corresponding latitude and longitude. This choice avoided the need for large one-hot encoding and enabled geographic visualization and modeling (see figure 2.3). Since the range of latitude and longitude differed, I standardized them to align with neural network input expectations.

2.2.4 Categorical variables

Film and music preference variables were handled via derived indicators. For example, if respondents had no known genre or title associated with a music/film preference, I assumed non-response or non-interest, and added binary flags (‘music_no‘ , ‘film_no‘) to preserve this information without discarding rows.
Time variable was split into day, month and year components to increase modeling flexibility and to allow for potential time-dependent trends or seasonal patterns. Year was removed post-verification since all responses came from 2022.
Several features had a natural ordering (e.g. education level, income, Fishing frequency,...). These were encoded using an ordinal encoder (from the package ‘sklearn‘) to preserve this structure while maintaining numerical compatibility with downstream models.
To handle categorical variables without introducing multicollinearity, I used a reduced-rank one-hot encoding strategy ($(k-1)$ encoding). For each of the mutually exclusive binary indicators (e.g. gender, vehicule, meat consumption), one category was dropped. Problematic categories (e.g., overlapping or misaligned encodings) were cleaned manually based on logical assumptions.

2.2.5 Highly imbalanced datasets

My dependent variable is highly imbalanced. To counteract this feature, I used an oversampling techniques I compare the performance of RandomOverSampler and SMOTE (Synthetic Minority Oversampling technique).

- **The Random Over-sampler** simply randomly samples point with replacement from the minority classes till reaching a certain predefinetely fixed treshold number of samples per classes.
- **SMOTE** creates new samples based on interpolation. The techniques generates samples next to the original samples using a k-Nearest Neighbors classifier. In our case, the nearest neighbors search relies on the value difference metric (VDM). “A new sample is generated where each feature value corresponds to the most common category seen in the neighbors samples belonging to the same class.” [3]

I trained my model, a neural network, on both oversampled datasets and compared their accuracies on the same test set. The two techniques yield similar accuracies. Therefore, I chose to stick with the random over-sampler because it only relies on real data instances.
The proportions of each class was heuristically chosen based on various test. It results in a training set after oversampling where each class, except the one in majority was augmented by 33%, resulting in 5000 supplementary rows.

2.2.6 Response variable and people_predict variable

The variable ‘people_predict‘ could be seen as a leakage of information from the response variable ‘op_intent‘, since a major proportion of people tend to predict their own party as the winner. (See Figure 2.2). For this reason, I was initially hesitant to include it as a feature in my model. However, the moderate association (Cramer's V = 0.313) indicates that while related, they are not redundant. Including ‘people_predict‘ provides additional context on respondents' perceptions, which may capture factors beyond their stated vote intention. The inclusion of this variable alone led to a significant increase in the accuracy of my model.
Furthermore, the occurrence of "Other" and "Did not vote" were deleted from my response variable in order to reduce the number of low occurring classes and improve the model's accuracy.

2.2.7 Parallel dataset construction for method compatibility

To ensure compatibility, and the feasibility across all explainability methods and my model, two parralel representations of the dataset were constructed.

The first representation is a fully **encoded** dataset, where categorical variables are transformed into numerical formats compatible with neural networks and attribution-based explainers such as SHAP and LIME, i.e. one-hot encoding into $k-1$ categories and ordinal encoding.
The second representation was fully reconstructed to represent **the original categorical structure** of the data and is used by example-based method, counterfactual explanations using the DiCE algorithm, which requires semantically valid categorical values. Furthermore, this representation enables a more interpretable presentation of explanations in general, as the explanations refer to human-readable categories rather than encoded vectors.

Both datasets are derived from the same cleaned intermediate preprocessing stage and contain identical observations. The oversampling is only applied to the training encoded dataset to prevent data leakage. The test sets remain untouched in both representations for unbiased evaluation and total alignment of obervations. The distinction lies solely in the feature representation, not in the underlying data. This design allows each explainability method to operate in a representation consistent with its assumptions, while ensuring comparability across methods.

2.2.8 Observation alignment via absolute indexing

To ensure strict correspondence between the two dataset representations, an identifier, denoted as `absolute_index`, was introduced at an early stage of preprocessing.
The index uniquely identifies each respondent and is preserved throughout the entire pipeline, including encoding, imputation, shuffling, oversampling and dataset splitting. It serves as primary key to align the encoded and raw categorical datasets, as well as the response variables.
Consistency checks were systematically performed after each major transformation to verify that datasets remained perfectly aligned. This mechanism prevented silent indexing errors and guaranteed that individuals can be tracked and compared across all stages of analysis.

2.2.9 Resolution of inconsistent categorical encodings

During the encoding into $k-1$ categories to limit multicollinearity in the encoded dataset, certain categorical variables were found to violate the mutual exclusivity assumption instilled by one-hot encoding. In particular, the sexual orientation variables occasionally contained multiple active categories for a single individual. This is likely

to be an answer to a sub-category hidden behind the "Other" option.
To restore the consistency, these cases were corrected by enforcing a single active category per observation : the generic "other" was deactivated in favor of the more specific selected category.
While this step introduces explicit assumptions and modification of the raw data, it was necessary to ensure a consistent encoding. It concerned around 4.99% of the observations.

2.2.10 Final dataset size

After all the preprocessing steps minus the oversampling, out of the initial 64k observations, 71% of the data was retained.

2.3 The neural network model

2.3.1 Model role and setup

The predictive model used throughout this work is a feed-forward neural network trained on the fully encoded version of the dataset. The model is treated as a black-box predictor, and no interpretability constraints are imposed at training time.

The input consists exclusively of numerical features obtained after preprocessing, encoding and resampling. The output layer produces a probability distribution over the five remaining political parties using a softmax activation, and the model is trained using categorical cross-entropy loss.

2.3.2 Architecture

The final architecture is composed of a sequence of fully connected layers with varying widths and activation functions, where dropout layers are intertwined. The architecture is summarized as follows:
An input layers feeds into successive dense layers of sizes 256, 64, 256, 192, 224, 128, 96 and 96 neurons respectively. Activation functions include *tanh, selu, softplus* and *relu* selected empirically during hyperparameter tuning. Dropout is applied at multiple depths, with rates ranging from 20% to 50%, in order to limit overfitting and improve generalization.

The output layer consists of five neurons with a softmax activation, corresponding to the five target classes. A batch size of 512, which was empirically determined, was used for all experiments. A learning rate scheduler was employed to reduce the learning rate when the validation loss reached a plateau during training.

2.3.3 Hyperparameter optimization

Hyperparameter optimization was conducted using a two-phase strategy using Hyperband tuner.
Hyperband is a "resource-aware" hyperparameter optimization algorithm designed to efficiently explore large hyperparameter spaces while managing the allocation of computational resources. It builds on the idea that poor-performing configurations can often be identified early in training. The saved computational costs are relocated toward more promising model configurations.
The algorithm randomly samples configurations of hyperparameters and combines it with an aggresssive early-stopping strategy inspired by successive halving. It starts with a large number of configurations evaluated with a small computational budget (e.g., a few training epochs). At each stage, the algorithm only keeps the best performing configurations and it relocates the computational ressources of the abandoned configurations to the remaining ones, increasing their budget. This process is repeated among different brackets, leading in the end to the best performing model found across the, hopefully, wide hyperparameter space searched. In this work, the budget allocated was 150 epochs to the first Hyperband and 100 for the second one.

In a first phase, a Hyperband search was used to identify a suitable network architecture (number of layers, number of neurons per layers, activation functions per layers, dropout presence and rates, etc.)
Once the architecture was fixed, a second Hyperband procedure was applied to optimize the learning rate of the Adam optimizer. In this second phase, Hyperband explored learning rates values sampled on a scale between 10^{-4} and 10^{-1}. The objective function was the validation loss, computed on the test set.

2.3.4 Performance

To account for the instrinsic stochasticity of neural networks training, the final model was trained and evaluated 20 times on the same training and testing splits. Across the runs, the model achieved an average accuracy of approximately 61%. For comparions, the results of a previously trained model on this dataset was still attached to it. It yielded an accuracy of 52%.
While this comparison should be interpreted with caution, as implementation details (data splits, preprocessing steps, model, etc.) differ, it still gives a baseline performance and it suggests that the proposed modeling pipeline yields better results.
The model has the following architecture :

- Input Layer of size (174,)
- 256 units/neurons, activation tanh
- dropout layer, rate 0.2
- 64 neurons, selu
- 256 neurons, softplus
- dropout, rate 0.5
- 192 neurons, selu
- dropout, rate 0.4
- 224 neurons, relu
- dropout 0.4
- 128 neurons, relu
- 96 neurons, selu
- dropout, rate 0.3
- 96 neurons,selu
- Output 5 neurons, softmax

Chapter 3

Post-hoc

3.1 Introduction

Explainability techniques can be divided into four non-exclusive categories :

- **Scope-based** : refer to the "range" of the interpretability techniques, whether it is a ***local explainer***, which only explains a "specific decision or instance" or a ***global explainer***, which provides the reasoning for the whole dataset. Global model interpretability, for black-box models, is hard to achieve in practice.
- **Complexity-based** : Named after the beliefs in the questioned interpretability and accuracy tradeoff. In my opinion, it contains the core of XAI and the two opposite ways of explaining it. Either one uses techniques to explain a "black-box" model, this is called ***post-hoc*** interpretability. Or one builds an ***intrinsic interpretable model*** which is self-explanatory and has interpretability built right in.
- **Model-based** : A technique that is applicable to only one kind of model is deemed a ***model-specific*** method. (Technique made only for neural networks for instance). A technique that is applicable to any kind of model is called ***model-agnostic***. Model-agnostic interpretations are usually post-hoc.
- **Methodology-based** : ***backpropagation-based*** (also called gradient-based) methods may be used to back-propagate a significant signal from the output to the input. (i.e. Saliency maps). ***Perturbation-based*** algorithm uses techniques to change the feature set of a given input instance and investigate the impact of the changes on the output. (i.e. SHAP)

In this work, one will focus mostly on post-hoc methods. They can be grouped around three important characteristics : **(i)** Attribution methods, **(ii)** visualization methods, **(iii)** example-based explanation methods.

3.2 LIME

"Attribution methods calculate the attribution of a training instance feature directly by deleting, masking, or changing the input instance, then a forward pass on the modified input is executed before comparing the obtained results to the original output." To achieve this task, a sub-family of this class uses surrogate model : distinct, generally intrinsic interpretable, model created to explain the black-box model.
One example of such technique is Locally Interpretable Model-agnostic Explainer, LIME, which acts as a local surrogate method. The intuition behind the model is as follows : it uses an interpretable model (sparse linear model in the original paper) to locally explain the decision boundary of a black-box model around a chosen point x by sampling perturbed points in its vicinity. (In our multi-class problems, LIME typically explains class probabilities, not boundaries.)

The method introduces an interpretable binary representation x' indicating the presence or absence of components of the original instance. Perturbed samples z' are generated by randomly masking componenets of x'. Then, each z' is mapped back to an input-space instance z, on which the black-box model f is evaluated. The perturbed points z are weighted by their proximity to x via a kernel π_x to ensure locality. In the original paper, the distance metrics used were cosine distance for text and $L2$ distance for image.
An interpretable surrogate model $g \in G$ is then fitted by solving :

$$g_x \triangleq \text{argmin}_{g \in G} \sum_{i=1}^{N} \pi_x(z_i) \ell\big(f(z_i), g(z_i')\big) + \Omega(g) \tag{3.1}$$

Where

- g_x is the surrogate explanation model $\in G$ (e.g. sparse linear model as in the original paper [43]). g_x admits a parameter vector β_x interpreted as the local explanation of f around x.
- f is the complex model
- g is the interpretable model
- π_x is a proximity measure between an instance z to x, to define a locality around x. Defined as an exponential kernel in the original paper.
- $\ell\big(f(z_i), g(z_i')\big)$ is loss function quantifying how faithful g is in approximating f in the locality defined by π_x. Typically a weighted least squares on probabilities.
- $\Omega(g)$ measure the complexity of the interpretable function. E.g. in the case of sparse linear model, the number of non-zero coefficients or the depth of the tree in a Decision Tree.

3.3 SHAP

3.3.1 Introduction

SHapley Additive exPlanations, or SHAP, is an additive feature attribution method. The goal of SHAP is to explain the prediction of an instance $\mathbf{x}$ by computing the contribution of each feature to the prediction. It is a local explainer.
This method is based on the Shapley value from the game theory (Lloyd Shapley, 1953), it calculates each feature's average marginal contribution across all possible coalitions. A coalition is an ensemble, a group, of variables.
It is the only solution that satisfies four fundamental properties: efficiency, symmetry, additivity, and the dummy player (or null player) property, which are widely accepted as defining a fair distribution. Fairness here means an allocation that satisfy reasonable ethical and logical conditions. Shapley formalized this intuitive fairness idea as axioms, properties —efficiency, symmetry, additivity and dummy player property.
Lloyd Shapley proved that the shapley value is the only solution that satisfies the 4 properties. The additive feature attribution methods have an explanation model that is a linear function of binary variables. (LIME belongs to the same category.) They are defined as follows :

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z_j' \tag{3.2}$$

- g is the explanation model
- $\phi_j \in \mathbb{R}$ is the feature attribution value
- M is the number of input features, a.k.a. the maximum coalition size.

- $\mathbf{z}' = (z'_1, ..., z'_M)^T \in \{0,1\}^M$

In the coalition vector, $\mathbf{z}'$, an entry of 1 means that the corresponding feature value is "present" and 0 that it is "absent".

3.3.2 The properties

The original paper build a theorethical framework, based on the game theory and original definition of the Shapley value, to showcase a new way of estimating the shapley value, the SHapley Additive exPlanations (SHAP), respecting the four properties —making it a fair method— while being more computationally efficient. SHAP is one of the most theoretically grounded method in the XAI field.

Local accuracy

The local accuracy inforces the fact that "when approximating the original model f for a specific input x, local accuracy requires the explanation model to at least match the output of f for the simplified input x' (which corresponds to the original input x)." [30]

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{3.3}$$

The explanation model $g(x')$ matches the original model $f(x)$ when $x = h_x(x')$, where $\phi_0 = f(h_x(0))$ represents the model output with all simplified inputs toggled off (i.e. missing).

Missingness

The missingness property enforces that "features that are missing in the original input should have no impact on the explanation model." In other words, if a feature is missing in the original input, then its attribution value should be zero. LIME obeys this property.

$$x'_i = 0 \implies \phi_i = 0 \tag{3.4}$$

This property is simply there to act as "minor book-keeping property". The Missingness property enforces that missing features get a Shapley value of 0. In practice, this is only relevant for features that are constant.

