Model-Agnostic Counterfactual Explanations for Consequential Decisions

Amir-Hossein Karimi MPI-IS*
Gilles Barthe MPI-SP/IMDEA Software Institute
Borja Balle† -
Isabel Valera MPI-IS*

Abstract

Predictive models are being increasingly used to support consequential decision making at the individual level in contexts such as pretrial bail and loan approval. As a result, there is increasing social and legal pressure to provide explanations that help the affected individuals not only to understand why a prediction was output, but also how to act to obtain a desired outcome. To this end, several works have proposed optimization-based methods to generate *nearest counterfactual explanations*. However, these methods are often restricted to a particular subset of models (e.g., decision trees or linear models) and differentiable distance functions. In contrast, we build on standard theory and tools from formal verification and propose a novel algorithm that solves a sequence of satisfiability problems, where both the distance function (objective) and predictive model (constraints) are represented as logic formulae. As shown by our experiments on real-world data, our algorithm is: i) *model-agnostic* ({non-}linear, {non-}differentiable, {non-}convex); ii) *data-type-agnostic* (heterogeneous features); iii) *distance-agnostic* ($\ell_0, \ell_1, \ell_\infty$, and combinations thereof); iv) able to generate plausible and diverse counterfactuals for any sample (i.e., *100% coverage*); and v) at *provably optimal distances*.

1 Introduction

Data-driven predictive models are ubiquitously being used to support or even substitute humans in decision making in a wide variety of real-world contexts including, e.g., selection process for hiring, loan approval, or pretrial bail. However, as algorithmic methods are increasingly used to make *consequential decisions* at the individual-level – i.e., decisions that may have significant consequences for the individuals they decide about – the debate about their lack of transparency and explainability becomes more heated. To make things worse, while the verdict is still out as to what constitutes a *good explanation* [Doshi-Velez and Kim, 2017, Freitas, 2014, Kodratoff, 1994, Murdoch et al., 2019, Lipton, 2018, Rudin, 2018, Rüping, 2006], there already exists clearly defined legal requirements for explanations in the context of consequential decision making. For example, the EU General Data Protection Regulation ("GDPR") grants individuals the *right-to-explanation* [Voigt and Von dem Bussche, , Wachter et al., 2017a], via requiring institutions to provide explanations to individuals that are subject to their (semi-)automated decision making systems.

A growing number of works on interpretable machine learning have recently focused on the definitions of, and mechanisms for providing, good explanations for predictor-based decision making systems. In the context of consequential decision making, it is widely agreed that a good explanation should provide answers to the following two questions [Doshi-Velez and Kim, 2017, Gunning, 2019, Wachter et al., 2017b]: (i) "*why the model outputs a certain prediction for a given individual?*"; and, (ii) "*what features describing the individual would need to change to achieve the desired output?*"

Here, we focus on answering the second question, or equivalently, on generating *counterfactual explanations*. Of specific importance is the problem of finding the *nearest counterfactual explanation* – i.e., identifying the set of features resulting in the desired prediction while remaining at minimum distance from the original set of features describing the individual. Existing approaches tackling this problem suffer from various limitations: they either propose solutions that are tailored to particular mod-

* MPI for Intelligent Systems, Tübingen.
† Now at DeepMind.

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

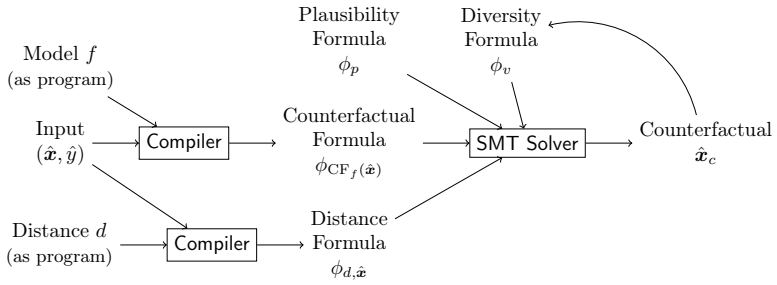


Figure 1: Architecture Overview for Model-Agnostic Counterfactual Explanations (MACE)

els, e.g., decision trees [Tolomei et al., 2017]; rely on classical optimization tools, thus being restricted to convex predictive models and distances [Russell, 2019, Ustun et al., 2019]; or, solve a relaxed version of the original optimization problem using gradient-based approaches, thus being restricted to differentiable models and distance functions [Wachter et al., 2017b] and lacking optimality guarantees. Additionally, it is important to consider that in the context of consequential decision-making, the features describing individuals are semantically meaningful and heterogeneous (i.e., mixed continuous & discrete); and can either be acted upon (e.g., bank account balance), or immutable and should be safeguarded from change (e.g., sex, race). A good explanation should account for these semantics (i.e., be *plausible*[1]) to be useful for the individual, a requirement that most existing approaches fail to address.

Our contributions. In this paper, we propose a *model-agnostic* approach to generate nearest counterfactual explanations, namely MACE, under any given *distance function* (or convex combinations thereof); while, at the same time, easily supporting additional *plausibility* constraints. Moreover, our approach readily encodes natural notions of distance for *heterogeneous feature* spaces, which are common in consequential decision making systems (e.g., loan approval) and consist of mixed numerical (e.g., age and income) and nominal features (e.g., gender and education level). To this end, in MACE we map the nearest counterfactual problem into a sequence of *satisfiability* (SAT) problems, by expressing both the predictive model and the distance function (as well as the plausibility and diversity constraints) as logic formulae. Each of these satisfiability problems aims to verify if there exists a counterfactual explanation at a distance smaller than a given threshold, and can be solved using standard SMT (satisfiability modulo theories) solvers. Moreover, we rely on a binary search strategy on the distance threshold to find an approximation to the nearest (plausible) counterfactual with an *arbitrary degree of accuracy*, and a lower bound on distance such that no counterfactual provably exists at a smaller distance. Finally, once nearest counterfactuals are found, diversity constraints may be added to the satisfiability problems to find alternative counterfactuals. The overall architecture of MACE is illustrated in Figure 1.