Consistency

The consistency property enforces that "if a model changes so that the contribution of a feature increases or stays the same regardless of what other features are present, then the attribution for that feature should not decrease." In other words, if a feature has a higher impact on the model output, then its attribution value should be higher. Let $f_x(z') = f(h_x(z'))$ and z'_{-i} indicate that $z'_i = 0$. For any two models f and f', if

$$f'_x(z') - f'_x(z'_{-i}) \geq f_x(z') - f_x(z'_{-i}) \tag{3.5}$$

for all inputs $z' \in \{0,1\}^M$, then $\phi_i(f', x) \geq \phi_i(f, x)$. With a the consistency axiom, some of the standard Shapley axioms (linearity/additivity, dummy/null, symmetry) become redundant in the characterization. For the interested reader, see the original paper's "Supplementary Material" section. [30]

Theorem 1

"Only one possible explanation model f follows Definition 1 and satisfies Properties 1, 2 and 3 : "

$$\phi_i(f,x) = \sum_{z' \subseteq x'} \frac{|z'|!(M-|z'|-1)!}{M!}[f_x(z') - f_x(z'_{-i})] \tag{3.6}$$

Where

- $|z'|$ is the number of non-zero entries, or features, present in z'
- $z' \subseteq x'$ represents all vectors where the non-zero entries are a subset of the non-zero entries in x'.
- $M!$ is the number of ways to form a coalition of features
- M is the total number of input features or the maximum coalition size.
- ϕ_i is the shapley value.
- $f_x(z') = f(h_x(z')) = E[f(z)|z_S]$ where S is the set of non-zeros entries in binary vector z', i.e. $\bar{S}$ represent the missing features. $S \subseteq F$, where F is the set of all features.

The theorem 1 comes from cooperative game theory, see Lloyd S. Shapley original paper.
"Young (1985) demonstrated that Shapley values are the only set of values that satisfy three axioms similar to Property 1, Property 3 and a final property that we show to be redundant in this setting (see Supplementary Material)"
Following this theorem, one introduce the SHAP values as the solution of equation 3.3.2.
The novelty of the SHAP methods compared to the classical Shapley values is the representation of missing feature, in the binary vector z', using an conditional expectation. The reason behind this choice is that most models do not accept missing values and it represents the interdependence between features in real-world data.
In $h_x(.)$, the mapping function to go from the simplified binary space to the original input space lies one of the big difference between SHAP and LIME.
Missingness is implemented via perturbation (replacement/sampling) to create explicit samples in the original space, whereas SHAP treats missing features as variables to be integrated out under a background distribution. It is a change of paradigm : SHAP considers missing feature as random variables. In short, we approximate $f_x(z')$ in the following way.

$$\begin{aligned} f_x(z') = f(h_x(z')) &= \mathbb{E}[f(Z) \mid Z_S = z_S] && \text{(3.7a)} \\ &= \mathbb{E}_{z_{\bar{S}}|z_S}[f(z)] && \text{(3.7b)} \\ &\approx \mathbb{E}_{z_{\bar{S}}}[f(z)] \qquad \text{(independence)} && \text{(3.7c)} \\ &\approx f\big([z_S, \mathbb{E}(z_{\bar{S}})]\big) \qquad \text{(linearity)} && \text{(3.7d)} \end{aligned}$$

Two optional assumptions SHAP can use two optional assumption as computational shortcuts : features independence 3.7c and model linearity 3.7d.
The features independence 3.7c allows an equality between the conditional and marginal distribution : if the features are independent, conditionning on z_S gives no extra information about $z_{\bar{S}}$. It enables stastically and computationally simpler computations.

$$E_{z_{\bar{S}}|z_S}[f(z)] \rightarrow E_{z_{\bar{S}}}[f(z)] \tag{3.8}$$

The model linearity approximation 3.7d allows us to turn an intractable integral into a single algebraic evaluation.

$$E_{z_{\bar{S}}}[f(z)] \rightarrow f([z_S, E[z_{\bar{S}}]]) \tag{3.9}$$

In other words : independence simplifies the distribution, linearity simplifies the model.

Consequences In practice, both assumptions introduce a bias in our computations. In real-world data, the features are rarely independent. Therefore $p(Z_{\bar{S}}|Z_S) \neq p(Z_{\bar{S}})$. This bias can be important when there is a strong dependence between features.
And most model, to model complex interactions, are rarely linear, meaning by Jensen's inequality that $E_{z_{\bar{S}}}[f(z)] \neq f([z_S, E[z_{\bar{S}}]])$.
The assumptions trade computational efficiency for accuracy (through the introduction of bias).

Optimization

Starting with the LIME equation, and knowing theorem 1 and that LIME belongs to the family of Additive Attribution methods, with an adequate choice of the loss function L, weighting kernel $\pi_{x'}$, and regularization term Ω, the solution of this equation satisfy the three aforementioned properties and is, therefore, the Shapley values.
Now we train the linear model $g(.)$ by optimizing the following loss function L :

$$L(f, g, \pi_{\mathbf{x}}) = \sum_{\mathbf{z} \in \mathbf{Z}} \left(f(h_{\mathbf{x}}(\mathbf{z}')) - g(\mathbf{z}')\right)^2 \pi_{\mathbf{x}}(\mathbf{z}') \tag{3.10}$$

Where

$$\pi_{x'}(z') = \frac{(M-1)}{\binom{M}{|\mathbf{z}'|}\ |\mathbf{z}'|(M - |\mathbf{z}'|)} \tag{3.11}$$

- M is the maximum coalition size, i.e. the number of input features.
- $|\mathbf{z}'|$ is the number of non-zero entries, or features, present in $\mathbf{z}'$.
- $\binom{M}{|\mathbf{z}'|}$ is the number of possible coalitions of size $|\mathbf{z}'|$.

The proof of the above result, called "Theorem 2", is in the original paper's "Supplementary Material" section.

3.4 Diverse Counterfactual Explanations (DiCE)

Counterfactual (CF) explanation is a sub-family of the broader "example-based" explanation method. The latter, as their name hints, selects a particular instance of the datasets (e.g. one of the most representative ones) to explain the behavior of ML models or to explain the underlying data distribution. Many Example-based explanations are model-agnostic because they only require intput-output access.
CF explanations are a constrastive form of example-based explanations : they explain the prediction of an instance by constructing a instance, hence using synthetic data generation, such that the model's prediction changes to a predefined different output. In this thesis we use Counterfactual Explanations (DiCE) [34] framework, which extends the single counterfactual explanation to multiple counterfactual explanations, following property of proximity, sparsity and diversity enabling users to visualize different recourse (CF explanation) options. DiCE is local, post-hoc and model-agnostic explanation technique.
The criteria making a good counterfactual explanation under DiCE are :

- **Proximity** : CF explanations must be as close as possible to the original input.
- **Sparsity** : It should change as few features as possible.
- **Diversity** : It is desirable to generate multiple diverse counterfactual explanations to enable one to pick among multiple viable options of generating a different outcome.
- **Real-world constraints** : A counterfactual instance should have feature values that are likely, i.e. follows the human laws/logic. E.g. a person cannot have negative age.

3.4.1 Optimization

The optimization expression is represented as :

$$C(\mathbf{x}) = \underset{c_1,\ldots,c_k}{\arg\min} \frac{1}{k}\sum_{i=1}^{k} \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k}\sum_{i=1}^{k} dist(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{ dpp\_diversity}(c_1, ...c_k) \tag{3.12}$$

Where $\mathbf{c}_i$ is the counterfactual example, k is the number of generated CF, $f(.)$ is our black box predictive model, yloss is a metric that computes the distance between $f(.)$'s prediction for $\mathbf{c}_i$ and the desired outcome y, d is the total number of input features (i.e. the dimension of x), $\mathbf{x}$ is the original input and dpp_diversity$(.)$ is the diversity metric. λ_1 and λ_2 are hyperparameters that balance the three parts of the loss. And k is the number of counterfactual examples generated.

3.4.2 Choice of the loss

The loss function is intuitive : a valid counterfactual only requires that $f(\mathbf{c})$ be greater or lesser than f's threshold (typically 0.5 in a binary case). We do not need it to be close to extremum 0 or 1. In fact, it would encourage large changes to $\mathbf{x}$ towards the counterfactuals class and generate less feasible counterfactuals.

Therefore, the loss chosen is the hinge loss. It ensures a zero penalty as long as $f(\mathbf{c})$ is above a fixed threshold, i.e. above 0.5 when the desired class is 1. Further, it imposes a penalty proportional to the difference between $f(c)$ and 0.5 when the classifier is correct, and a heavy penalty when $f(c)$ does not indicate the desired counterfactual class.

$$\text{hinge\_loss} = \max(0, 1 - z * \text{logit}(f(\mathbf{c}))) \tag{3.13}$$

Where z is -1 when $y = 0$ and 1 when $y = 1$, and logit$(f(c))$ is the unscaled output of the ML model.

3.4.3 Proximity and diversity term

The *proximity* constraint is represented as follows :

$$\text{proximity} = -\frac{1}{k}\sum_{i=1}^{k} dist(c_i, \mathbf{x}) \tag{3.14}$$

Where $dist(c_i, \mathbf{x})$ represent a distance metric between a counterfactual c_i and the original input $\mathbf{x}$, and k is the number of generated counterfactual explanations.

The *diversity* constraint is represented as follows :

$$\text{dpp\_diversity} = det(\mathbf{K}) \tag{3.15}$$

Where $\mathbf{K}_{i,j} = \frac{1}{1+dist(c_i,c_j)}$ and $dist(c_i, c_j)$ denotes a distance metric between two counterfactual examples.

3.4.4 Choice of the distance function

In our case of only categorical values, one will not use a metric based on the relative frequency of different categorical levels because it does not represent the difficulty of changing a particular feature, which is a central aspect in CF explanations. In the DiCE paper, they use a simple distance "binary" metric. The distance is 1 if the

CF example's value for any categorical feature differs from the original input, otherwise it assigns zero.

$$\text{dist\_cat}(c, x) = \frac{1}{d_{cat}} \sum_{p=1}^{d_{cat}} I(\mathbf{c}^p \neq \mathbf{x}^p) \tag{3.16}$$

Where d_{cat} is the number of categorical feature.

Relative scale of features

Continuous features are just scaled between 0 and 1.
For categorical features, they convert each feature to one-hot encoding and consider it as continuous variable between 0 and 1 to simplify the problem. To encourage the solution to still look like a valid one-hot vector, they add a high penalty for each categorical feature to force its values for different levels to sum to 1. Therefore, the optimizer prefers solutions close to "valid" encodings.
At the end of the optimization, they pick the level with maximum value for each categorical feature to convert back to the original categorical representation.

Chapter 4

Intrinsically Interpretable Model

4.1 Optimal Sparse Decision Tree (OSDT)

Introduction

This section presents an intrinsically Interpretable Model, i.e. a machine learning model regarded as interpretable due to its structure and design choices. It is defined, as the original paper [26] in a binary classification setting.
This Decision Tree (DT) architecture proposes itself as an alternative to the current greedy optimization algorithm CART and C4.5. They grow decision trees from the top-down without backtracking, which means that the algorithms are unable to correct a past mistake about a split.
Optimal Sparse Decicion Tree aims at building, as its name implies, sparse decision tree with a certifiable optimality, set up by the theoretical framework that one will lay in the next sections.
"They find an optimal tree according to a regularized loss function that balances accuracy and the number of leaves. The algorithm is computationnaly efficient due to a collection of analytical bounds to perform massive reduction of the search space." [26]
"The algorithm is able to locate optimal trees and prove optimality (or closeness of optimality), in a reasonable amount of time for datasets of the sizes of tens of thousands or millions of observations and tens of features." [26]

Optimization framework

The bounds are the key of this new architecture and have been heavily inspired by CORELS (Certifiable Optimum RulE ListS). This architecture also has been considered for this work but the lack of any package for my framework (multi-class classification and class imbalanced) dissuades one.
To first define the framework, one will define the different variables used :

- leaf set $d = (p_1, p_2, ..., p_H)$ of length $H \geq 0$ is an H-tuple where p_k is the classification rule of the path from the root to leaf k.
- p_k is a boolean assertion, which evaluates to either true or false for each data point x_n indicating whether it is classified by leaf k, i.e. whether x_n falls into the leaf k. If p_k evaluates to true, we say that leaf k of leaf set d captures x_n.
- d_{un} represent the K leaves that will not be split.
- d_{split} represent the $H - K$ leaves that will be split.
- $\hat{y}_k^{(leaf)}$ label is the majority label of data captured by the leaf k.
- $\delta_{un} = (\hat{y}_1^{(leaf)}; ..., \hat{y}_K^{(leaf)}) \in \{0, 1\}^K$ are the predicted labels of leaves d_{un}

- $\delta_{split} = (\ \hat{y}_{K+1}^{(leaf)}, ..., \hat{y}_{H}^{(leaf)}\ ) \in \{0,1\}^{H-K}$ are the predicted labels of leaves d_{split}
- We define $cap(x_n, p_k)$ as a binary indication telling us if p_k captures the data point x_n.

$$cap(x_n, p_k) = \begin{cases} 1 \text{ if } p_k \text{ captures } x_n \\ 0 \text{ else} \end{cases} \tag{4.1}$$

- The normalized support of the set of leaves β, denoted supp(β, x), is the fraction of data captured by β :

$$\text{supp}(\beta, x) = \frac{1}{N} \sum_{n=1}^{N} cap(x_n, \beta) \tag{4.2}$$

- $\sigma(d)$ represents all descendents of d, i.e. is the set of child trees obtained by slitting d :

$$\sigma(d) = \left\{ (d'_{un}, \delta'_{un}, d'_{split}, \delta'_{split}, K', H_{d'}) : d'_{un} \supseteq d_{un},\ d' \supset d \right\} \tag{4.3}$$

Objective function For a tree d, we define its objective function as a combination of the misclassification error and a sparsity penalty on the number of leaves :

$$R(d, \mathbf{X}, \mathbf{y}) = l(d, \mathbf{X}, \mathbf{y}) + \lambda H_d \tag{4.4}$$

where :

- $l(d, \mathbf{X}, \mathbf{y})$ is the misclassification error of d, i.e. the fraction of training data with incorrectly predicted labels
- H_d is the number of leaves in the tree d.
- λ is a regularization term that penalizes biggers trees.

Statistical learning theory provides a guarantees for this problem ; minimizing the loss subject to a (soft or hard) constraint on model size leads to a low upper bound on test error from Occham's Razor Bound.

Hierarchical Objective Lower Bound An important feature is the decomposition of the loss to create a lower bound. The two parts correspond to the unchanged leaves and the leaves to be split :

$$l(d, \mathbf{x}, \mathbf{y}) = l_p(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y}) + l_q(d_{split}, \delta_{split}, \mathbf{X}, \mathbf{y}) \tag{4.5}$$

Where :

- $l(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y})$ represent the proportion of misclassified data in the unchanged leafs

$$l_p(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} cap(x_n, p_k) \wedge 1[\hat{y}_k^{(leaf)} \neq y_n] \tag{4.6}$$

- $l(d_{split}, \delta_{split}, \mathbf{X}, \mathbf{y})$ is the proportion of data in the leaves we are going to split that are misclassified.

$$l_q(d_{split}, \delta_{split}, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=K+1}^{H} cap(x_n, p_k) \wedge 1\left[\hat{y}_k^{(leaf)} \neq y_n\right] \tag{4.7}$$

- $cap(x_n, p_k)$ gives 1 if the data point x_n is captured in the path p_k leading to leaf k.

- $1\big[\hat{y}_k^{(leaf)} \neq y_n\big]$ is a mathematical condition :

$$1\big[\hat{y}_k^{(leaf)} \neq y_n\big] = \begin{cases} 1 & p_k \text{misclassified} \\ 0 & \text{else} \end{cases} \tag{4.8}$$

- $\wedge$ is define as the minimum operator. Here we take the minimum between two binary terms.

This leads to the definition of a lower bound $b(d_{un}, \mathbf{X}, \mathbf{y})$ on the objective by leaving out the latter loss :

$$b(d_{un}, \mathbf{X}, \mathbf{y}) = l_p(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y}) + \lambda H_d \leq R(d, \mathbf{X}, \mathbf{y}) \tag{4.9}$$

$b(d_{un}, \mathbf{X}, \mathbf{y})$ gives a lower bound on the objective of any child tree of d.

Theorem 1 (Hierarchical objective lower bound) *Define $b(d_{un}, \mathbf{X}, \mathbf{y}) = l_p(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y}) + \lambda H_d$, as in the previous equation. Define $\sigma(d)$ to be the set of all d's child trees whose unchanged leaves contain d_{un}.*
For tree $d = (d_{un}, \delta_{un}, d_{split}, \delta_{split}, K, H_d)$ with unchanged leaves d_{un}, let $d' = (d'_{un}, \delta'_{un}, d'_{split}, \delta'_{split}, K', H_{d'}) \in \sigma(d)$ be any child tree of d such that its unchanged leaves d'_{un} contain d_{un} and $K' \geq K$ and $H_{d'} \geq H_d$, then $b(d_{un}, \mathbf{X}, \mathbf{y}) \geq R(d', \mathbf{X}, \mathbf{y})$

In a sequence of trees where each tree is the parent of the following tree. The lower bounds of these trees increase monotonically because there is no scenario where extending the tree makes (I) error on unchanged leaves decrease, i.e. $l_p(d_{un}, \delta_{un}, \mathbf{X}, \mathbf{y}) \leq l(d')$ and (II) number of leaves decreases, i.e. $\lambda H_d \leq \lambda H_{d'}$.
A powerful consequence of theorem 3.1 is describe in Lemma 3.2.

Lemma 1 (Objective lower bound with one-step lookahead) *Let d be a H_d– leaf tree with a K-leaf prefix and let R^c be the current best objective. If $b(d_{un}, \mathbf{X}, \mathbf{y}) + \lambda \geq R^c$, then for any child tree $d' \in \sigma(d)$, its prefix d'_{un} starts with d_{un} and $K' > K$, $H_{d'} > H_d$ (because we add one leaf and it is a new subtree.), and it follows that $R(d', \mathbf{X}, \mathbf{y}) \geq R^c$. Hence the entire subtree can be pruned.*

This bound tends to be very powerful in practice in pruning the search space, because eventough we might have a candidate $b(d_{un}, x, y) \leq R^c$, if $b(d_{un}, x, y) + \lambda \geq R^c$, we can prune all its child trees.

Lower bounds on Node Support and Classification Accuracy In this subsection, three lower bounds on the fraction of correctly classified data and the normalized support of leaves in any optimal tree is presented. All of them depend on λ.

Theorem 2 (Lower bound on node support) *Let $d^* = (d_{un}, \delta_{un}, d_{split}, \delta_{split}, K, H_{d^*})$ be any optimal tree with objective R^*, i.e., $d^* \in \arg\min_d R(d, x, y)$. For an optimal tree, the support traversing through each internal node must be at least 2λ. That is for each child leaf pair p_k, p_{k+1} of a split, the sum of normalized supports of p_k, p_{k+1} should be no less than twice the regularization parameter, i.e. 2λ*

$$2\lambda \leq supp(p_k, x) + supp(p_{k+1}, x) \tag{4.10}$$

Following this theorem, for a tree d, if any of its internal nodes capture less than a fraction 2λ of the samples, it cannot be an optimal tree, even if $b(d_{un}, x, y) < R^*$ (the lower bound is lower than the lower bound of the optimal tree $R*$)
None of its child tree would be an optimal tree either. Thus, after evaluated d, we can prune the tree d and all of its children, in that case.

Theorem 3 (Lower bound on incremental classification accuracy) *Let* $d^* = (d_{un}, \delta_{un}, d_{split}, \delta_{split}, K, H_{d^*})$ *be any optimal tree with objective* R^**, i.e.,* $d^* \in \arg\min_d R(d, \mathbf{X}, \mathbf{y})$*. Let* d^* *have leaves* $d_{un} = (p_1, ..., p_K)$ *and labels* $\delta_{un} = (\hat{y}_1^{(leaf)}, ..., \hat{y}_{H_d^*}^{(leaf)})$ *For each leaf pair* p_k, p_{k+1} *with corresponding labels* $\hat{y}_k^{(leaf)}, \hat{y}_{k+1}^{(leaf)}$ *in* d^* *and their parent node (the leaf in the parent tree)* p_j *and its label* $\hat{y}_j^{(leaf)}$*, define* a_k *to be the incremental classification accuracy of splitting* p_j *to get* p_k *and* p_{k+1}*:*

$$\begin{aligned} a_k \equiv \frac{1}{N} \sum_{n=1}^{N} \Big\{ & cap(x_n, p_k) \wedge 1\big[\hat{y}_k^{(leaf)} = y_n\big] \\ & + cap(x_n, p_{k+1}) \wedge 1\big[\hat{y}_{k+1}^{(leaf)} = y_n\big] \\ & - cap(x_n, p_j) \wedge 1\big[\hat{y}_j^{(leaf)} = y_n\big] \Big\} \end{aligned}$$

In this case, λ provides a lower bound, $\lambda \leq a_k$.

Theorem 4 (Lower bound on classification accuracy) *Let* $d^* = (d_{un}, \delta_{un}, d_{split}, \delta_{split}, K, H_{d^*})$ *be any optimal tree with objective* R^**, i.e.,* $d^* \in \arg\min_d R(d, \mathbf{X}, \mathbf{y})$*. For each leaf* $(p_k, \hat{y}_k^{(leaf)})$ *in* d^**, the fraction of correctly classified data in leaf* k *should be no less than* λ*.*

$$\lambda \leq \frac{1}{N} \sum_{n=1}^{N} cap(x_n, p_k) \wedge 1\big[\hat{y}_k^{(leaf)} = y_n\big] \tag{4.11}$$

Thus, in a leaf we consider extending by splitting on a particular feature, if that proposed split leads to less than λ correctly classified data going to either side of the split, then this split can be excluded, and we can exclude that feature anywhere further down the tree extending that leaf.

To summarize the Theorems

- Theorem 1 and Lemma 1 lay the ground of the lower boundary and the consequence of the monotonically increasing lower bound within any child of parent tree d.
- We apply Theorem 3 when we split the leaves. We need only split leaves whose normalized supports are no less than 2λ. It checks if a split will represent enough data.
- We apply Theorem 4 when constructing the trees. For every new split, we check the incremental accuracy for this split. If it is less than λ, we further split at least one of the two child leaves. The boundary checks if a split will increase the accuracy.
- Both Theorem 3 and 4 are Theorems on pairs of leaves.
- Theorem 4 regulates the accuracy within a single leaf by giving a lower bound for the classification accuracy in case of split.

Implementation specificities

Four core ideas made this implementation save $> 97\%$ of the execution time.

i) Data structures OSDT maintains compact representations of leaf l :

- The number of data points captured by each leaf, and an identifier for each of them through a binary vector of length N.
- The predicted label and loss of the leaf.
- A set of clauses definining the leaf.
- The lower bound on the leaf misclassification error.
- An indicator of the set of dead features. (Following theorem 4, a dead feature means a feature we cannot use to split anymore because it would not lead to a sufficient amount of correctly classified data points.)
- A boolean indicator indicating if the leaf is dead. Following Theorem 2, a leaf is dead if it captures less than 2λ of the data points.

Additional information for the entire trees is also stored :

- A set of leaves in the tree.
- The objective.
- The lower bound of the objective.
- A binary vector to record the split leaves d_{split} and unchanged leaves d_{un}.

ii) Incremental computation OSDT never recomputes misclassification error, leaf covers, predictions, or lower bounds from scratch. Instead, these quantities are updated incrementally each time a leaf is split, because a child tree differs from its parent in exactly one leaf. This follows the same principle used in CORELS ("incremental evaluation of rule-list error").[19] It reduces the per-node cost to amortized constant time. It simply uses the previously computed quantities and stored quantities as noted above.
For instance, the binary vector storing the split and unchanged leaves is useful because a parent and a child tree share the same unchanged leaves. With the knowledge given by the Lemma 1, storing the lower bound of the objective takes its full value.

iii) Ordering of the worklist The queue dictates the order in which trees are evaluated.
The queue establishes the priority of exploration of the search space. Each entry of the queue corresponds to a tree and the queue serves as a worklist.
It is filled, as highlighted in the pseudo-code of the algorithm 1, by adding all the possible child trees of every tree beating the current best lower bound on the objective function.
The ordering of the worklist represent a scheduling policy (i.e. an agenda on to where we are going to allocate resources next.)
The **curiosity metric** produce the scheduling policy. They use this metric-based ordering compared to other structure-based policy, e.g. breadth first search or depth first search, because it gave the best results in term of runtime and memory consumption. For the interested readers, check the article [19].

iv) Symmetry-aware map Symmetry-aware map, helps the algorithm avoid computing several times the same value for a particular leaf and to do not create equivalent trees multiple times.
A leaf corresponds to all data points that satisfy the binary assertions along the path from the root to that leaf. Since every data point follows exactly one path,the leaves of a decision tree are mutually exclusive.
Internally, OSDT represents a partial tree as a **list of its current leaves**, each leaf storing the subset of samples that reach it. The important idea is that the **order** in which these leaves appear in the list has no effect on the semantics of the partial tree: if two partial trees induce the same partition of the dataset at their leaves, they represent the same search state, even if the leaves appear in different positions in the list.
The Symmetry-Aware Map uses this fact to avoid redundant computation.
It converts each partial tree into a **canonical representation** (for example, by sorting the leaf identifiers), so that two trees that induce the same partition end up with the same canonical form. If this canonical form has already been seen, we do not reconstruct it since the algorithm knows we have already explored that area of the search space; if not, we compute the bounds for the tree and insert it into the list.
In essence: if two partial trees have leaves that contain exactly the same data subsets, then the order in which these leaves are stored in memory does not matter—they correspond to the same partition of the dataset, and only one of them needs to be explored.

v) Pseudo-code of the algorithm

Algorithm 1 Branch-and-bound for learning optimal decision trees.

Input: Objective function $R(d, \mathbf{x}, \mathbf{y})$, objective lower bound $b(d_{\text{un}}, \mathbf{x}, \mathbf{y})$, set of features $S = \{s_m\}_{m=1}^{M}$, training data $(\mathbf{x}, \mathbf{y}) = \{(x_n, y_n)\}_{n=1}^{N}$, initial best known tree d^0 with objective $R^0 = R(d^0, \mathbf{x}, \mathbf{y})$; d^0 could be obtained as output from another (approximate) algorithm, otherwise, $(d^0, R^0) = (\text{null}, 1)$ provides reasonable default values. The initial value of δ_{split} is the majority label of the whole dataset.
Output: Provably optimal decision tree d^* with minimum objective R^*

0.4em

$(d^c, R^c) \leftarrow (d^0, R^0)$ ▷ Initialize best tree and objective
$Q \leftarrow \text{queue}\big([(\emptyset, (), (), \delta_{\text{split}}, 0, 0)]\big)$ ▷ Initialize queue with empty tree
while Q not empty **do** ▷ Stop when queue is empty
  $d = (d_{\text{un}}, \delta_{\text{un}}, d_{\text{split}}, \delta_{\text{split}}, K, H) \leftarrow Q.\text{pop}()$ ▷ Remove tree d from the queue
  if $b(d_{\text{un}}, \mathbf{x}, \mathbf{y}) < R^c$ **then** ▷ **Bound:** Apply Theorem 1
    $R \leftarrow R(d, \mathbf{x}, \mathbf{y})$ ▷ Compute objective of tree d
    if $R < R^c$ **then** ▷ Update best tree and objective
      $(d^c, R^c) \leftarrow (d, R)$
    end if
    for all every possible combination of features to split d_{split} **do** ▷ **Branch:** Enqueue d_{un}'s children
      split d_{split} and get new leaves d_{new}
      for all each possible subset d'_{split} of d_{new} **do**
        $d'_{\text{un}} = d_{\text{un}} \cup (d_{\text{new}} \setminus d'_{\text{split}})$
        $Q.\text{push}\big((d'_{\text{un}}, \delta_{\text{un}}, d'_{\text{split}}, \delta_{\text{split}}, K', H')\big)$
      end for
    end for
  end if
end while
$(d^*, R^*) \leftarrow (d^c, R^c)$ ▷ Identify provably optimal solution

Summary

To summarize, the Optimal Sparse Decision Tree algorithm builds decision trees by optimizing a regularized objective that balances accuracy and model complexity. Unlike greedy top-down methods such as CART or C4.5, OSDT does not commit to early splits: instead, it maintains a global view of the search space and relies on analytical lower bounds to discard all tree structures that cannot outperform the current best one.

By the construction of its theoretical framework, the method is certifiably optimal: once the search terminates, the algorithm proves that no feasible tree—within the regularized objective—can achieve a better score. When the full optimal tree cannot be reached within the computational budget, the same mechanism provides a certificate of near-optimality by reporting how far the incumbent is from the best possible bound.
This efficiency is obtained through a combination of implementation ideas: compact data structures storing leaf statistics, incremental computation of errors and bounds, aggressive pruning via theoretical guarantees on node support and accuracy, and a carefully designed exploration policy. Together, these design choices allow OSDT to explore the relevant part of the search space while avoiding redundant or provably sub-optimal branches.

Model Implementation

The OSDT is optimized using GOSDT solver. The training and test partitions were aligned with the neural-network baseline by reusing the same pre-defined indices (`absolute_idx`).

Feature encoding and predicate generation. Because OSDT operates on binary predicates, the original mixed-type tabular data were transformed into a binary feature space in two steps. First, starting from the raw dataset, categorical variables were one-hot encoded using a full *k*-category encoding.
Second, a threshold-guessing binarizer was applied to generate candidate binary predicates of the form $x_j \leq t$ using a gradient-boosting based heuristic. This produced a binary predicate matrix of dimension $(35540, 11)$ (train) and $(8886, 11)$ (test), where the initial number of feature after encoding was 191.

Model extraction and explanation format. After training, global explanations correspond to the full set of root-to-leaf decision rules, while local explanations are instance-specific decision paths obtained by traversing the tree from the root to the predicted leaf. To reinforce interpretability, predicate indices were mapped to human-readable conditions via a manually defined semantic dictionary.

Reported model summary The final OSDT model achieved a training accuracy of 57.19% and a test accuracy of 57.25%, for a training time of 26.988 seconds. The model has slightly lower performance than the neural network. Its complexity was summarized by 35 internals nodes, 36 leafs nodes, maximum depth observed of 6, and mean decision-path length on the test set 5. The binary predicate space contained 11 candidate predicates, of which 90% were used by the final tree.

Chapter 5

Comparisons

5.1 Introduction

One will follow and define the theoretical framework laid by D. Canha et al (2025) [14]. It is one of the most recent and well constructed framework up to date in the field. They use 11 mathematically defined properties, and their sub-properties, to have a general picture of the strength and weaknesses of the XAI method. The strength of a functionally-grounded evaluation is that it does not require human evaluation and can be automated. It offers a common ground onto which to compare different XAI methods. It is inspired by the Ethics Guidelines for Trustworthy AI published by the European Union's High Level Expert Group (AI HLEG) [20]. The guidelines identify 3 axes participating to the transparency of an AI system :

- Explainability : how the model works, how it makes decisions.
- Traceability : how data and labels have been gathered, what classifier has been used, etc...
- Communication : how explanations can be adapted to the explainee depending on his/her social situation.