Our experimental validation on real-world datasets show that MACE not only achieves 100% coverage by design, but also generates explanations that are significantly closer than previous approaches [Tolomei et al., 2017, Ustun et al., 2019]. We also provide qualitative examples showcasing the flexibility of our approach to generate actionable counterfactuals by extending our plausibility constraints to restrict changes to a subset of (non-immutable) features. The Python implementation of our algorithms and the datasets used in our experiments are available at `https://github.com/amirhk/mace`.

2 First-order predicate logic

In this section, we briefly recall basic concepts of first-order predicate logic, which MACE builds upon. We distinguish between *function symbols* (for instance, addition $+$ and multiplication $\times$) and *predicate symbols* (for instance, equality $=$ or lesser than $<$). Function symbols are used to build *expressions*, and predicate symbols are used to build *atomic formulae*. Examples of valid expressions are x, $x+2$, $(-x)+2$ and $(x+2)\times(y+3)$. Examples of valid atomic formulae are $e < e'$, $e \leq e'$ or $e = e'$. A (quantifier-free) *formula* is a Boolean combination of atomic formulae. That is, a formula is built from atomic formulae using conjunction $\wedge$, disjunction $\vee$, and negation $\neg$. Formulae have an *interpretation* over their intended domain. For instance, a formula about real-valued expressions has a natural interpretation as a subset of $\mathbb{R}^n$, where n denotes the number of variables that appear in the formula. The interpretation is obtained by mapping every variable into a value, e.g., a real number. For example, $(2, 1)$ belongs

[1]We emphasize that while our formulation for generating counterfactuals seems similar to that of adversarial perturbations (image domain), the goals are different: while our goal is to provide actionable and plausible counterfactuals, the goal of adversarial examples is to be imperceptible to humans and hence plausible in the human-perception space, but not in the data space.

in the interpretation of $(x+2)\times(y+3)\leq x\times y+16$ since the mapping $x\mapsto 2, y\mapsto 1$ assigns true because $16\leq 18$. We say that a formula is *satisfiable* if its interpretation as a subset of $\mathbb{R}^n$ is non-empty.

The *satisfiability problem* consists in checking whether or not a formula is satisfiable. Satisfiability problems can be verified automatically using *satisfiability modulo theories* (SMT) solvers like Z3 [de Moura and Bjørner, 2008] or CVC4 [Barrett et al., 2011]. We refer to [Kroening and Strichman, 2008] for an exposition of the basic algorithms used by SMT solvers. For the purpose of the next sections, it suffices to assume a given *satisfiability oracle* SAT. For our experiments, we use off-the-self SMT solvers to realize the oracle. We use SMT solvers as black-box, but it is interesting to note that our formulae fall in the linear fragment of the theory of reals (i.e. all formulae that only contain expressions of degree 1 when viewed as multi-variate polynomials over variables), which can be decided efficiently using the Fourier-Motzkin algorithm.

3 Counterfactual spaces for predictive models

This section defines a logical representation of counterfactual explanations for predictive models, which are functions mapping input feature vectors $\boldsymbol{x}\in\mathcal{X}$ into decisions $y\in\{0,1\}$. [2] Given a predictive model $f:\mathcal{X}\to\{0,1\}$, we can define the *set of counterfactual explanations* for a (factual) input $\hat{\boldsymbol{x}}\in\mathcal{X}$ as $\mathrm{CF}_f(\hat{\boldsymbol{x}})=\{\boldsymbol{x}\in\mathcal{X}\mid f(\boldsymbol{x})\neq f(\hat{\boldsymbol{x}})\}$. In words, $\mathrm{CF}_f(\hat{\boldsymbol{x}})$ contains all the inputs x for which the model f returns a prediction different from $f(\hat{\boldsymbol{x}})$. We also remark that $\mathrm{CF}_f(\hat{\boldsymbol{x}})$ is the set of preimages of $1-f(\hat{\boldsymbol{x}})$ under f.

For a broad class of predictive models, it is possible to construct *counterfactual formulae* capturing membership in CF_f. We do so by computing the characteristic formula ϕ_f of the model. For a predictive model $f:\mathcal{X}\to\{0,1\}$, and pair of input and output values $\boldsymbol{x}$ and y, the *characteristic formula* ϕ_f verifies that $\phi_f(\boldsymbol{x},y)$ is valid if and only if $f(\boldsymbol{x})=y$. Thus, given a factual input $\hat{\boldsymbol{x}}$ with $f(\hat{\boldsymbol{x}})=\hat{y}$ and ϕ_f we define the *counterfactual formula* as

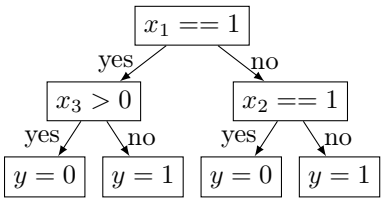
$$\phi_{\mathrm{CF}_f(\hat{\boldsymbol{x}})}(\boldsymbol{x})=\phi_f(\boldsymbol{x},1-\hat{y}) \tag{1}$$

Intuitively, the formula on the right hand side of (1) says that "$\boldsymbol{x}$ is a counterfactual for $\hat{\boldsymbol{x}}$ if either $f(\hat{\boldsymbol{x}})=0$ and $f(\boldsymbol{x})=1$, or $f(\hat{\boldsymbol{x}})=1$ and $f(\boldsymbol{x})=0$". It is thus clear from the definition that an input $\boldsymbol{x}$ satisfies $\phi_{\mathrm{CF}_f(\hat{\boldsymbol{x}})}$ if and only if $\boldsymbol{x}\in\mathrm{CF}_{f(\hat{\boldsymbol{x}})}$. Moreover, (1) shows that, to construct counterfactual formulae $\phi_{\mathrm{CF}_f(\hat{\boldsymbol{x}})}$, we only require the characteristic formulae of the corresponding predictive models, ϕ_f, and the value of $\hat{y}$. To obtain such characteristic formulae we assume that predictive models are represented by programs in a core programming language with assignments, conditionals, sequential composition, syntactically bounded loops and return statements. This allows us to use techniques from the program verification literature. Specifically, we use the so-called predicate transformers [Dijkstra, 1968, Hoare, 1969, Floyd, 1993, Flanagan and Saxe, 2001]. The description of the general procedure is provided in Appendix A. For ease of exposition, we illustrate the construction of characteristic formulae through two examples, a decision tree and a multilayer perceptron.