In the following analysis, the results are to be understand as what they are : a comparison between XAI methods belonging to different model's families, under a benchmark framework primarily thought for post-hoc attribution based methods, in the context of this classification task on (mostly) categorical data and not-so-accurate predictions models. Some methods will be advantaged or disadvantaged depending on their nature and the context of the evaluation.
Therefore, all results should be interpreted with caution, not as ground truth, and taking into the aforementioned context.
The results in sections "Implementation and Results" are presented in the following way "**method name** (*number*)". The *number* represents the score achieved for the sub-metric. For instance, "**DiCE** (3)" corresponds to a score of 3 for a given metric.

5.2 F1 - Representativeness

5.2.1 F1: Normative definition

Representativeness is about how deeply the XAI method describes the internal workings of the black-box and what type of model and input data it supports. It is broken down into 4 sub-properties denoted by F1.1 to F1.4.

Scope (F1.1)

This property evaluates whether the method give local or global explanations (the entire model behavior vs a single prediction). D. Canha et al (2025) claims that Local Explanations are preferred over Global ones, which explains the following scoring.

$$m_{f1.1} = \begin{cases} 3 & \text{the XAI method supports both Global and Local expl.,} \\ 2 & \text{the XAI method is local} \\ 1 & \text{the XAI method is Global} \end{cases} \tag{5.1}$$

Portability (F1.2)

This property addresses the range of the ML models to which the XAI method can be applied : either *model-specific* methods or *model-agnostic* methods. The former can only be used on certain kind of model, while the latter can be applied to any kind of models. The higher scores are assigned to model-agnostic methods because of their generability.

$$m_{f1.2} = \begin{cases} 2 & \text{the XAI method is model-agnostic} \\ 1 & \text{the XAI method is model-specific} \end{cases} \tag{5.2}$$

Access (F1.3)

This property's role is to evaluate the degree to which the XAI method requires access to the training data or the model itself to generate an explanation.
The metric is divided into two sub-properties : Data Access and Model Access. Together, they quantify the level of privacy-preserving capability offered by the XAI method.

$$m_{f1.3A} = \begin{cases} 3 & \text{no (training) data access required} \\ 2 & \text{data required only for initialization} \\ & \text{(e.g. creating an explainer object)} \\ 1 & \text{data required for each explanation} \\ 0 & \text{(full dataset required for (re)training)} \end{cases} \quad m_{f1.3B} = \begin{cases} 3 & \text{no model required (any function } f(X)) \\ 2 & \text{prediction function access only} \\ & \text{queries to a trained model} \\ 1 & \text{partial access (e.g. gradients)} \end{cases} \tag{5.3}$$

Practicality (F1.4)

This property measures the applicability across diverse data types and the efficiency of handling large datasets by the XAI method. It is divided into two sub-properties : $m_{f1.4A}$ Applicability and $m_{f1.4B}$ Scalability.

$$m_{f1.4A} = \begin{cases} 2 & \text{data-agnostic} \\ 1 & \text{partially data-specific} \\ & \text{(requires preprocessing)} \\ 0 & \text{fully data-specific} \end{cases} \qquad m_{f1.4B} = \begin{cases} 2 & \text{highly scalable} \\ 1 & \text{moderately scalable} \\ 0 & \text{not scalable} \end{cases} \tag{5.4}$$

5.2.2 F1 : Implementation and results

F1.1 Scope

LIME (2) Lime is a local explanation technique by definition and does not provide global explanations.

KernelSHAP (2) KernelSHAP produces local explanation ; KernelSHAP cannot give global explanations : approximated Shapley values, highly volatile and coefficients from different regressions cannot be meaningfully aggregated. Final score : 2

DiCE (2) DiCE generates instance-level counterfactual explanations and is inherently local.

OSDT (3) OSDT's learned decision tree constitutes a complete global explanation, and each prediction is explained locally by a decision path from the root to a leaf.

F1.2 Portability

LIME (2) : Model-agnostic; relies only on access to a prediction function.

KernelSHAP (2) : Model-agnostic.

DiCE (2) : Formulation is model-agnostic and data-agnostic, relying only on access to a prediction function and a defined distance metric. Final score : 2

OSDT (1) : Model-specific; OSDT's explanations are inseparable from the predictive model itself.

F1.3 Access

LIME (2 + 2) : Data only required for the initialization and it requires only an access to the predictive function, through querying a trained model.

KernelSHAP (2 + 2) : Same justification as LIME.

DiCE (2 + 2) : Same justification as LIME.

OSDT (3 + 0) : Its explanations do not require access to any external predictive model, as the explanatory structure is identical to the classifier itself. However, OSDT requires the full access to the training data to construct the explanatory model.

F1.4 Practicality

LIME (2 + 1) : Data agnostic, and moderately scalable. It can be computationnaly expensive for large datasets.

KernelSHAP (2 + 0) : Data agnostic, but impractical for large datasets.

DiCE (2 + 1) : It is most commonly applied to tabular data in practice, but the formulation is data-agnostic. DiCE can be applied to moderately large datasets.

OSDT (1 + 1) : It is primarily applicable to tabular data with finite, discretizable features. OSDT can be applied to moderately large datasets, it exhibits significant computational overhead as model complexity increases.

Summary Table below summarizes the F1 scores across all methods. We can observe that **LIME** and **DiCE** have the highest scores, indicating higher representativeness under this metrics implementation choice.

Table 5.1: F1 Practicality scores

	Scope (F1.1)	Portability (F1.2)	Access (F1.3)	Practicality (F1.4)	Total
LIME	2	2	4	3	**11**
KernelSHAP	2	2	4	2	10
DiCE	2	2	4	3	**11**
OSDT	3	1	3	2	9

5.3 F2 - Structure

This new property focus on the *how* the explanations are presented and their visual aspects.

5.3.1 F2 : Normative definition

Expressive power F2.1

This property assesses the extent to which the XAI method supports diverse and comprehensible representation formats or languages. n is the number of distinct explanatory outputs (e.g. importance scores, counterfactuals) $|F|$ is the number of unique representation formats (e.g. textual summary, visual graph of text), C is a predefined set of comprehensible formats including : decision trees, text summary, bar plots, rule sets or paths, $1(f \in C)$ is an indicator function that returns 1 if the format f belongs to the set C, 0 otherwise.

$$m_{f2.1} = n + |F| \frac{\sum_{f \in F} 1(f \in C)}{|F|} \tag{5.5}$$

In this work, Expressive Power is assessed based on the explanatory outputs explicitly produced and presented by each method, and the possibilities allowed by the explanation's format to be visually showcased.

Graphical integrity (F2.2)

This property assesses if the explanation makes a distinction between positive and negative attributions.

$$m_{F2.2} = \begin{cases} 1 & \text{pos. and neg. attributions are visually distinguishable} \\ 0 & \text{they are not visually distinguishable} \end{cases} \tag{5.6}$$

Morphological Clarity (F2.3)

This property assesses whether the explanation visually discrimates more relevant attributions from less relevant ones.

$$m_{F2.3} = \begin{cases} 1 & \text{the most relevant attributions are visually distinguisable} \\ 0 & \text{they are not visually distinguishable from less relevant ones} \end{cases} \tag{5.7}$$

Layer Separartion (F2.4)

This property assesses whether the explanation visualization omits or occludes the original input instance, which should be visible for explainee inspection.
In this work, **Layer Separation** is applied conservatively, focusing stricly on the information contained in the explanation output, not on what might be accessed outside of the explanations.

$$m_{f2.4} = \begin{cases} 1 & \text{The original input instance is visible} \\ 0 & \text{the original input instance is ommited or occluded} \end{cases} \tag{5.8}$$

5.3.2 F2 : Implementation and Results

F2.1 Expressive power

Expressive power quantifies the visual versatility of methods.

LIME (3.5) : Produces a single explanatory output : a local feature coefficients. Two representations formats are considered : table summary and a bar plot.

KernelSHAP (3.5) : Produces a single explanatory output : Shapley values. The same two representation formats as lime are considered.

DiCE (2.0) : Produces counterfactual instances as explanation and supports a single representation format : table-based summary of the counterfactual instances.

OSDT (4.0) : Produces decision paths/rules as explanations and supports two unique representation formats : rule sets and decision trees.

F2.2 Graphical integrity

LIME (1) : Positive and negative attributions are visually distinguishable in the bar plot.

KernelSHAP (1) : Same justification as LIME.

DiCE (0) : Counterfactual instances do not provide positive or negative attributions.

OSDT (0) : Same justification as DiCE.

F2.3 Morphological Clarity

LIME (1) : The bar plot visually distinguishes the most relevant attributions.

KernelSHAP (1) : Same justification as LIME.

DiCE (0) : Counterfactual instances do not visually distinguish relevant features.

OSDT (1) : Although OSDT does not assign graded attributions, its tree structure provides a *visually distinguishable* hierarchy of features.

F2.4 Layer separation

LIME (0) : The original input instance is occluded in the bar plot.

KernelSHAP (0) : Same justification as LIME.

DiCE (0) : The original input instance is not shown in the counterfactual instances.

OSDT (0) : The original input instance is occluded in the decision paths/rules.

Summary The below table summarizes the F2 scores across all sub-metrics. We can observe that **LIME** and **KernelSHAP** have the highest scores, indicating, under this metrics implementation choice, a more interpretable visual structure.

Table 5.2: F2 Structure scores

	Expr. Power (F2.1)	Graph. Int. (F2.2)	Morph. Clarity (F2.3)	Layer Sep. (F2.4)	Total
LIME	3.5	1	1	0	**5.5**
KernelSHAP	3.5	1	1	0	**5.5**
DiCE	2.0	0	0	0	2
OSDT	4.0	0	1	0	5

5.4 F3 - Selectivity

5.4.1 F3 : Normative definition

This property evaluates the length of the explanation by the methods. Based on Miller's Law, 7 ± 2 is the optimal number of information that an Human can process and remember. This property is inspired by it, reaching its maximum value when the number of explanations is 7 or it allows adjustment to $s = 7$, where s is the explanation size.

$$m_{f3} = \begin{cases} 1 & \text{if } s = 7 \text{ or tunable to } s = 7 \\ \exp(-\frac{(s-7)^2}{2\sigma^2}) & \text{if } s \neq 7 \text{and not tunable} \end{cases} \tag{5.9}$$

5.4.2 F3 : Implementation and Results

LIME (1) : The method allows one to tune the number of feature's score seen, therefore yielding a maximum score.

KernelSHAP (1) : The methods allows one to tune the number of Shapley values seen.

DiCE (0.9) : The best approximation of the explanation size s for DiCE is the average number of feature changes across multiple counterfactual, data points and target classes. The approximation is $s_{\text{DiCE}=8}$.This value is used as input to the benchmark's selectivity scoring function. However, the confidence interval of the standard deviation of this metric is $[5.5, 7.6]$. This highlights the high variability across instances and target classes in the number of feature changes required to achieve counterfactuals.

OSDT (0.6) : For OSDT, local explanations correspond to the decision paths from the root to a leaf node, leading to a prediction. Therefore, we computed the **Selectivity** metric as the mean number of conditions (feature tests) in the decision paths across all test instances. This number here is (rounded to) 5.

Summary The below table summarizes the F3 scores across all methods. We can observe that **LIME**, **KernelSHAP** and **DiCE** outperform **OSDT** in terms of quantity of information shown. It is mostly due to the computational choice of the metric that will be criticized in the discussion section. Overall, all methods perform relatively well in this metric.

Table 5.3: F3 Selectivity scores

	Total
LIME	**1.0**
KernelSHAP	**1.0**
DiCE	0.9
OSDT	0.6

5.5 F4 - Contrastivity

5.5.1 F4 : Normative definition

This property indicates whether or not the method discriminates the explanation from an alternative outcome, like counterfactual explanation would do. This idea is divided into two sub-properties : Contrastivity Level F4.1 and Target Sensitivity F4.2

Contrastivity Level (F4.1)

This sub-property assesses how effectively the generated explanation provides contrastivity relative to a standard reference point (e.g. in Shapley's value, $\phi_0 = E[f(X)]$ is used as a baseline and is the expected prediction of the model over the data distribution, which is approximate in practice by the average prediction of the model). The higher scores are given to methods that explain the model output compared to multiple reference points.

$$m_{f4.1} = \begin{cases} 2 & \text{the expl. compares the output with multiple events} \\ 1 & \text{the expl. compares the output with a single reference point} \\ 0 & \text{no Contrastivity provided} \end{cases} \tag{5.10}$$

Target Sensitivity (F4.2)

This property test the robustness of the method against adversarial attacks. A reliable explanation should be sensitive to the alteration of inputs that produce (significant) changes in the model's outputs.
Methods for generating adversarial samples differ based on data type. For our case in tabular data, we will use a nearest counterfactual approach.
E_1 represents the explanation before perturbation, E_2 represents the explanation after perturbation, $d(E_1, E_2)$ represent the distance between the two explanations (e.g. Euclidean distance for feature contributions), and d_{max} the maximum distance for normalization.

$$m_{f4.2} = \frac{d(E_1, E_2)}{d_{max}} \tag{5.11}$$

5.5.2 F4 : Implementation and Results

F4.1 Contrastivity Level : scores

LIME (1) : LIME compares the output with a single reference point, the predefined baseline, which is most commonly the locally weighted, average prediction of the black-box model in the neighborhood of the instance. The term β_0 in g_x from equation 3.1.

KernelSHAP (1) : KernelSHAP explains predictions relative to ϕ_0 in 3.2, which represents the expected model output over the background distribution.

DiCE (2) : DiCE shows by definition several multiple different CF explanations.

OSDT (0) : Although decision trees implicitly encode alternative deicison paths corresponding to different outcomes, OSDT explanations [1], following the exact wording of the sub-metric, do not explicitly articulate contrasts with a reference outcome.

F4.2 Target Sensitivity

LIME (0.3) : The euclidean distance is computed between the explanation of the original input and the explanation of the closest counterfactual to that reference input, targeting each alternative class. The closest valid counterfactual was chosen to mimic a *slight* perturbation. The sensitivity metric is summarized by the median, assuming equal relevance of alternative classes. The interquartile range is 0.1750, implying a substantial variation among classes in the explanation robustness against minimal perturbation.

KernelSHAP (0.4) : The methodology is same as for LIME. The interquartile range is 0.4000 implying a large variation.

DiCE (0.0) : The notion of target sensitivity to small perturbations of the input is unbridgeable with the design objective of the method. Consequently, this metric is not evaluated for DiCE.

OSDT (0.5) : Target-conditioned explanation generation is degenerate, since the explanation is uniquely determined by the model structure and instance. We therefore reinterpret target sensitivity as a measure of structural contrastivity, quantifying whether a predicted class is supported by a single dominant decision rationale or by multiple competing rule paths. We replace instance-level target perturbation with a global structural dispersion measure over decision paths. The sensitivity proxy is defined as $1 - p^{(c)}_{max}$, for each class c.

$$\text{sensitivity} = 1 - p^{(c)}_{max} \quad \text{where} \quad p^{(c)}_{max} = \max_{p \in \mathcal{P}_c} P(\text{path} = p \mid \hat{y} = c) \tag{5.12}$$

is the maximum conditional probability of a single decision path among all paths $\mathcal{P}_c$ predicting class c. The probability is computed as the proportion of point in that class sharing this unique path.
The final result is the median across classes of the sensitivity proxy metric to reduce sensitivity to class imbalance (i.e. outliers). Higher value of this metric, implying small p_{max}, indicates the absence of a single dominant decision path. The interquartile range is 0.4182 implying large variation among the proportion of predicted points by the dominant path by classes. It means that certain classes have one dominant decision path, i.e. mostly the same local explanation for instances, while others do not have a dominant decision path but several different ones.

[1] When we refer to OSDT's explanations, we are talking about a decision path from the root to a leaf leading to a class' prediction

Summary The table 5.4 summarize the results and showcase that, despite an absence of value for the F4.2, DiCE arises as the most contrastive method.

Table 5.4: F4 Contrastivity scores

	Contrastivity Level (F4.1)	Target Sensitivity (F4.2)	Total
LIME	1	0.3	1.3
KernelSHAP	1	0.4	1.4
DiCE	2	0.0	**2.0**
OSDT	0	0.5	0.5

5.6 F5 - Interactivity

5.6.1 F5 : Normative definition

This property quantify the interactivity of the method. Ideally, explanations should allow the explainer and the explainee to interact with each other, e.g. simply to adapt the explanation to the level of expertise or simply explore the explanations.
We can divide it into two sub-properties :

Interaction level (F5.1)

This property evaluates the degree to which the XAI methods allow interactivity with the user.

$$m_{f5.1} = \begin{cases} 2 & \text{interactive control is provided by default} \\ 1 & \text{no default interaction but it can be implemented (e.g. via API)} \\ 0 & \text{no interaction is provided, and implementing it is complex} \end{cases} \tag{5.13}$$

Controllability (F5.2)

The property addresses how the explanation method can be controlled and if it has built-in enhancing mechanisms such as enabling user feedback.

$$m_{f5.2} = \begin{cases} 4 & \text{full user control with dynamic feedback improving explanations} \\ 3 & \text{partial control with significant user influence on explanations} \\ 2 & \text{limited control with pre-defined options for refinement} \\ 1 & \text{minimal control (e.g. visual exploration)} \\ 0 & \text{no control over explanations} \end{cases} \tag{5.14}$$

5.6.2 F5 : Implementation and Results

F5.1 Interaction level : scores

LIME (1) : No default built-in interaction but it can be implemented (e.g. API, Shiny)

KernelSHAP (1) : Same justification as LIME.

DiCE (1) : Same justification as LIME.

OSDT (1) : Same justification as LIME.

F5.2 Controllability : scores

LIME (2) : Besides visual exploration, explanations can be refined by choosing specific features or adjusting the number of features shown, offering limited control.

KernelSHAP (2) : Similar justification as LIME.

DiCE (2) : Provides limited controllability via desired target class, set of features allowed to vary, permitted value ranges, etc.

OSDT (1) : It does not provide any interactive functionality after training besides visual exploration.

Summary The table 5.5 shows similar results across the board due to the absence of interactivity as a central feature in chosen methods.

Table 5.5: F5 Interactivity scores

	Interaction Level (F5.1)	Controllability (F5.2)	Total
LIME	1	2	**3**
KernelSHAP	1	2	**3**
DiCE	1	2	**3**
OSDT	1	1	2

5.7 F6 - Fidelity

5.7.1 F6 : Normative definition

This property measures the extent to which the explanations is close to the true decision-making process of the underlying model.
For instance, it's crucial to know whether the information comes from a surrogate model, makes linearity assumptions about the model or neither of these.

Fidelity check (F6.1)

This property simply reflect the fidelity to the underlying model, i.e. "penalizing" techniques making linear-assumption or using surrogate model.
The goal is to highlight misleading conclusion on complex non-linear data draw by methods making unrealistic assumptions in this context, where explainability is the most important in this framework.
Therefore, we reward techniques without the aforementioned qualities.

$$m_{f6.1} = \begin{cases} 1 & \text{no surrogate model or linearity assumptions are used} \\ 0 & \text{a surrogate model OR a linearity assumptions are used} \end{cases} \tag{5.15}$$

Surrogate Agreement (F6.2)

This assesses, —if a surrogate model is used— the extent to which its predictions align with those of the black-box model.
The evaluation metric normalize the average prediction difference between the black-box and the surrogate model.

The solution is a score between 0 and 1 translating the alignment in prediction between the surrogate and black-box model.
When no surrogate model is used, the score is assign to the maximum value $m_{f6.2} = 1$

$$m_{f6.2} = 1 - \frac{\sum_{i=1}^{N} |b(x_i) - s(x_i)|}{N \max(b(x))} \tag{5.16}$$

where :

- $b(x_i)$ is the prediction of the black-box model for the instance x_i
- $s(x_i)$ is the prediction of the surrogate model for the instance x_i
- N is the number of instances used for the evaluation
- The denominator is simply the number of samples evaluated times the maximum value of the black-box model predictions. In classification cases $\max(b(x)) = 1$.

It should be mentioned that surrogate agreement evaluates faithfulness of the surrogate model to the black box, not the human interpretability or causal validity of the explanation. This dissociates F6 Fidelity metric from the F7 Faithfulness metric

5.7.2 F6 : Implementation and Results

F6.1 Fidelity check : scores

LIME (0) : It uses a surrogate model to locally approximate the behavior of the black-box model.

KernelSHAP (0) : It estimates Shapley values via a weighted linear regression fitted on simplified samples. It constitutes an approximation mechanism used to derive the explanation, is therefore considered surrogate-based.

DiCE (1) : It does neither rely on surrogate model nor on any linearity assumptions.

OSDT (1) : The method does neither make a linearity assumption nor use a surrogate model.

F6.2 Surrogate Agreement : scores

LIME (0.7) : Surrogate agreement is evaluated by comparing the predictions of the local linear surrogate learned by LIME with the predictions of the underlying neural network, following Equation 5.16. Agreement is computed over a set of randomly sampled instances, providing an empirical estimate of how well the surrogate reproduces the black-box model predictions beyond the single explained instance. The reported score corresponds to the mean surrogate agreement, with uncertainty quantified via a 95% confidence interval under an IID sampling assumption $[0.6617, 0.7383]$.

KernelSHAP (0.8) :

The same surrogate agreement procedure is applied to SHAP : predictions are reconstructed using the additive attribution model (base value plus feature attributions). Higher agreement is expected due to SHAP's theoretical guarantees of local accuracy, and the resulting score reflects the extent to which these guarantees hold empirically for the chosen dataset and target class. The 95% confidence interval is $[0.7653, 0.8347]$.

DiCE (1) : Does not use a surrogate model.

OSDT (1) : Does not use a surrogate model.

Summary The table 5.6 showcase the limited scope of the fidelity metric : DiCE and OSDT yield maximum result by not using surrogate or linearity assumption.

Table 5.6: F6 Fidelity scores

	Fidelity Check (F6.1)	Surrogate Agreement (F6.2)	Total
LIME	0	0.7	0.7
KernelSHAP	0	0.8	0.8
DiCE	1	1.0	**2.0**
OSDT	1	1.0	**2.0**

5.8 F7 - Faithfulness

5.8.1 F7 : Normative definition

In this property, we assesse how reliably the XAI method capture the behavior of the black-box model, whatever the scope. It's important to make the distinction between faithfulness and fidelity. Fidelity checks the alignment between the explanation model and the black-box model, in the case of post-hoc methods, and the whether the model makes linearity assumption. While Faithfulness, intents to compare the *explanations* given by the model with the nature of the black-box model.

Incremental Deletion(F7.1)

This property defines how the progressive removal of input features identified as relevant by the XAI methods impacts the predictive model's output f.

$$m_{f7.1} = \frac{1}{N}\sum_{j=1}^{N}\frac{\int_0^n \Big(f_j(i_r) - f_j(i)\Big)di}{\int_0^n f_j(i_r)di} \tag{5.17}$$

where :

- N is the number of instances evaluated
- n denotes the number of incrementally removed features.
- f_j is the black-box model prediction for the j^{th} instance".
- i_r represents the i^{th} feature removed based on a random explainer.
- i corresponds to the i^{th} relevant feature according to the XAI method.
- The integral $\int_0^n$ is the area under the curve (AUC) of the model's predicted probability after removing k features $(0 \leq k \leq n)$. A smaller AUC means the probability collapses quickly, indicating a correct importance ranking of the method, i.e. a faithful explanation.

RemOve And Retrain (ROAR) (F7.2)

This sub-property is tailored for global explanation methods. It compare the accuracy of the model after incrementally removing important features following the XAI methods and the accuracy of the model after removing random features.
To quantify, it estimates the area between the curves. (This metric was not computed for any ofthe studied methods because it requires global feature ranking, which is not avaibble for DiCE and OSDT, and is not meaningful in my

opinion for both LIME and SHAP. The global feature ranking provided by SHAP is more of a feature summary than a true global feature importance ranking.)

$$m_{f7.2} = \begin{cases} \frac{\int_0^n \big(a(i_r) - a(i)\big) di}{\int_0^n a(i_r) di} & \text{for classification task} \end{cases} \tag{5.18}$$

where :

- $a(i_r)$ is the accuracy of the model after retraining without a random feature.
- $a(i)$ is the accuracy of the model after retraining without a important feature following the XAI method.

White-box Check(F7.3)

The White-box Check's goal is again to test if the explanation model truly grasps the underlying reasoning of black-box model.
It uses a white-box surrogate model trained to mimic the black-box one and It will serve as explanation. His metric uses the proportion of sample for which both the surrogate model and the black-box model are aligned.
The surrogate model used depends on the explanation format. For instance, linear regression model serves effectively as white-box for LIME.
In this work, the white-box check will be measured on control synthetic data to test the explanation's model capacity to truly identify the model's reasoning.

$$m_{f7.3} = \begin{cases} 3 & \text{agreement} \geq 95\% \\ 2 & 80\% \leq \text{agreement} < 95\% \\ 1 & 60\% \leq \text{agreement} < 80\% \\ 0 & \text{agreement} < 60\% \end{cases} \tag{5.19}$$

5.8.2 F7 : Implementation and Results

F7.1 Incremental Deletion : scores

This metric was only evaluated for one class, "CAQ", due to lack of of time and of sufficient large amount of data points for minority class. The score for all the methods were computed on the same subset of individuals.
Instead of a Deletion, one decided to perturb the important features defined by the model in order to avoid having to retrain the model, which is sensitive to feature's order.
For DiCE and OSDT a concept as close as possible to feature importance was inferred, due to its absence in their design.

LIME (0.8) : Incremental deletion is computed by perturbing values in a decreasing order of importance defined by LIME instance-level feature's attribution. The probability decay curve is summarized via the area under the curve (AUC). A random-feature ordering is used as a baseline to normalize the score and control for arbitrary perturbations.

KernelSHAP (0.6) : Uses the same methodology as LIME with ordering the perturbation by SHAP attribution.

DiCE (0.5) : For DiCE, Incremental Deletion (F7.1) is adapted to the counterfactual explanation setting by interpreting feature importance through counterfactual feature changes. For each evaluated instance, counterfactuals are generated toward all alternative target classes. Features whose values differ from the original instance are treated as explanatory and prioritized in a feature ranking.
Faithfulness is assessed by incrementally reverting counterfactual features back to their original values and measuring the degradation of the predicted probability of the target class. The resulting probability curves

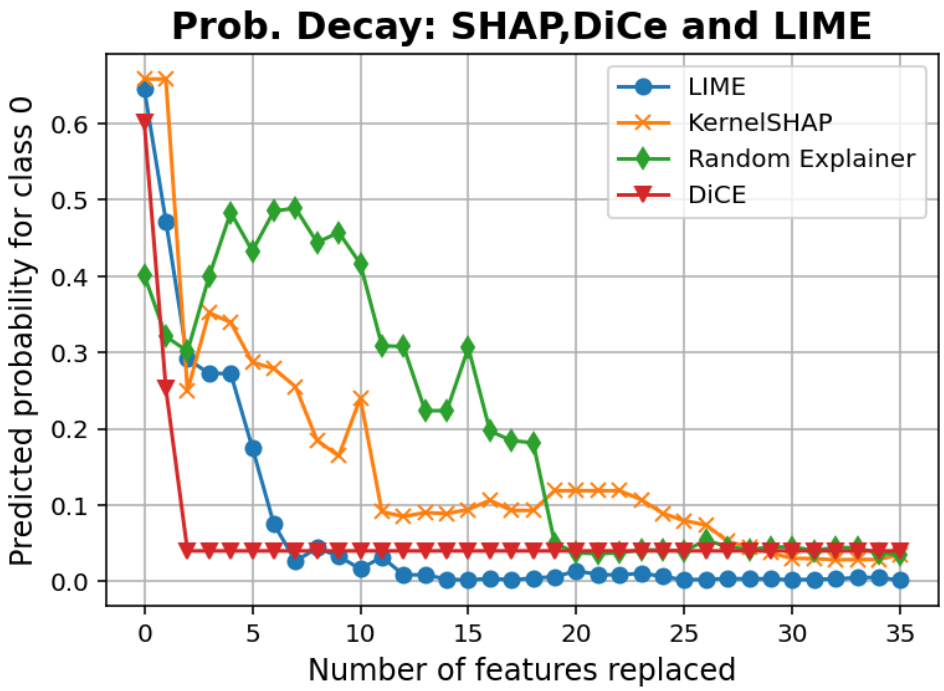
are summarized using the area under the curve (AUC). A random-feature baseline is used to control for arbitrary perturbations. Scores are averaged across counterfactuals, target classes, and instances to obtain the final F7.1 score.
Higher scores indicate that the features modified by DiCE are functionally responsible for the counterfactual prediction, providing evidence that the generated explanations faithfully reflect the model's decision logic.

OSDT (0.6) : Here "important features" for an explanation are the features on its decision path. The importance is structural. We rank them with the idea : the earlier a feature is on the decision, the more important it is, because an alternative decision would lead to drastically different decision path.
OSDT operates on a binary features sub space resulting from thresholding and categorical encoding. The incremental deletion becomes flipping the important binary features. It becomes evaluating label stability rather than probability decay, as the model is deterministic.
The deletion becomes a gradual flipping, starting from the most important to the least, of the binary features along the input's decision path. A binary indicator records whether the prediction changes with the modification of on (decision) path feature k :

$$\mathbb{I}\big[\hat{y}(x^k_{on}) \neq \hat{y}(x)\big] \tag{5.20}$$

For each k feature, we randomly change k off-path features multiple times, to try to reduce the variability of randomly choosing a feature. Then we average the binary indicators vector quantifying whether the prediction changes or not, to a scalar representing the average change in the input prediction due to changing an off-path feature. This quantity, by design, is always zero : flipping the value of feature that's not on the decision path will not impact the prediction.
Each instance yields a deletion curve describing how prediction stability degrades as a growing fraction of explanation-relevant features is removed.
To summarize this behavior independently of explanation length, deletion steps are expressed as a fraction of the decision path, and curves are aggregated across instances. Faithfulness is then quantified by the area under the resulting degradation curve, which captures both the speed and magnitude of prediction collapse. The computation becomes $F_{7.1} = auc_{\text{on path}} - auc_{\text{off path}}$, the latter being 0.