As a first example, consider the decision tree from Figure 2a which takes as input $(x_1,x_2,x_3)\in\{0,1\}^2\times\mathbb{R}$ and returns a binary output in $\{0,1\}$. Figure 2b provides the programming language description of this decision tree. To construct a formula representing the function $f(x)=y$ computed by this tree we first build a clause for each leaf in the tree by taking the conjunction of all the conditions encountered in the path from the root to the leaf. For example, the clause corresponding to the leftmost leaf on the tree in Figure 2a is $(x_1=1\wedge x_3>0\wedge y=0)$. Once all these clauses are constructed, the characteristic formula $\phi_f(\boldsymbol{x},y)$ corresponding to the full tree is obtained by taking the conjunction of all said clauses, as shown in Figure 2c.

As a second example we consider a feed-forward neural network with one hidden layer followed by a ReLU activation function, as depicted in Figure 3a. This model implements a function $f:\mathbb{R}^3\to\{0,1\}$, where the binary decision is taken by thresholding the value of the last hidden node. The programming language representation of this model is given in Figure 3b. In this case, the characteristic formula predicates over inputs $\boldsymbol{x}$, output y and program variables z_i and $\tilde{z}_i$ for each hidden node i representing the values on that node before and after the non-linear ReLU transformation, respectively. The characteristic formula is a conjunction, and each conjunct corresponds to one instruction of the program. For example, for the leftmost hidden node in the first layer of the network in Figure 3a the variable z_1 is associated with the clause $(z_1=x_1-x_2)$; and the variable $\tilde{z}_1$ corresponds to the value of z_1 after the ReLU, which can be written as the disjunction $(\tilde{z}_1=z_1\wedge z_1\geq 0)\vee(\tilde{z}_1=0\wedge z_1<0)$. For the output node – in this case, z_3 – we introduce a pair of clauses representing the thresholding operation, i.e. $(y=1\wedge z_3\geq 0)\vee(y=0\wedge z_3<0)$. Taking the conjunction of the formulas for each node we obtain the characteristic formula in Figure 3c.

[2] While here we assume binary predictor models, i.e., classifiers, our approach generalizes to regression problems where $y\in\mathbb{R}$ and more generally any other output domain.



(a) Graphical representation

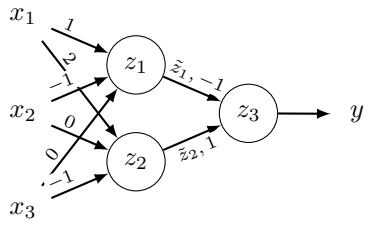
```
if x_1 == 1
  y = 0 if x_3 > 0 else 1
else
  y = 0 if x_2 == 1 else 1
return y
```

(b) Program (in Python)

$$\begin{aligned}\phi_f(\boldsymbol{x}, y) = &(x_1 = 1 \wedge x_3 > 0 \wedge y = 0)\\ &\vee (x_1 = 1 \wedge x_3 \leq 0 \wedge y = 1)\\ &\vee (x_1 = 0 \wedge x_2 = 1 \wedge y = 0)\\ &\vee (x_1 = 0 \wedge x_2 = 0 \wedge y = 1)\end{aligned}$$

(c) Characteristic formula

Figure 2: Decision tree: model, program and characteristic formula



(a) Graphical representation

```
z_1 = x_1 − x_2
z_2 = 2x_1 − x_3
z̃_1 = z_1 if z_1 >= 0 else 0
z̃_2 = z_2 if z_2 >= 0 else 0
z_3 = −z̃_1 + z̃_2
y = 1 if z_3 >= 0 else 0
return y
```

(b) Program (in Python)

$$\begin{aligned}\phi_f(\boldsymbol{x}, y) = &(z_1 = x_1 - x_2)\\ &\wedge (z_2 = 2x_1 - x_3)\\ &\wedge ((\tilde{z}_1 = z_1 \wedge z_1 \geq 0) \vee (\tilde{z}_1 = 0 \wedge z_1 < 0))\\ &\wedge ((\tilde{z}_2 = z_2 \wedge z_2 \geq 0) \vee (\tilde{z}_2 = 0 \wedge z_2 < 0))\\ &\wedge (z_3 = -\tilde{z}_1 + \tilde{z}_2)\\ &\wedge ((z_3 \geq 0 \wedge y = 1) \vee (z_3 < 0 \wedge y = 0))\end{aligned}$$

(c) Characteristic formula

Figure 3: Multilayer perceptron: model, program and characteristic formula

4 Finding the nearest counterfactual

Based on the counterfactual space $\mathrm{CF}_f(\hat{\boldsymbol{x}})$ defined in the previous section, we would like to produce counterfactual explanations for the output of a model f on a given input $\hat{\boldsymbol{x}}$ by trying to find a *nearest counterfactual*, which is defined as:

$$\hat{\boldsymbol{x}}^* \in \underset{\boldsymbol{x} \in \mathrm{CF}_f(\hat{\boldsymbol{x}})}{\mathit{argmin}}\ d(\boldsymbol{x}, \hat{\boldsymbol{x}}) \ . \tag{2}$$

For the time being, we assume that a notion of distance between instances, d, is given. For convenience, and without loss of generality, we also assume that d takes values in the interval $[0, 1]$.

4.1 Main algorithm

Our goal now is to leverage the representation of $\mathrm{CF}_f(\hat{\boldsymbol{x}})$ in terms of a logic formula to solve (2). To this end, we map the optimization problem in (2) into a sequence of satisfiability problems, which can be verified or refuted by standard SMT solvers. We do so by first converting the expression $d(\boldsymbol{x}, \hat{\boldsymbol{x}}) \leq \delta$, where $\delta \in [0, 1]$, into a logic formula $\phi_{d,\hat{\boldsymbol{x}}}(\boldsymbol{x}, \delta)$, which is valid if and only if $d(\boldsymbol{x}, \hat{\boldsymbol{x}}) \leq \delta$. We assume here that the distance d function is expressed by a program in the same language that we used to represent the models in Section 3. In particular, we can leverage the procedure detailed in Appendix A to automatically construct $\phi_{d,\hat{\boldsymbol{x}}}$. Then, both the counterfactual formula $\phi_{\mathrm{CF}_f(\hat{\boldsymbol{x}})}(\boldsymbol{x})$ and the distance formula $\phi_{d,\hat{\boldsymbol{x}}}(\boldsymbol{x}, \delta)$ are combined into the logic formula:

$$\phi_{\hat{\boldsymbol{x}},\delta}(\boldsymbol{x}) = \phi_{\mathrm{CF}_f(\hat{\boldsymbol{x}})}(\boldsymbol{x}) \wedge \phi_{d,\hat{\boldsymbol{x}}}(\boldsymbol{x}, \delta) \ ,$$

which is satisfiable if and only if there exists a counterfactual $\boldsymbol{x} \in \mathrm{CF}_f(\hat{\boldsymbol{x}})$ such that $d(\boldsymbol{x}, \hat{\boldsymbol{x}}) \leq \delta$. To check whether the above formula is satisfiable we use the satisfiability oracle $\mathsf{SAT}(\psi(\boldsymbol{x}))$ which returns either an instance $\boldsymbol{x}$ such that $\psi(\boldsymbol{x})$ is valid, or "unsatisfiable" if no such $\boldsymbol{x}$ exists.

Note that, while the oracle SAT allows us to verify if there exist counterfactual explanations at distance smaller or equal than a given threshold δ, solving optimization (2) requires finding a nearest counterfactual. To do so, we apply a binary search strategy on the distance threshold $\delta \in [0, 1]$ that allows us to find *approximately* nearest counterfactuals with a pre-specified degree of accuracy. This is implemented in Algorithm 1, which for an accuracy parameter $\epsilon > 0$ makes at most $O(\log(1/\epsilon))$ calls to SAT and returns a counterfactual $\hat{\boldsymbol{x}}_\epsilon \in \mathrm{CF}_f(\hat{\boldsymbol{x}})$ such that $d(\hat{\boldsymbol{x}}_\epsilon, \hat{\boldsymbol{x}}) \leq d(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{x}}) + \epsilon$, where $\hat{\boldsymbol{x}}^*$ is some solution of the optimization problem in (2). This mild dependence on the accuracy ϵ allows Algorithm 1 to trade-off finding arbitrarily accurate solutions of (2) with the number of calls made to the satisfiability oracle. Note that Algorithm 1 may also account for potential plausibility or diversity constraints (refer to next section for further details).

We remark here our approach to find nearest counterfactuals is agnostic to the details of the model and distance being used; the only requirement is that they must be expressable in a fairly general programming language. As a consequence, we can handle a wide variety of predictive models, including both differentiable – such as, logisitic regression and multilayer perceptron – and non-differentiable predictive models – e.g.,

Algorithm 1: Binary Search for Nearest Counterfactuals with Satisfiability Oracle

Input: Factual $\hat{\boldsymbol{x}}$, counterfactual formula $\phi_{\text{CF}_f(\hat{\boldsymbol{x}})}$, distance formula $\phi_{d,\hat{\boldsymbol{x}}}$, constraints formula $\phi_{g,\hat{\boldsymbol{x}}}$, accuracy ϵ

Output: Counterfactual $\hat{\boldsymbol{x}}_\epsilon$, distance $\delta_{\max} = d(\hat{\boldsymbol{x}}_\epsilon, \hat{\boldsymbol{x}})$, lower bound $\delta_{\min}$ on (2)

Let $\delta_{\min} \leftarrow 0$ and $\delta_{\max} \leftarrow 1$
while $\delta_{\max} - \delta_{\min} > \epsilon$ **do**
  Let $\delta \leftarrow \frac{\delta_{\min}+\delta_{\max}}{2}$
  Let $\phi_{\hat{\boldsymbol{x}},\delta}(\boldsymbol{x}) \leftarrow \phi_{\text{CF}_f(\hat{\boldsymbol{x}})}(\boldsymbol{x}) \wedge \phi_{d,\hat{\boldsymbol{x}}}(\boldsymbol{x}, \delta) \wedge \phi_{g,\hat{\boldsymbol{x}}}$
  Let $\boldsymbol{x} \leftarrow \mathsf{SAT}(\phi_{\hat{\boldsymbol{x}},\delta})$
  if $\boldsymbol{x}$ *is "unsatisfiable"* **then**
    Let $\delta_{\min} \leftarrow \delta$
  else
    Let $\hat{\boldsymbol{x}}_\epsilon \leftarrow \boldsymbol{x}$ and $\delta_{\max} \leftarrow \delta$
return $\hat{\boldsymbol{x}}_\epsilon$, $\delta_{\min}$, $\delta_{\max}$

decision trees and random forest– as well as a wide variety of distance functions (refer to next section for further details). Moreover, the bound $\delta_{\min}$ returned by Algorithm 1 provides a certificate that any solution $\hat{\boldsymbol{x}}^*$ to (2) must satisfy $d(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{x}}) > \delta_{\min}$. This is because whenever $\mathsf{SAT}(\psi(\boldsymbol{x}))$ returns "unsatisfiable" it does so by internally constructing a proof that the formula $\psi(\boldsymbol{x})$ is not valid.

4.2 Distance, Plausibility, and Diversity

Next we discuss additional criteria in the form of logic clauses that guide the satisfiability problem towards generating a counterfactual explanation with desired properties.