The following plot shows the probability decay induced by Incremental Deletion for KernelSHAP, LIME and DiCE for a single instance. The plot clearly shows the correct identification of important features by the method by emphasizing a difference between method's feature importance and random feature selection.



F7.2 ROAR : scores

The ROAR metric was ommitted from the analysis because it requires a global feature ranking, which LIME and SHAP are unable to meaningfully provide. DiCE is a instance-level explanation and does not fournish any informations about global feature importance. Although global feature information could be derived from OSDT, the time required to implement it and the unavailability of any possible comparison with other metrics lead to an non implementation of it.

F7.3 White box check : scores

LIME (1) : A synthetic classification task with a known linear data-generating process is created. Ground-truth feature influence, i.e. instance level-deviation from a baseline prediction, is computed analytically and serves as a reference explanation. LIME's explanations are directly compared to this influence vector. The evaluation assesses whether the method recovers the true feature influence structure under perfect conditions. LIME reached a agreement of 62%.

KernelSHAP (2) : Same explanation as LIME. KernelSHAP exhibits the highest agreement, 91% among the post-hoc metrics.

DiCE (0) : Same setup as LIME and KernelSHAP, but DiCE, being a example-based method, is evaluated through relevance inferred from counterfactual feature changes. A feature is marked as relevant if it changes in at least half of the generated counterfactuals. Agreement is computed as the proportion of features for which DiCE's relevance decision matches the ground truth relevance decision. The latter being defined as a binary indicator labelling the most relevant features on the instance level. DiCE agreement is 53%.

OSDT (2) : A synthetic classification task is constructed using an simple rule-based decision tree, which serves as the true white-box mechanism and ground truth. OSDT is trained on a binarized predicate representation, as it is required. For each test instances, the model's prediction is compared to the ground-truth rule output. Faithfulness is simply computed as the proportion of instances for which both decisions agree, i.e. the model correctly identify the reasoning of the underlying problem. OSDT yielded an agreement of 92%

One understands that this test is trivial and does not prove much besides the model's capabilities in simple framework.

Summary The table 5.7 highlights the two most faithful methods as defined in this framework : OSDT and KernelSHAP. DiCE's score reflects the difficulty in adapting example-based methods to certain criteria of the benchmark, which were mainly designed for post-hoc attribution methods.

Table 5.7: F7 Faithfulness scores

	Incremental Deletion (F7.1)	ROAR (F7.2)	White-Box Check (F7.3)	Total
LIME	0.8	NaN	1	1.8
KernelSHAP	0.6	NaN	2	**2.6**
DiCE	0.5	NaN	0	0.5
OSDT	0.6	NaN	2	**2.6**

5.9 F8 - Truthfulness

5.9.1 F8 : Normative definition

This property checks if the explanations are aligned with common knowledge of the user's true world. It includes being accordant with prior relevant domain knowledge and beliefs of the "explainee" but also to detect biased models.

Reality Check (F8.1)

This property test if the XAI methods prevents the generation of unrealistec data samples, ensuring alignment with real-world knowledge. For instance, a feature age should not be negative.
It is divised into two sub-properties :
Feature constraints consistency ($m_{f8.1A}$) :It ensures that the generated explanations do not violate feature bounds.
Feature correlation consistency ($m_{f8.1B}$) : It ensures that the generated explanations follows the same observed correlation as in the training data.
The final score range from 0 (least realistic) to 2 (most realistic)

$$m_{f8.1} = m_{f8.1A} + m_{f8.1B} \tag{5.21}$$

$$m_{f1.3A} = m_{f8.1A} = \begin{cases} 1 & \text{expl. respect feature constraints} \\ 0 & \text{they does not} \end{cases} \qquad m_{f1.3B} = m_{f8.1B} = \begin{cases} 1 & \text{expl. represt feature correlations} \\ 0 & \text{they does not} \end{cases} \tag{5.22}$$

Bias Detection (F8.2)

This property evaluates whether the XAI method can reveal biases within the model or dataset. By accomplishing this property, the method goes beyond the scope of simply giving information, overall, it help improving the model reliability.
This method can be implemented using biased models or synthetic data to showcase the XAI method's ability to expose biases. It includes Husky-vs-Wolf Classifier,Gendered Occupation Models, Simulated Bias (through

Synthetic Data), which will neither be used nor study in this work.
In this work, Bias Detection measures potential to reveal bias, not guaranteed bias identification.

$$m_{f8.2} = \begin{cases} 1 & \text{bias is exposed by the XAI method} \\ 0 & \text{bias is not detected} \end{cases} \tag{5.23}$$

5.9.2 F8 : Implementation and Results

F8.1 Reality Check : scores

LIME (0 + 0) : A) LIME does not enforce hard constraints on feature validity when doing local perturbations. Nonetheless, perturbations are made in the vicinity of an existing point. If allowed, the score would be of 0.5. B) It perturbs features independently when generaing local samples, without enforcing any joint feature dependencies.

KernelSHAP (0 + 0) : A) It relies on background data to approximate feature contributions through sampling. This strategy uses empirical data distribution, and de facto following the world-reality of the data. Nonetheless, it does not enforce explicit feature constraints. As of LIME, the score would be of 0.5. B) KernelSHAP assumes feature independence when estimating Shapley values through sampling.

DiCE (1 + 0) : A) DiCE enable user-defined feature constraints such as permitted ranges. The contraints are enforced during CF generation, ensuring all explanations remain consistents. B) DiCE does not explicitly model or enforce feature correlations during counterfactual generation. It does not guarantee that generated CFs preserve obsersed dependencies in the training data.

OSDT (1 + 1) : A) The method does not generate synthetic instances as part of explanation, therefore it cannot violate feature constraints via generation.
B) Correlation consistency is defined over generated explanation instances. Since OSDT produces no synthetic instances, the metric is non-diagnostic for this method and is satisfied by default.

F8.2 Bias Detection : scores

Following the definition, all the studied methods have the tools be able to reveal bias by studying their outputs. For instance, LIME and KernelSHAP can highlights systematic dominance of certain features.

Summary The table 5.8 shows an outperforming OSDT method, but it is mostly dued to the lack of the metric's capabilites and scope for such model.

Table 5.8: F8 Truthfulness scores

	Reality Check (F8.1)	Bias Detection (F8.2)	Total
LIME	0	1	1
KernelSHAP	0	1	1
DiCE	1	1	2
OSDT	2	1	**3**

5.10 F9 - Stability

5.10.1 F9 : Normative definition

This property ensure that XAI method is robust to small changes in the input. It is divided into two sub-property.

Similarity (F9.1)

The property checks whether similar points (neighbors), belonging to the same class, have similar explanations. This method for defining neighbors can vary based on the task and data type.

$$m_{f9.1} = \frac{1}{N}\sum_{i=1}^{N}\begin{cases}\frac{1}{n}\sum_{j=1}^{n}\frac{1}{1+d(e_i,e_i^{(j)})} & \text{if pairwise distance is meaningful}\\ \\ 1-\frac{1}{k}\sum_{l=1}^{k}\sigma_l/\mu_l & \text{otherwise}\end{cases} \tag{5.24}$$

Where :

- N is the number of evaluated instances
- n corresponds to the number of neighboring samples for each evaluated instance.
- e_i is the explanation generated for the reference instance i^{th} .
- $e_i^{(j)}$ is the explanation generated for the neighbor j or reference instance i.
- $d(e_i, e_i^{(j)})$ measure the pairwise distance between the reference explanations e_i and it's neighbors $e_i^{(j)}$
- k corresponds to the number of components in the explanation
- σ_l is the standard deviation of the l^{th} component across neighbors
- μ_l represent the mean of the l^{th} component across neighbors.

Identity (F9.2)

This property mesure the variability of the explanation method. For an identical input, rather than similar ones like in F9.1, the method is use multiple times to measure its consistency. A higher variability indicates greater instability in the XAI method. Identity metric isolates intrinsic stochasticity in the explanation algorithm itself.

$$m_{f9.2} = \frac{1}{N}\sum_{i=1}^{N}\begin{cases}\frac{1}{r}\sum_{h=1}^{r}\frac{1}{1+d(e_i^{(0)},e_i^{(h)})} & \text{if pairwise dist. is meaningful}\\ \\ 1-\frac{1}{k}\sum_{l=1}^{k}\sigma_l/\mu_l & \text{otherwise}\end{cases} \tag{5.25}$$

Where :

- N denotes the number of evaluated instances
- r represent the number of repeated runs for the same instance
- $e_i^{(h)}$ represents the explanation generated for the instance i at the h^{th} run.
- $e_i^{(0)}$ represents the reference explanation for the instance i.
- k corresponds to the number of components in the explanation
- σ_l is the standard deviation of the l^{th} component across runs
- μ_l represent the mean of the l^{th} component across runs.

5.10.2 F9 : Implementation and Results

The same underlying idea was used for all methods : A points are randomly sampled, called anchor points; and N nearest point based on the Gower distance were selected. The Gower distance is distance metric deisigned for datasets containing heterogeneous feature types. It defines the distance between two instances as the average of feature-wise dissimilarities, where each feature contributes a value in $[0, 1]$, ensuring comparability across scales and data types. In this work, feature-wise dissimilarties are computed as follows :

- Categorical features contribute 0 if values are equal and 1 otherwise.
- Ordinal features contribute the absolute difference between values, normalized by the feature's range.
- Continuous features contribute the absolute difference between values, normalized by the observed range.

The final distance is obtained by averaging these contributions across all fetures.
The points all belong to the same classes, "CAQ". It overestimates the true similarity across the full data set distribution, but is acceptable since all methods use the same standard.

F9.1 Similarity : scores

LIME (0.1), KernelSHAP (0.1) : Explanation similarity is evaluated by measuring the proximity of attribution or surrogate coefficient vectors across A neighborhoods containing N similar instances.
After a feature-wise normalization, similarity is computed between a reference instance and its neighbors using a distance-based similarity defined as follows :

$$\text{sim}_a = \frac{1}{N-1} \sum_{j=2}^{N} sim(e_1, e_j) \tag{5.26}$$

Where

$$e_i = \begin{cases} \frac{\phi(x_i)-\mu_a}{\sigma_a+\epsilon} & \text{For KernelSHAP,} \\ \\ \frac{\beta(x_i)-\mu_a}{\sigma_a+\epsilon} & \text{For LIME} \end{cases} \tag{5.27}$$

and μ_a is mean vector for all the features $f \in \mathcal{F}$ in batch a and σ_a is the standard deviation vectors for all features $f \in \mathcal{F}$ in the batch a and $\text{sim}(e_1, e_j) = \frac{1}{1+\|e_1-e_j\|_2}$. The similarity is averaged locally and then globally. This metric captures the stability of attribution patterns (KernelSHAP), and local surrogate models (LIME) under small input perturbations.

DiCE (0.4) : The similarity notion is not naturally defined for example-based explanations. For each batch a of similar instances, counterfactual explanations are generated independently. Since DiCE explicitly promotes diversity among genreated counterfactuals, a canonical counterfactual, i.e. the closest following Gower distance of the anchor point, per instance, is selected. Then, each canonical CF is represented by a binary feature-change vector $S_i = \{f \in \mathcal{F} | x_{i,f}^{cf} \neq x_{i,f}\}$ where f is a feature of the feature set $\mathcal{F} = \{1, ..., d\}$, indicating which features differ between the original instance i and its canonical counterfactual cf.

This representation captures which features must be modified, rather than the magnitude of those modifications, unquantifiable under DiCE. Similarity within neighborhood a is computed as the mean pairwise Jaccard similarity between the feature-change vectors. It measures the extent to which nearby instances require changing the same features to obtain the same class, Formally, for batch a of N instances, similarity is defined as :

$$\text{sim}_a = \frac{2}{N(N-1)} \sum_{1 \leq i < j \leq N} \frac{|S_i \cap S_j|}{|S_i \cup S_j|} \tag{5.28}$$

Where S_i the feature change-vector of instance i, S_j is the feature change-vector of instance j, we sum over $i < j$ to not take into consideration the same pairs, $\frac{2}{N(N-1)} = frac1\binom{N}{2}$ is the total number of terms. This

quantity measures the extent to which nearby instances require modifying the same features to obtain the same target class.

OSDT (0.8) : For the OSDT metric, the similarity of explanations is the similarity of decision paths. The distance between the original point's explanation, i.e. decision path, and every points of the batch is computed. The distance is $1 - L/M$, where L is the number of common split (same feature, and same boolean evaluation), between the two decision path and M is the maximum length decision path between the two. Explanations not sharing any common split are assigned a maximum distance of 1.

The similarty is computed across batch as follow and then average :

$$\text{sim}_a = \sum_{n=1}^{N} \mathbb{E}\Big(\frac{1}{d_n}\Big) \tag{5.29}$$

Where d_n is the distance between the anchor point's decision path and the n of the same batch a ,$a = 1, ..., A$ and is the number of batch, and $n = 1, ..., N - 1$ is the number of points per batch.
The final similarity is the average of similarities across batch.

F9.2 Identity : scores

LIME (0.1), KernelSHAP (0.2) : For KernelSHAP and LIME, which both rely on stochastic sampling procedures, repeated explanations are expected to exhibit variability. F9.2 quantifies this variability by measuring the dispersion of attribution or surrogate coefficient vectors across runs, thereby isolating intrinsic randomness from input-induced instability. The same data points were used as the A anchors points as in the previous section. Explanations are generated 10 times per instances.

DiCE (0.9) : Identity is assessed in terms of consistency of the counterfactual explanations generated for the same input across R runs. For this work, the identity metric is sub-divided into two sub-metrics.

The first notion is called feature-identity and focuses on feature-level consistency : whether DiCE consistenly identifies the sames features are requiring modification to change the prediciton. It executes the same steps as in the previous section, i.e. canonical CF into feature-change vector (see section 5.10.2). After the computation of the R feature-change vector, the metric is computed as the mean pairwise Jaccard similarty between these feature-change vectors :

$$\text{Id}_{\text{feature}}(x) = \frac{2}{R(R-1)} \sum_{1 \le k < \ell \le R} \frac{|S_k(x) \cap S_\ell(x)|}{|S_k(x) \cup S_\ell(x)|}. \tag{5.30}$$

Where S_k the feature change-vector of instance k, S_ℓ is the feature change-vector of instance ℓ, we sum over $i < j$ to not take into consideration the same pairs.
A high feature-identity score indicates that DiCE consistently modifies the same subset of features across repeated runs for a given instance, suggesting stable identification of relevant features.
The second metric is the distance-identity : it quantifoes how close, within a set of R repetitions, for one instance i, are its canonical CF. The pairwise distances aer computed using the Gower distance. The distance-identity score for given instance is obtained :

$$\bar{d}(x_i) = \frac{2}{R(R-1)} \sum_{1 \le k < \ell \le R} d_G\big(x_i^{cf,k}, x_i^{cf,\ell}\big) \tag{5.31}$$

Where $x_i^{cf,k}$ is k counterfactual cf of the instance i.

The mean distance $\bar{d}(x_i)$ is converted into a similarity-based score via :

$$\text{Id}_{\text{distance}}(x_i) = \frac{1}{1 + \bar{d}(x_i)} \tag{5.32}$$

The global score is obtained by averaging across all instances for which at least two canonical counterfactuals are availables. The final score for DiCE is :

$$\text{F9.2}_{\text{DiCE}} = \frac{1}{2}\text{Id}_{\text{feature}} + \frac{1}{2}\text{Id}_{\text{distance}} \tag{5.33}$$

This balanced aggregation reflects the dual nature of counterfactual explanations, which must be stable both in feature selection and in proposed actions.

OSDT (1) : For OSDT, explanations are deterministic and correspond exactly to the model's decision path. Therefore, the identity metric F9.2 is structurally maximized (equal to 1) for all instances, and does not provide additional discriminative power.

We report the maximum value by construction.

Summary In table 5.9, we can observe drastically different performance along methods. KernelSHAP and LIME, as expected, perform poorly due to its stochastic nature for one and its scope for the other. DiCE, despite it is stochastic nature, perform surprisingly well. The implementation choices might be a credible explanation for the unexpected success : they are implemented in a way to curb its diversity property and selecting only the closest CF generated. OSDT is most stable method under this framework, which can explain its deterministic nature.

Table 5.9: F9 Stability scores

	Similarity (F9.1)	Identity (F9.2)	Total
LIME	0.1	0.2	0.3
KernelSHAP	0.1	0.2	0.3
DiCE	0.4	0.9	1.3
OSDT	0.8	1.0	**1.8**

5.11 F10 - (Un)Certainty

5.11.1 F10 : Normative definition

This property provide the level of confidence in the XAI method's output to end-users, i.e. non ML-practitioners. It's score from 0 (no confidence measures) to 5 (fully transparent confidence reporting).

$$m_{f10} = \sum_{i=1}^{5} c_i \tag{5.34}$$

Where c_i corresponds to a binary indicator for the i^{th} confidence aspect :

- $c_1 = 1$: Confidence in the black-box model's result is reported (e.g. by displaying the model accuracy)
- $c_2 = 1$: Confidence in the XAI explanation is reported.
- $c_3 = 1$: Random processes in explanation generation are disclosed (e.g., sampling/random perturbation)

- $c_4 = 1$: The instance's distance from the training data distribution is indicated.
- $c_5 = 1$: Addition confidence indicators/measures are provided.

5.11.2 F10 : Implementation and Results

LIME (2) :

C1) The method report confidence in the black-box model's out. It display the probability of the outcome. Score of 1. *C2)* Confidence in the explanation not reported. *C3)* Random processes are not disclosed. *C4)* Distance from the training data distribution not indicated. *C5)* LIME provides de R^2 scores, the coefficient of determination. It represents the proportion of the variation in the dependent variable that is predictable from the independent variables.

KernelSHAP (1) :

C1) The method report confidence in the black-box model's out. It display the probability of the outcome. Score of 1. *C2)* Confidence in the explanation not reported. *C3)* Random processes are not disclosed. *C4)* Distance from the training data distribution not indicated. *C5)* No substantial additional confidence indicators.

DiCE (2) :

C1) It shows the explanation outputs. Score of 1. *C2)* Confidence in the explanation not reported. *C3)* Not openly disclosed. *C4)* Distance from the training data distribution not indicated, but distance from the decision boundary can be deduced. Score of 1. *C5)* No substantial additional confidence indicators.

OSDT (1) : *C1)* The explanation output does not explicitly report confidence measures (e.g. probabilty) associated with the predicted outcome. *C2)* While OSDT produces an optimality certificate under its theoretical framework, it does not provide user-facing measures quantifying the confidence of individuals explanations. *C3)* No random processes to be disclosed. Score of 1. *C4)* The instance's distance from the training data distribution is not indicated. *C5)* No substantial additional confidence indicators.

Summary The table 5.10 shows that DiCE and LIME project the highest level of confidence toward non-ML practitioners.

Table 5.10: F10 (Un)Certainty scores

	Total
LIME	**2**
KernelSHAP	1
DiCE	**2**
OSDT	1

5.12 F11 - Speed

5.12.1 F11 : Normative definition

This property assesses the computation time required by the XAI method to generate and explanation. Its purpose is to quantify the real-world readiness of the methods. The computation time is computed from initialization to the production of the first explanation.

$$m_{f11} = \begin{cases} 4 & t \leq 0.1\text{sec} \\ 3 & 0.1 < t \leq 1\text{sec} \\ 2 & 1 < t \leq 5\text{sec} \\ 1 & 5 < t \leq 10\text{sec} \\ 0 & t > 10\text{sec} \end{cases} \tag{5.35}$$

5.12.2 F11 : Implementation and Results

Summary The table 5.11 shows that LIME is quickest studied explainer, followed by DiCE. As expected KernelSHAP is the least performing post-hoc methods, due to its expensive approximation strategy and its guaranteed axioms. And the framework is not fitted for intrinsic interpretable model : we are comparing model training and explanation generation versus solely explanation generation.

Table 5.11: F11 Speed scores

	Total
LIME	**3**
KernelSHAP	1
DiCE	2
OSDT	0

Chapter 6

Results and Conclusion

6.1 Results

As already mentioned in the beginning of section 5, the results are to be understand as what they are : a comparison between XAI methods belonging to different model's families, under a benchmark framework primarly thought for post-hoc attribution based methods, in the context of this classification task on (mostly) categorical data and not-so-accurate predictions models. Some methods will be advantaged or disadvantaged depending on their nature and the context of the evaluation.
Therefore, all results should be interpreted with caution, not as ground truth, and takin in the aforementioned context.For comparison purpose, all the metric have been normalized using min-max scaling.

The benchmark reaveals, distinct, almost, non-overlapping performances profiles across methods, with no method dominating across all functional properties, figure 6.1 illustrates it.

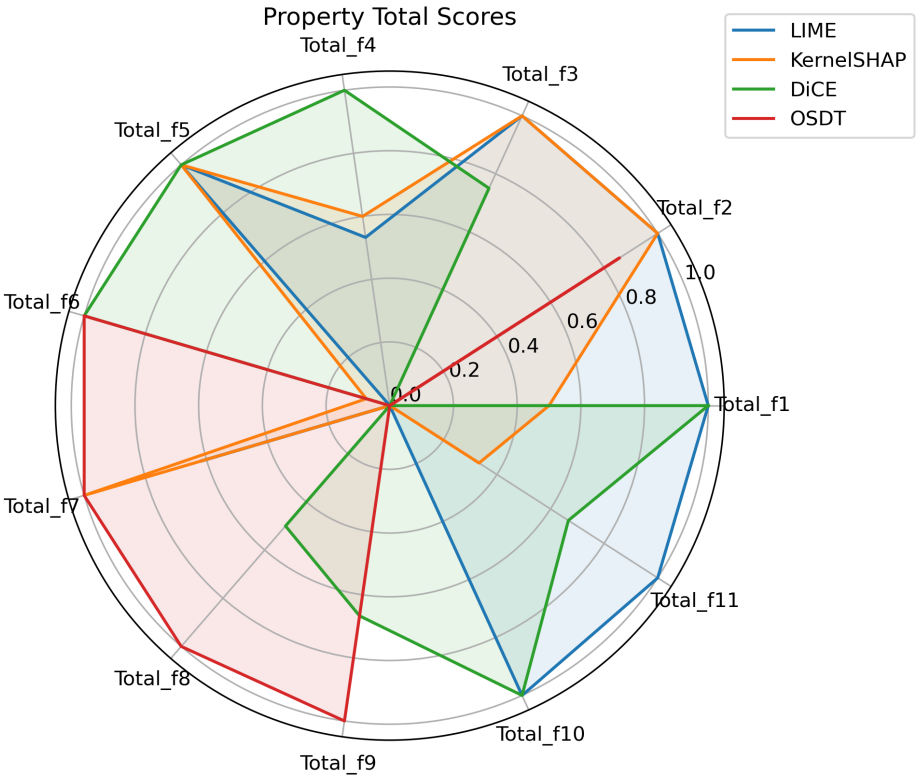


Figure 6.1

The two plots 6.2 shows the total scores for SHAP and LIME on the left and DiCE and OSDT on the right. It perfectly illustrates the domain of expertise of certain methods : LIME and KernelSHAP perform well on the Representativeness(F1), Selectivity(F3), Structure (F2), Speed (F11) and Certainty (F10) metrics, which can be labelled as "Communication-oriented properties" ; OSDT performs well on "Model-alignement" properties : the Fidelity (F6), Faithfulness (F7), Truthfulness (F8) and Stability (F9) ; and DiCE on constrastive reasoning (F4).
The benchmark underlines intuitives knowledge linked to the methods' structure : OSDT is more stable than LIME and SHAP ; DiCE gives the more contrastive information. It is a sign that the benchmark is able to effectively captures the strengths and weaknesess of methods in its current shape.

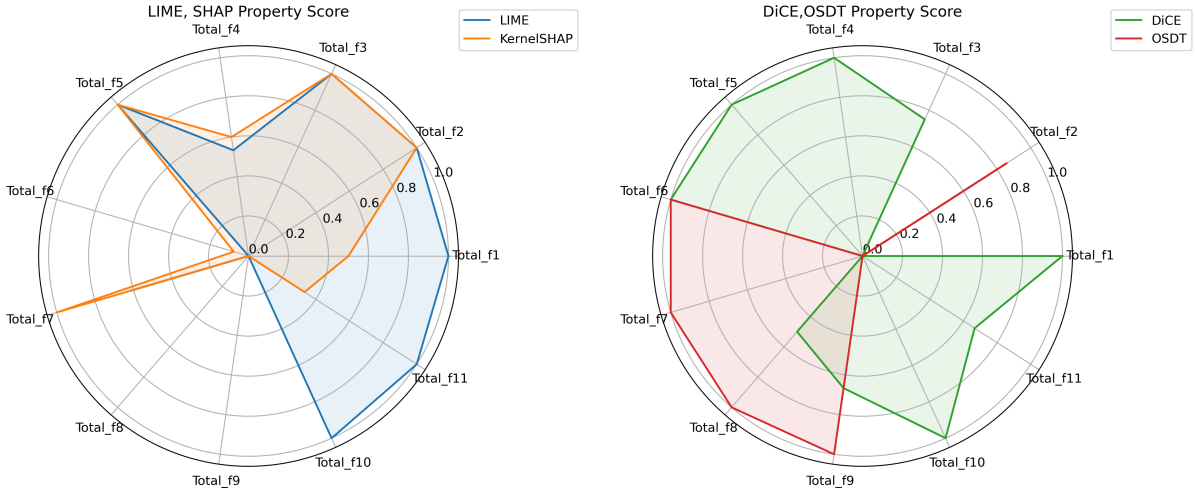


Figure 6.2: Comparison of the predictions and intentions of voters

6.2 Discussion

This work applied a functionally ground benchmark to compare four XAI approaches that produces different explanatory objects (attributions scores, counterfactual instances, and decision-path logic). In this setting — multi-class prediciton on an almost fully categorical tabular dataset— the benchmark does not induce a single ranking of methods. Instead, it reveals distinct performance profiles, aligned with different functional goals and architecture, supporting the view that interpretability is a multi-faceted rather than reducible to a single scalar score.

The Stability property (F9) effectively separates methods with intrinsic stochasticity (or locality dependence) from deterministic explanation mechanisms. The Identity and Similarity sub-metrics produce a clear ranking from attribution-based methods (LIME/KernelShap) to intrinsic interpretable model (OSDT), consistent with the fact that local surrogate fitting and sampling-based approximations introduce variability across runs. In contrast, OSDT explanations are uniquely determined by the learned decision structure and the input instance.
Hower, DiCE's second place should be interpreted in the light of the implementation choice to select a canonical "closest counterfactual", which curb its diversity objective.
Furthermore, the Constrastivity property (F4), as mentioned in the previous section, behaves as intented in the sense that it isolates the explanatory paradigm that explicitly encodes alternatives : DiCE.

Despite being presented as general-purpose for post-hoc methods, several properties appears to have been formulated with post-hoc attribution outputs in mind, and this assumption becomes visible when applying them example-baed explanations. For instance, F2 (Structure) rewards graphical integrity via signed positive/negative contributions.

Interactivity (F5) appears to lack discriminability with is near-constant values across methods in this study. It is largely because the scoring captures whether some form of parameter control and *potential* interface implementation (e.g. API) are present.

Finally, several methodological constraints limit the scope of the numerical comparisons. Faithfulness (F7) and stability (F9) were evaluated only for one class due to both time and sample size constraints. ROAR (F7.2) was not computed in this setting, reducing the reach of F7's claims.
Parts of the evaluation rely on *paradigm translations* (e.g. mapping counterfactual changes to feature importances proxies), which are necessare for cross-method comparison but introduce additional assumptions.
Furthermore, the uncertainty introduced by the modest accuracy of the models, mitigated by choice to analyze points in which the model was highly confident of, adds another layer of cautions with the results of this analysis.
The models were trained on a single training-testing split. The perfect modus operandi would have been to add one black-box, one white box models ; train the models ; effectuate my whole analysis, for every class, for different training-test splits; repeat several times on each splits to remove the bias introduced but the splitting choice.

Interpreted through the AI HLEG [20] transparency framing (Explainability, Traceability, COmmunication), the benchmark results translate into practical selection guidance rather than a global ranking of methods. In this case study, communication-oriented properties — captured by F2 or F3— favor attribution-based methods or example-based methods. Whereas explainability, as model alignement — capturing Fidelity/Faithfulness/stability-type criteria — favors OSDT and decision path logic. Traceability is only indirectly implemented by the benchmark metrics and is instead supported here by the exploration of of the data and preprocessing transparency developed in earlier chapters.
The results do not revolutionaze our understanding, but instead offers guidance via a *concret* case analysis of the benchmark that aligns with real-world expectations.

6.3 Conclusion

This thesis compared *LIME*, *KernelSHAP*, *DiCE*, and an intrinsic interpretable model (*OSDT via GOSDT* solver) under a functionally grounded benchmark in a multi-class classification setting on an almost fully categorical tabular dataset. The evaluation does not yield a universally dominant method. Rather than being inconclusive, this outcome provides evidence that interpretability properties are *trade-off driven*, *dependent on the explanatory paradigm*—i.e., on whether explanations are expressed as feature contributions, counterfactual instances, or model-intrinsic decision logic. **Consequently, the benchmark's result supports goal-conditioned method selection rather than global ranking.**
Returning to the motivating application—predicting political-party alignment from lifestyle and demographic survey features—the results suggest that methods producing concise, communicable, instance-level summaries are best aligned with communication toward non-expert users. In this benchmark instantiation, attribution-based explanations (LIME/KernelSHAP) score higher on communication-facing properties such as representational structure and tunable selectivity, while counterfactual explanations (DiCE) provide the most explicit contrastive information by articulating how an outcome could change. In contrast, intrinsic explanations (OSDT) are most aligned with activity requiring high stability, exhibiting strong performance on model-alignment and reliability-related properties, including stability and faithfulness proxies.
More broadly, the benchmark does not collapse heterogeneous explanation types into a single ordering; it separates them into distinct functional profiles that correspond to different transparency goals. Taken together, the results support interpretability as a multi-faceted construct: the studied methods do not compete along a single latent "interpretability" axis, but instead realize different functional goals—*communication*, *contrastive reasoning/recourse*, and *model-aligned transparency*. This framing reconciles the absence of a single winner with the

fact that each method remains valuable under the goal it is designed to serve.
Finally, several limitations bound the strength and generality of the numerical comparisons. Some benchmark properties are weakly discriminative in practice (e.g., interactivity) or become partially non-diagnostic for certain paradigms, and cross-family method comparison sometimes requires semantic reinterpretation rather than mechanical metric reuse.