Distance. We first discuss several forms for the distance function $d(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_\epsilon)$ that can be used to define the notion of nearest counterfactual. To this end, we first remark that in consequential decision making the input feature space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_J$ is often heterogeneous – for example, gender is categorical, education level is ordinal, and income is a numerical variable. We define an appropriate distance metric for every kind of variable in the input feature space of the model as:

$$\delta_j(x_j, \hat{x}_j) = \begin{cases} |x_j - \hat{x}_j|/R_j & \text{if } x_j \text{ is numerical} \\ \mathbb{I}[x_j \neq \hat{x}_j] & \text{if } x_j \text{ is categorical} \\ |x_j - \hat{x}_j|/R_j & \text{if } x_j \text{ is ordinal} \end{cases},$$

where R_j corresponds to the range of the feature x_j and is used to normalize the distances for all input features, such that $\delta_j : \mathcal{X}_j \times \mathcal{X}_j \to [0, 1]$ for all j, independently on the feature type. By defining the distance vector $\boldsymbol{\delta} = (\delta_1, \cdots, \delta_J)$ (being J the total number of input features), one can now write the distance between instances as:

$$d(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_\epsilon) = \alpha||\boldsymbol{\delta}||_0 + \beta||\boldsymbol{\delta}||_1 + \gamma||\boldsymbol{\delta}||_\infty \ , \qquad (3)$$

where $||\cdot||_p$ is the p-norm of a vector, and $\alpha, \beta, \gamma \geq 0$ such that[3] $(\alpha+\beta)/J+\gamma = 1$. Intuitively, 0-norm is used to restrict the number of features that changes between the initial instance $\hat{\boldsymbol{x}}$ and the generated counterfactual $\hat{\boldsymbol{x}}_\epsilon$; the 1-norm is used to restrict the average change distance between $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{x}}_\epsilon$; and ∞-norm is used to restrict maximum change across features. Any distance of this type can easily be expressed as a program.

Plausibility. Up to this point, we have only considered minimum distance as the only requirement for generating a counterfactual. However, this might result in unrealistic counterfactuals, such as e.g., decrease the age or change the gender of a loan applicant. To avoid unrealistic counterfactuals, one may introduce additional *plausibility constraints* in the optimization problem in Eq. (2). This is equivalent to adding a conjunction in the constraint formula $\phi_{g,\hat{\boldsymbol{x}}}$ in Algorithm 1 that accounts for any additional plausibility formulae ϕ_p, which ensure that: i) each feature in the counterfactual should be data-type and data-range consistent with the training data; and ii) only actionable features [Ustun et al., 2019] are changed in the resulting counterfactual.

First, since here we are working with heterogeneous feature spaces, we require all the features in the counterfactual to be consistent in both the data-types (categorical, ordinal, etc.) and the data-ranges with the training data. In particular, if a categorical (ordinal) feature is one-hot (thermometer) encoded to be used as input to the predictive model, e.g., a logistic regression classifier, we make sure that the generated counterfactual provides a valid one-hot vector (thermometer) for such feature. Likewise, for any numerical feature we ensure that its value in the counterfactual falls into observed range in the original data used to train the predictive model.

Moreover, to account for a non-actionable/immutable feature x_j, i.e., a feature whose value in the counterfactual explanation should match its initial value, we set ϕ_p to be $(x_j = \hat{x}_j)$. Similarly, we account for variables that only allow for increasing values by setting $\phi_p = (x_j \geq \hat{x}_j)$.

Diversity. Finally, one might be interested in generating a (small) set of diverse counterfactual explanations for the same instance $\hat{\boldsymbol{x}}$. To this end, we iteratively call Algorithm 1 with a constraints formula ϕ_v that includes

[3] Constraints on the distance hyperparameters ensure that the overall distance $d(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_\epsilon) \in [0, 1]$. To this end, since $\max||\cdot||_0 = \max||\cdot||_1 = J, \max||\cdot||_\infty = 1$, the hyperparameters must satisfy $(\alpha + \beta)/J + \gamma = 1$.

Table 1: Comparison of approaches for generating counterfactual explanations, based on the supported model types, data types, distance types, plausibility constraints (actionability, data type/range consistency), and optimal distance guarantees.

Approach	Models	Data types	Distances	Plausibility	Optimal Distance
Proposed (MACE)	tree, forest, lr, mlp	heterogeneous	$\ell_p \; \forall \; p$	✓	✓
Minimum Observable (MO)	-	heterogeneous	$\ell_p \; \forall \; p$	✓	x
Feature Tweaking (FT)	tree, forest	heterogeneous	$\ell_p \; \forall \; p$	x	x
Actionable Recourse (AR)	lr	numeric, binary	ℓ_1, ℓ_∞	x[6]	x

diversity clauses to ensure that the newly generated explanation is substantially different from all the previous ones. We can encode diversity by forcing that the distance between every pair of counterfactual explanations is greater than a given value. For example, we can take[4] $\phi_v = \bigwedge_i \left(\bigvee_{j \in J} (x_j \neq \hat{x}^i_{\epsilon,j}) \right)$ to restrict repetitive counterfactuals by enforcing subsequent counterfactuals to have 0-norm distance at least 1 from all previous counterfactuals.

5 Experiments

In this section, we empirically demonstrate the main properties of MACE compared to existing approaches.

Datasets. We evaluate MACE at generating counterfactual explanations on three real-world datasets in the context of loan approval (Adult [Adult data, 1996] and Credit [Yeh and Lien, 2009] datasets) and pretrial bail (COMPAS dataset [Larson et al., 2016]). All the three datasets present heterogeneous input spaces.

Baselines. We compare the performance of MACE at generating the nearest counterfactual explanations with: the *Minimum Observable* (MO) approach [Wexler et al., 2019], which searches in the dataset for the closest sample that flips the prediction; the *Feature Tweaking* (FT) approach [Tolomei et al., 2017], which searches for the nearest counterfactual lying close to the decision boundary of a Random Forest; and the *Actionable Recourse* (AR) [Ustun et al., 2019], which solves a mixed integer linear program to obtain counterfactual explanations for Linear Regression models. Table 1 summarizes the main properties of all the considered approaches to generate counterfactuals.

Metrics. To assess and compare the performance of the different approaches, we recall the criteria of good explanations for consequential decisions: i) the returned counterfactual should be as near as possible to the factual sample corresponding to the individual's features; ii) the returned counterfactual must be plausible (refer to Section 4.2). Hence, we quantitatively compare the performance of MACE with the above approaches in terms of i) the *normalized distance* $\boldsymbol{\delta}$; and ii) *coverage* Ω indicating the percentage of factual samples for which the approach generates plausible (in type and range) counterfactuals.

Experimental set-up. We consider as predictive models decision trees, random forest, logistic regression, and multilayer perceptron, which we train on the three datasets using the Python library scikit-learn [Pedregosa et al., 2011], with default parameters.[5] Furthermore, to demonstrate the off-the-shelf flexibility in the various setups described, we build MACE atop the open-source PySMT library [Gario and Micheli, 2015] with the Z3 [de Moura and Bjørner, 2008] backend. In Appendix C.2, we provide a thorough empirical evaluation of the *computational cost* of the off-the-shelf PySMT solver – including run-time comparisons between MACE and other baselines, – as well as a discussion on the choice of ϵ trading-off arbitrarily accurate solutions of (2) with the number of calls made to the satisfiability oracle.

For each combination of approach, model, dataset, and distance, we generate the nearest counterfactual explanations for a held-out set of 500 instances classified as negative by the corresponding model. Here we consider the ℓ_0, ℓ_1, ℓ_∞ norms as a measure of distance to identify the nearest counterfactuals. Unfortunately, we found that FT not once returned a plausible counterfactual. As a consequence, we modified the original implementation of FT, to ensure that the generated counterfactuals are plausible. The resulting *Plausible Feature Tweaking* (PFT) projects the set of candidate counterfactuals into a plausible domain before selecting the nearest counterfactual amongst them. This was not possible for AR because the approach only returns a single counterfactual, with no avail if it is not plausible.[6]

Coverage and distance results. Table 2 shows the

[4] $\hat{\boldsymbol{x}}^i_{\epsilon,j}$ is the j-th dimensions of the i-th counterfactual.

[5] For the multilayer perceptron, we used two hidden layers with 10 neurons each to avoid overfitting. See Appendix B.1 for model selection details.

[6] Importantly, Actionable Recourse does support actionability and data-range plausibility, however, it lacks support for data-type plausibility – Appendix B.3 describes the failure points of AR, as reported by the authors.

Table 2: Coverage Ω computed on $N = 500$ factual samples. For comparison, $\Omega_{\text{MACE}} = \Omega_{\text{MO}} = 100\%$ always, by definition and by design, respectively. Cells are shaded when tests are not supported. Higher % is better.

		Adult			Credit			COMPAS		
		ℓ_0	ℓ_1	ℓ_∞	ℓ_0	ℓ_1	ℓ_∞	ℓ_0	ℓ_1	ℓ_∞
tree	PFT	0%	0%	0%	68%	68%	68%	74%	74%	74%
forest	PFT	0%	0%	0%	99%	99%	99%	100%	100%	100%
lr	AR		18%	0.4%		100%	100%		100%	100%

Table 3: Percentage of improvement in distances, computed as $100 * \mathbb{E}[1 - \boldsymbol{\delta}_{\text{MACE}}/\boldsymbol{\delta}_{\text{Other}}]$. $N = \Omega_{\text{MACE}} \cap \Omega_{\text{Other}}$ factual samples. Cells are shaded when tests are not supported. The higher the %, the better the improvement.

		Adult			Credit			COMPAS		
		ℓ_0	ℓ_1	ℓ_∞	ℓ_0	ℓ_1	ℓ_∞	ℓ_0	ℓ_1	ℓ_∞
tree	MACE ($\epsilon = 10^{-3}$) vs MO	47%	80%	70%	67%	66%	47%	1%	5%	5%
	MACE ($\epsilon = 10^{-5}$) vs MO	47%	81%	72%	67%	96%	94%	1%	5%	5%
	MACE ($\epsilon = 10^{-3}$) vs PFT				53%	87%	85%	14%	56%	54%
	MACE ($\epsilon = 10^{-5}$) vs PFT				53%	97%	96%	15%	55%	54%
forest	MACE ($\epsilon = 10^{-3}$) vs MO	51%	81%	69%	68%	61%	38%	1%	6%	6%
	MACE ($\epsilon = 10^{-5}$) vs MO	51%	82%	71%	68%	97%	96%	1%	6%	6%
	MACE ($\epsilon = 10^{-3}$) vs PFT				53%	84%	81%	4%	28%	27%
	MACE ($\epsilon = 10^{-5}$) vs PFT				53%	96%	96%	4%	28%	27%
lr	MACE ($\epsilon = 10^{-3}$) vs MO	62%	92%	86%	80%	82%	80%	3%	8%	6%
	MACE ($\epsilon = 10^{-5}$) vs MO	62%	93%	88%	80%	82%	81%	3%	6%	6%
	MACE ($\epsilon = 10^{-3}$) vs AR		3%	89%		39%	67%		10%	38%
	MACE ($\epsilon = 10^{-5}$) vs AR		5%	91%		42%	71%		10%	38%
mlp	MACE ($\epsilon = 10^{-3}$) vs MO	60%	92%	91%	77%	85%	91%	1%	3%	3%
	MACE ($\epsilon = 10^{-5}$) vs MO	60%	93%	93%	77%	96%	96%	1%	3%	3%

coverage Ω of all the approaches based only on data-range and data-type plausibility. Note that, since by definition both MACE and MO have 100% coverage, we have not depicted these values in the table. In contrast, PFT fails to return counterfactuals for roughly 15% of the Credit and COMPAS datasets, while both PFT and AR achieve minimal coverage on the Adult dataset.[7] Focusing on those factual samples for which PFT and AR return plausible counterfactuals, we are able to compute the relative distance reductions achieved when using MACE as compared to other approaches, as shown in Table 3 (additionally, Figure 4 in Appendix B shows the distribution of the distance of the generated plausible counterfactual for all models, datasets, distances, and approaches). Here, we observe that MACE results in significantly closer counterfactual explanations than competing approaches, with an average decrease in distance of 70.2% for Adult, 75.4% for Credit, and 21.1% for COMPAS. As a consequence, the counterfactuals generated by MACE would require significantly less effort on behalf of the affected individual in order to achieve the desired prediction.