Bibliography

[1] "(PDF) Comprehensive Review of Deep Reinforcement Learning Methods and Applications in Economics". In: *ResearchGate* (Dec. 9, 2024). DOI: 10.3390/math8101640. URL: https://www.researchgate.net/publication/344351321_Comprehensive_Review_of_Deep_Reinforcement_Learning_Methods_and_Applications_in_Economics (visited on 02/06/2025).

[2] *2. Over-sampling — Version 0.13.0*. URL: https://imbalanced-learn.org/stable/over_sampling.html#naive-random-over-sampling (visited on 07/19/2025).

[3] *2. Over-sampling — Version 0.15.Dev0*. URL: https://imbalanced-learn.org/dev/over_sampling.html (visited on 12/13/2025).

[4] *A Liberal Plan for an Assertive, United, and Prosperous Quebec*. URL: https://plq.org/en/press-release/a-liberal-plan-for-an-assertive-united-and-prosperous-quebec/ (visited on 10/08/2025).

[5] Amina Adadi and Mohammed Berrada. "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access* 6 (2018), pp. 52138–52160. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2870052. URL: https://ieeexplore.ieee.org/document/8466590 (visited on 02/21/2025).

[6] Sajid Ali et al. "Explainable Artificial Intelligence (XAI): What We Know and What Is Left to Attain Trustworthy Artificial Intelligence". In: *Information Fusion* 99 (Nov. 1, 2023), p. 101805. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2023.101805. URL: https://www.sciencedirect.com/science/article/pii/S1566253523001148 (visited on 02/21/2025).

[7] Elaine Angelino et al. *Learning Certifiably Optimal Rule Lists for Categorical Data*. Aug. 3, 2018. DOI: 10.48550/arXiv.1704.01701. arXiv: 1704.01701 [stat]. URL: http://arxiv.org/abs/1704.01701 (visited on 11/04/2025). Pre-published.

[8] Elaine Angelino et al. "Learning Certifiably Optimal Rule Lists for Categorical Data". In: ().

[9] *Artificial Intelligence*. In: *Wikipedia*. Feb. 2, 2025. URL: https://en.wikipedia.org/w/index.php?title=Artificial_intelligence&oldid=1273565751 (visited on 02/06/2025).

[10] *Artificial Intelligence and Digitalisation of Judicial Cooperation*. URL: https://eucrim.eu/articles/artificial-intelligence-and-digitalisation-of-judicial-cooperation/ (visited on 02/06/2025).

[11] Andrew Bell et al. "It's Just Not That Simple: An Empirical Study of the Accuracy-Explainability Trade-off in Machine Learning for Public Policy". In: *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '22. New York, NY, USA: Association for Computing Machinery, June 20, 2022, pp. 248–266. ISBN: 978-1-4503-9352-2. DOI: 10.1145/3531146.3533090. URL: https://dl.acm.org/doi/10.1145/3531146.3533090 (visited on 12/12/2025).

[12] Matthieu Bellucci et al. "Towards a Terminology for a Fully Contextualized XAI". In: *Procedia Computer Science* 192 (2021), pp. 241–250. ISSN: 18770509. DOI: 10.1016/j.procs.2021.08.025. URL: https://linkinghub.elsevier.com/retrieve/pii/S187705092101512X (visited on 12/11/2025).

[13] Leo Breiman. “Statistical Modeling: The Two Cultures (with Comments and a Rejoinder by the Author)”. In: *Statistical Science* 16.3 (Aug. 2001), pp. 199–231. ISSN: 0883-4237, 2168-8745. DOI: 10.1214/ss/1009213726. URL: https://projecteuclid.org/journals/statistical-science/volume-16/issue-3/Statistical-Modeling--The-Two-Cultures-with-comments-and-a/10.1214/ss/1009213726.full (visited on 07/09/2025).

[14] Dulce Canha et al. “A Functionally-Grounded Benchmark Framework for XAI Methods: Insights and Foundations from a Systematic Literature Review”. In: *ACM Comput. Surv.* 57.12 (July 14, 2025), 320:1–320:40. ISSN: 0360-0300. DOI: 10.1145/3737445. URL: https://dl.acm.org/doi/10.1145/3737445 (visited on 12/11/2025).

[15] Chaofan Chen et al. “This Looks Like That: Deep Learning for Interpretable Image Recognition”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/adf7ee2dcf142b0e11888e72b43fcb75-Abstract.html (visited on 12/12/2025).

[16] Eunsuk Chong, Chulwoo Han, and Frank C. Park. “Deep Learning Networks for Stock Market Analysis and Prediction: Methodology, Data Representations, and Case Studies”. In: *Expert Systems with Applications* 83 (Oct. 15, 2017), pp. 187–205. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2017.04.030. URL: https://www.sciencedirect.com/science/article/pii/S0957417417302750 (visited on 02/06/2025).

[17] Evangelia Christodoulou et al. “A Systematic Review Shows No Performance Benefit of Machine Learning over Logistic Regression for Clinical Prediction Models”. In: *Journal of clinical epidemiology* 110 (2019), pp. 12–22. ISSN: 0895-4356. DOI: 10.1016/j.jclinepi.2019.02.004.

[18] *Coalition Avenir Québec*. In: *Wikipedia*. Oct. 7, 2025. URL: https://en.wikipedia.org/w/index.php?title=Coalition_Avenir_Qu%C3%A9bec&oldid=1315543357 (visited on 10/08/2025).

[19] *CORELS: Learning Certifiably Optimal RulE ListS*. URL: https://corels.cs.ubc.ca/corels/ (visited on 11/04/2025).

[20] *Ethics Guidelines for Trustworthy AI | Shaping Europe’s Digital Future*. URL: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai (visited on 12/12/2025).

[21] Petko Georgiev et al. “Low-Resource Multi-task Audio Sensing for Mobile and Embedded Devices via Shared Deep Neural Network Representations”. In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1.3 (Sept. 11, 2017), 50:1–50:19. DOI: 10.1145/3131895. URL: https://doi.org/10.1145/3131895 (visited on 02/06/2025).

[22] Sofie Goethals. “The Non-Linear Nature of the Cost of Comprehensibility”. In: (2022).

[23] Micah Goldblum et al. *The No Free Lunch Theorem, Kolmogorov Complexity, and the Role of Inductive Biases in Machine Learning*. June 7, 2024. DOI: 10.48550/arXiv.2304.05366. arXiv: 2304.05366 [cs]. URL: http://arxiv.org/abs/2304.05366 (visited on 07/09/2025). Pre-published.

[24] Michael Greenacre and Raul Primicerio. *Multivariate Analysis of Ecological Data*. Bilbao: Fundación BBVA, 2013. 331 pp. ISBN: 978-84-92937-50-9.

[25] Robert C. Holte. “Very Simple Classification Rules Perform Well on Most Commonly Used Datasets”. In: *Machine Learning* 11.1 (Apr. 1, 1993), pp. 63–90. ISSN: 1573-0565. DOI: 10.1023/A:1022631118932. URL: https://doi.org/10.1023/A:1022631118932 (visited on 07/09/2025).

[26] Xiyang Hu, Cynthia Rudin, and Margo Seltzer. *Optimal Sparse Decision Trees*. Sept. 26, 2023. DOI: 10.48550/arXiv.1904.12847. arXiv: 1904.12847 [cs]. URL: http://arxiv.org/abs/1904.12847 (visited on 11/17/2025). Pre-published.

[27] *Interpretable Machine Learning*. URL: https://christophm.github.io/interpretable-ml-book/ (visited on 09/10/2025).

[28] Ulf Johansson et al. “Trade-Off Between Accuracy and Interpretability for Predictive In Silico Modeling”. In: *Future medicinal chemistry* 3 (Apr. 1, 2011), pp. 647–63. DOI: 10.4155/fmc.11.23.

[29] Meng Li et al. “Shapley Value: From Cooperative Game to Explainable Artificial Intelligence”. In: *Autonomous Intelligent Systems* 4.1 (Feb. 9, 2024), p. 2. ISSN: 2730-616X. DOI: 10.1007/s43684-023-00060-8. URL: https://doi.org/10.1007/s43684-023-00060-8 (visited on 12/11/2025).

[30] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. Nov. 25, 2017. DOI: 10.48550/arXiv.1705.07874. arXiv: 1705.07874 [cs]. URL: http://arxiv.org/abs/1705.07874 (visited on 11/04/2025). Pre-published.

[31] *Machine Learning*. In: *Wikipedia*. Feb. 3, 2025. URL: https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=1273675406 (visited on 02/06/2025).

[32] Ričards Marcinkevičs and Julia E. Vogt. “Interpretable and Explainable Machine Learning: A Methods-Centric Overview with Concrete Examples”. In: *WIREs Data Mining and Knowledge Discovery* 13.3 (2023), e1493. ISSN: 1942-4795. DOI: 10.1002/widm.1493. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1493 (visited on 02/21/2025).

[33] Riccardo Miotto et al. “Deep Learning for Healthcare: Review, Opportunities and Challenges”. In: *Briefings in Bioinformatics* 19.6 (May 6, 2017), pp. 1236–1246. ISSN: 1467-5463. DOI: 10.1093/bib/bbx044. PMID: 28481991. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6455466/ (visited on 02/06/2025).

[34] Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. “Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations”. In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. Jan. 27, 2020, pp. 607–617. DOI: 10.1145/3351095.3372850. arXiv: 1905.07697 [cs]. URL: http://arxiv.org/abs/1905.07697 (visited on 09/10/2025).

[35] Meike Nauta et al. “From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI”. In: *ACM Comput. Surv.* 55 (13s July 13, 2023), 295:1–295:42. ISSN: 0360-0300. DOI: 10.1145/3583558. URL: https://dl.acm.org/doi/10.1145/3583558 (visited on 02/21/2025).

[36] *Nos Valeurs – Parti Conservateur Du Québec*. URL: https://www.conservateur.quebec/parti/nos-valeurs/ (visited on 10/08/2025).

[37] *Notion – The all-in-one workspace for your notes, tasks, wikis, and databases.* Notion. URL: https://www.notion.so (visited on 02/06/2025).

[38] *Page Non Trouvée – Parti Conservateur Du Québec*. URL: https://www.conservateur.quebec/liberte_22_economie?utm_source=chatgpt.com (visited on 10/08/2025).

[39] *Parti Québécois*. In: *Wikipedia*. Sept. 15, 2025. URL: https://en.wikipedia.org/w/index.php?title=Parti_Qu%C3%A9b%C3%A9cois&oldid=1311396573 (visited on 10/08/2025).

[40] Maximilian Pichler and Florian Hartig. “Machine Learning and Deep Learning—A Review for Ecologists”. In: *Methods in Ecology and Evolution* 14.4 (2023), pp. 994–1016. ISSN: 2041-210X. DOI: 10.1111/2041-210X.14061. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.14061 (visited on 12/12/2025).

[41] *Québec Solidaire*. In: *Wikipedia*. Sept. 28, 2025. URL: https://en.wikipedia.org/w/index.php?title=Qu%C3%A9bec_solidaire&oldid=1313934325 (visited on 10/08/2025).

[42] *Résultats des élections générales*. Élections Québec. Nov. 25, 2021. URL: https://www.electionsquebec.qc.ca/resultats-et-statistiques/resultats-generales/2022-10-03/ (visited on 10/13/2025).

[43] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. Aug. 9, 2016. DOI: 10.48550/arXiv.1602.04938. arXiv: 1602.04938 [cs]. URL: http://arxiv.org/abs/1602.04938 (visited on 12/11/2025). Pre-published.

[44] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Anchors: High-Precision Model-Agnostic Explanations". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 25, 2018). ISSN: 2374-3468. DOI: 10.1609/aaai.v32i1.11491. URL: https://ojs.aaai.org/index.php/AAAI/article/view/11491 (visited on 09/10/2025).

[45] Benedek Rozemberczki et al. *The Shapley Value in Machine Learning*. May 26, 2022. DOI: 10.48550/arXiv.2202.05594. arXiv: 2202.05594 [cs]. URL: http://arxiv.org/abs/2202.05594 (visited on 12/11/2025). Pre-published.

[46] Cynthia Rudin. *Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead*. Sept. 22, 2019. DOI: 10.48550/arXiv.1811.10154. arXiv: 1811.10154 [stat]. URL: http://arxiv.org/abs/1811.10154 (visited on 12/12/2025). Pre-published.

[47] Cynthia Rudin et al. *Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges*. July 10, 2021. DOI: 10.48550/arXiv.2103.11251. arXiv: 2103.11251 [cs]. URL: http://arxiv.org/abs/2103.11251 (visited on 07/09/2025). Pre-published.

[48] Cynthia Rudin et al. "Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges". In: *Statistics Surveys* 16 (none Jan. 2022), pp. 1–85. ISSN: 1935-7516. DOI: 10.1214/21-SS133. URL: https://projecteuclid.org/journals/statistics-surveys/volume-16/issue-none/Interpretable-machine-learning-Fundamental-principles-and-10-grand-challenges/10.1214/21-SS133.full (visited on 02/07/2025).

[49] Cynthia Rudin et al. *Amazing Things Come From Having Many Good Models*. July 10, 2024. DOI: 10.48550/arXiv.2407.04846. arXiv: 2407.04846 [cs]. URL: http://arxiv.org/abs/2407.04846 (visited on 12/11/2025). Pre-published.

[50] Lesia Semenova, Cynthia Rudin, and Ronald Parr. "On the Existence of Simpler Machine Learning Models". In: *2022 ACM Conference on Fairness Accountability and Transparency*. June 21, 2022, pp. 1827–1858. DOI: 10.1145/3531146.3533232. arXiv: 1908.01755 [cs]. URL: http://arxiv.org/abs/1908.01755 (visited on 12/13/2025).

[51] *Sparsity - an Overview | ScienceDirect Topics*. URL: https://www.sciencedirect.com/topics/computer-science/sparsity (visited on 02/07/2025).

[52] *What Is Deep Learning? | IBM*. June 17, 2024. URL: https://www.ibm.com/think/topics/deep-learning (visited on 02/06/2025).

[53] *What Is Machine Learning (ML)? | IBM*. Sept. 22, 2021. URL: https://www.ibm.com/think/topics/machine-learning (visited on 02/06/2025).

[54] Ian R. White, Patrick Royston, and Angela M. Wood. "Multiple Imputation Using Chained Equations: Issues and Guidance for Practice". In: *Statistics in Medicine* 30.4 (2011), pp. 377–399. ISSN: 1097-0258. DOI: 10.1002/sim.4067. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.4067 (visited on 03/15/2025).

[55] Mengwei Xu et al. "A First Look at Deep Learning Apps on Smartphones". In: *The World Wide Web Conference*. WWW '19: The Web Conference. San Francisco CA USA: ACM, May 13, 2019, pp. 2125–2136. ISBN: 978-1-4503-6674-8. DOI: 10.1145/3308558.3313591. URL: https://dl.acm.org/doi/10.1145/3308558.3313591 (visited on 02/07/2025).

[56] Yujia Zhang et al. *"Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations*. June 4, 2019. DOI: 10.48550/arXiv.1904.12991. arXiv: 1904.12991 [cs]. URL: http://arxiv.org/abs/1904.12991 (visited on 12/11/2025). Pre-published.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
Faculté des sciences
Place des Sciences, 2 bte L6.06.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/sc