Plausibility contraints. While performing a qualitative analysis of generated counterfactuals we observed that many of them require changes in features that are often protected by law such as, age, race, and gender [Barocas and D. Selbst, 2016]. As an example, for a trained random forest, the counterfactuals generated by both the MACE and MO approaches required individuals to change their age. Worse yet, for a substantial portion of the counterfactuals, a reduction in age was required, which is not even possible. To further study this effect, we regenerate counterfactual explanations for those samples for which age-change was required, with an additional plausibility constraint ensuring that the age shall not change (results with constraints to ensure non-decreasing age are shown in Appendix C.3). The results presented in Table 4 show interesting results. First, we observe that the additional plausibility constraint for the age incurs significant increases in the distance of the nearest counterfactual – being, as expected, more pronounced for the ℓ_1 and the ℓ_∞ norms, since the ℓ_0 norm only accounts for the number of features that change in the counterfactual but not for how much they change. For the ℓ_0 norm, as expected, we find that for the 66 factual samples (i.e., $13.2\% \times 500$) for which the unrestricted MACE required age-change, the addition of the no-age-change constraint results in counterfactuals at very similar distance. In fact, of the newly generated counterfactuals, 8/66 only require a change in Occupation, and 19/66 only require a change in Capital Gains, therefore remaining at the same distance as the original counterfactual. In contrast, for the ℓ_1 and the ℓ_∞ norms we find that the

[7] The Adult dataset comprises a realistic mix of integer, real-valued, categorical, and ordinal variables common to consequential scenarios; further details in Appendix B.2.

Table 4: Percentage of factual samples for which the nearest counterfactual sample requires a change in age for a random forest trained on the Adult dataset, and the corresponding increase in distance to nearest counterfactual when restricting the approaches not to change age: $100 \times \mathbb{E}[\delta_{\text{restr.}}/\delta_{\text{unrestr.}} - 1]$. Lower % is better.

	ℓ_0		ℓ_1		ℓ_∞	
	% age-change	rel. dist. increase	% age-change	rel. dist. increase	% age-change	rel. dist. increase
MACE ($\epsilon = 10^{-5}$)	13.2%	9.0%	20.4%	100.3%	84.4%	32.8%
MO	78.8%	50.9%	92.0%	245.7%	95.6%	193.3%

Table 5: A diverse set of generated counterfactuals is presented for an individual from the Credit dataset.

	Latest Bill	Latest Payment	University Degree	Will default next month?
Factual	\$370	\$40	some	yes
CF #1	\$368	\$1448	some	no
CF #2	\$0	\$1241	some	no
CF #3	\$0	\$390	graduate	no

restricted counterfactual incurs a significant increase in the distance (cost) with respect to the unrestricted counterfactual. These results suggest that the predictions of the random forest trained on the Adult data are strongly correlated to the age, which is often legally and socially considered as unfair. This suggests that counterfactuals found with MACE may assist in qualitatively ascertaining if other desiderata, such as fairness, are met [Doshi-Velez and Kim, 2017, Weller, 2017].

Diversity constraints. Finally, we present a situation where MACE can be used to generate counterfactuals under both plausibility and diversity constraints. Consider a loan borrower from the Credit dataset identified with the following features[8]: John is a married male between 40-59 years of age with "some" university degree. Financially, over the last 6 months, John has been struggling to make payments on his bank loan. Given his circumstances, a logistic regression model trained on the historical dataset has predicted that John will default on his loan next month. To prevent this default, the bank uses MACE (ℓ_1 distance, $\epsilon = 10^{-3}$) to generate the diverse suggestions in Table 5, via successive runs of Algorithm 1. Each new run augments the constraints formula (already including plausibility constraints on his age, sex, and marital status) with an additional clause enforcing ℓ_0 diversity as discussed in Section 4.2. The returned counterfactuals (of which only 3 are shown), present John with diverse courses of action: either reduce spending and make a lump-sum payment on the debt (CF #2) or continue spending the same as before, but make an even larger payment to account for continued expenditures (CF #1). Alternatively, providing documents confirming a graduate degree would put John in a low-risk (no default) bracket (CF #3). We invite the reader to imagine parallels to the above situation for Adult and COMPAS datasets.

[8]Complete feature list in Appendix C.4

6 Conclusions

In this work, we have presented a novel approach for generating counterfactual explanations in the context of consequential decisions. Building on theory and tools from formal verification, we demonstrated that a large class of predictive models can be compiled to formulae which can be verified by standard SMT-solvers. By conjuncting the model formula with formulae corresponding to distance, plausibility, and diversity constraints, we demonstrated on three real-world datasets and four popular predictive models that the proposed method not only achieves perfect coverage, but also generates counterfactuals at more favorable distances than existing optimization-based approaches. Furthermore, we showed that the proposed method can not only provide explanations for individuals subject to automated decision making systems, but also inform system administrators regarding the potentially unfair reliance of the model on protected attributes.

There are a number of interesting directions for future work. First, MACE can naturally be extended to support counterfactual explanations for multi-class classification models, as well as regression scenarios. Second, extending the multi-faceted notion of plausibility defined in Section 4.2 (actionability, data type-/range consistency, which focus on individual features), it would be interesting to account for statistical correlations and unmeasured confounding factors among the features when generating counterfactual explanations (i.e., *realizability*). Third, we would like also to explore how different notions of diversity may help generating meaningful and useful counterfactuals. Finally, in our experiments we noticed that the running time of MACE directly depends on the efficiency of the SMT solver. As future work we aim to make the proposed method more scalable on large models by investigating recent ideas that have been developed in the context of formal verification of deep neural networks [Huang et al., 2017, Katz et al., 2017, Singh et al., 2019] and optimization modulo theories [Nieuwenhuis and Oliveras, 2006, Sebastiani and Tomasi, 2012].

References

[Adult data, 1996] Adult data (1996). https://archive.ics.uci.edu/ml/datasets/adult.

[Barocas and D. Selbst, 2016] Barocas, S. and D. Selbst, A. (2016). Big data's disparate impact. *SSRN Electronic Journal*.

[Barrett et al., 2011] Barrett, C., Conway, C. L., Deters, M., Hadarean, L., Jovanovic, D., King, T., Reynolds, A., and Tinelli, C. (2011). CVC4. In Gopalakrishnan, G. and Qadeer, S., editors, *Proceedings of the 23rd International Conference on Computer Aided Verification (CAV '11)*, volume 6806, pages 171–177. Springer.

[Cytron et al., 1991] Cytron, R., Ferrante, J., Rosen, B. K., Wegman, M. N., and Zadeck, F. K. (1991). Efficiently computing static single assignment form and the control dependence graph. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 13(4):451–490.

[de Moura and Bjørner, 2008] de Moura, L. M. and Bjørner, N. (2008). Z3: an efficient SMT solver. In Ramakrishnan, C. R. and Rehof, J., editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008*, volume 4963, pages 337–340. Springer.

[Dijkstra, 1968] Dijkstra, E. W. (1968). A constructive approach to the problem of program correctness. *BIT Numerical Mathematics*, 8(3):174–186.

[Doshi-Velez and Kim, 2017] Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

[Flanagan and Saxe, 2001] Flanagan, C. and Saxe, J. B. (2001). Avoiding exponential explosion: Generating compact verification conditions. In *ACM SIGPLAN Notices*, volume 36, pages 193–205. ACM.

[Floyd, 1993] Floyd, R. W. (1993). *Assigning Meanings to Programs*, pages 65–81. Springer Netherlands, Dordrecht.

[Freitas, 2014] Freitas, A. A. (2014). Comprehensible classification models: a position paper. *ACM SIGKDD explorations newsletter*, 15(1):1–10.

[Gario and Micheli, 2015] Gario, M. and Micheli, A. (2015). Pysmt: a solver-agnostic library for fast prototyping of smt-based algorithms. In *SMT Workshop 2015*.

[Gunning, 2019] Gunning, D. (2019). Darpa's explainable artificial intelligence (XAI) program. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, pages ii–ii. ACM.

[Hoare, 1969] Hoare, C. A. R. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580.

[Huang et al., 2017] Huang, X., Kwiatkowska, M., Wang, S., and Wu, M. (2017). Safety verification of deep neural networks. In Majumdar, R. and Kuncak, V., editors, *Computer Aided Verification - 29th International Conference, CAV*, volume 10426, pages 3–29. Springer.

[Katz et al., 2017] Katz, G., Barrett, C. W., Dill, D. L., Julian, K., and Kochenderfer, M. J. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. In Majumdar, R. and Kuncak, V., editors, *Computer Aided Verification - 29th International Conference, CAV*, volume 10426, pages 97–117. Springer.

[Kodratoff, 1994] Kodratoff, Y. (1994). The comprehensibility manifesto. *KDD Nugget Newsletter*, 94(9).

[Kroening and Strichman, 2008] Kroening, D. and Strichman, O. (2008). *Decision Procedures: An Algorithmic Point of View*. Springer Publishing Company, Incorporated, 1 edition.

[Larson et al., 2016] Larson, J., Mattu, S., Kirchner, L., and Angwin, J. (2016). https://github.com/propublica/compas-analysis.

[Lipton, 2018] Lipton, Z. C. (2018). The mythos of model interpretability. *Queue*, 16(3):30:31–30:57.

[Murdoch et al., 2019] Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080.

[Nieuwenhuis and Oliveras, 2006] Nieuwenhuis, R. and Oliveras, A. (2006). On SAT modulo theories and optimization problems. In Biere, A. and Gomes, C. P., editors, *Theory and Applications of Satisfiability Testing - SAT*, volume 4121, pages 156–169. Springer.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

[Rosen et al., 1988] Rosen, B. K., Wegman, M. N., and Zadeck, F. K. (1988). Global value numbers and redundant computations. In *Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 12–27. ACM.

[Rudin, 2018] Rudin, C. (2018). Please stop explaining black box models for high stakes decisions. *arXiv preprint arXiv:1811.10154*.

[Rüping, 2006] Rüping, S. (2006). *Learning interpretable models*. PhD dissertation, Technical University of Dortmund.

[Russell, 2019] Russell, C. (2019). Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pages 20–28. ACM.

[Sebastiani and Tomasi, 2012] Sebastiani, R. and Tomasi, S. (2012). Optimization in SMT with $\mathcal{LA}(\mathbb{Q})$ cost functions. In Gramlich, B., Miller, D., and Sattler, U., editors, *Automated Reasoning - 6th International Joint Conference, IJCAR*, volume 7364, pages 484–498. Springer.

[Singh et al., 2019] Singh, G., Gehr, T., Püschel, M., and Vechev, M. T. (2019). An abstract domain for certifying neural networks. *PACMPL*, 3(POPL):41:1–41:30.

[Tolomei et al., 2017] Tolomei, G., Silvestri, F., Haines, A., and Lalmas, M. (2017). Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 465–474. ACM.

[Ustun et al., 2019] Ustun, B., Spangher, A., and Liu, Y. (2019). Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 10–19. ACM.

[Voigt and Von dem Bussche,] Voigt, P. and Von dem Bussche, A. The EU general data protection regulation (GDPR).

[Wachter et al., 2017a] Wachter, S., Mittelstadt, B., and Floridi, L. (2017a). Why a right to explanation of automated decision-making does not exist in the general data protection regulation. *International Data Privacy Law*, 7(2):76–99.

[Wachter et al., 2017b] Wachter, S., Mittelstadt, B., and Russell, C. (2017b). Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31(2).

[Weller, 2017] Weller, A. (2017). Challenges for transparency. In *Workshop on Human Interpretability in Machine Learning (ICML)*.

[Wexler et al., 2019] Wexler, J., Pushkarna, M., Bolukbasi, T., Wattenberg, M., Viégas, F., and Wilson, J. (2019). The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics*, 26(1):56–65.

[Yeh and Lien, 2009] Yeh, I.-C. and Lien, C.-h. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480.