TRUSTAI

TRANSPARENT, RELIABLE
& UNBIASED SMART TOOL

D4.3: Results of the interaction with WP3: generation of explainable expressions by iterated dialog with the user.

Lead participant: University of Tartu

March 28, 2024

DOCUMENT CONTROL PAGE

DOCUMENT	**D4.3 – Results of the interaction with WP3: generation of explainable expressions by iterated dialog with the user.**
TYPE	**Report**
DISTRIBUTION LEVEL	Public
DUE DELIVERY DATE	31/03/2024
DATE OF DELIVERY	31/03/2024
VERSION	0.2
DELIVERABLE RESPONSIBLE	UT
AUTHOR (S)	Marharyta Domnich (UT), Raul Vicente Zafra (UT), Eduard Barbu (UT), Rodion Krjutškov (UT)
OFFICIAL REVIEWER/s	Gonçalo Reis Figueira (INESC TEC)

DOCUMENT HISTORY

VERSION	AUTHORS	DATE	CONTENT AND CHANGES
0.1	University of Tartu	22/03/2024	Integration of Energy and Retail case explanations.
0.2	University Of Tartu	28/03/2024	Integration of experts feedback and interactive chat information.
0.3	University Of Tartu	29/03/2024	Integration of the work based on iterated dialogue interface
0.5	University of Tartu	31/03/2024	Reviewer (INESC TEC) corrections and suggestions integrated.

ACKNOWLEDGEMENTS

NAME	PARTNER
Nikos Sakkas	AIT
André Morim	LTPlabs
David Viana	LTPlabs
Gonçalo Reis Figueira	INESC TEC
Fábio Neves Moreira	INESC TEC
Peter Bosman	CWI
Tanja Alderliesten	CWI
Evi Sijben	CWI

Executive Summary

The present deliverable *D4.3, "Results of the interaction with WP3: generation of explainable expressions by iterated dialog with the user"* reports on how the explanations given in the form of counterfactuals assist in the process of model interpretation and selection by the users. In particular, the work in WP3 has produced a novel framework for local and contrastive explanations in the form of counterfactual explanations. The counterfactual search developed in WP3 (and integrated in the TRUST-AI platform) includes two novel terms that enhance the coherence and feasibility of the counterfactual explanations. Constraints specific to each use case are also incorporated in the production of counterfactual explanations. The framework is applied to the online retail and energy use cases where the learning loop is completed by the users feedback after analyzing the counterfactuals explanations. For the energy case counterfactuals found by the framework are satisfactory and for the online retail case the analysis identified the need for enhanced model training.

Language Models (LLMs) enhance human-ML model communication, enabling direct user interactions for inquiries on data, model explanations, and hypotheticals. We also present in this deliverable the work towards a specialized LLM-based conversational interface for the energy case, improving human interactions with GP models by providing customized explanations.

Table of Contents

1 Introduction **9**
2. Counterfactual search with CoDiCE 11
2.1 Directional coherence 12
2.2 Using diffusion distance to search for more feasible transitions 14
2.3. User specified constraints for enhanced actionability 15
2.4 Transformed space counterfactual analysis 15
3. Counterfactual analysis for GP model validation and improvement **16**
3.1 Online retail analysis 16
3.1.1 Interaction with explanation module on original space features 16
3.1.2 Interaction with explanation module on transformed space features 20
3.1.3 Learning loop summary 21
3.2 Energy case 23
3.2.1 Counterfactual analysis for energy use case 24
3.2.2 Learning loop summary 26
4. Iterated dialog interface based on LLMs **27**
4.1 Introduction 27
4.2 TalkToModel Framework 27
4.3 Integration of Energy Use case Data 28
4.4. Future work 31
Conclusion **33**
References **34**

Abbreviations and Acronyms

AI	Artificial Intelligence
D2.1	Deliverable 2.1
D3.1	Deliverable 3.1
EC	European Commission
EU	European Union
HCXAI	Human-centred Explainable AI
UC	Use-case
WP	Work Package
XAI	Explainable Artificial Intelligence

Task 4.3. GP-GOMEA and SMGP as well as some baseline GPs will be confronted with WP3: providing sample models, and transforming them in explanations, and checking (with WP2) the relevance of these explanations, suggesting new variables to be used for the Symbolic Regression step. In this task, this multiple loop will only be instantiated with the toy problems derived from the use cases, their application to the real-size problem being done in each of the use-case WPs.

Deliverable 4.3. Results of the interaction with WP3: generation of explainable expressions by iterated dialog with the user.

1 Introduction

WP3 has developed a framework to produce local and contrastive explanations in the form of counterfactuals (type of explanation highlighted in D2.1 and identified by users as their preferred type of explanation). The main novelty of the framework in WP3 is the incorporation of relevant human behavioral heuristics in producing and selecting explanations, including the formalization of the feasibility and coherence characteristics of a local explanation. While local explanations aim to elucidate the model's behavior for individual instances, contrastive explanations focus on why the model made a particular decision instead of an alternative. Indeed, counterfactual explanations describe how the current input or situation would have to change to obtain the desired outcome (as predicted by the model). However, even for an individual instance there are multiple counterfactual explanations that can be selected, and the effectiveness of such explanations might be highly context-dependent and vary significantly across different users and use cases. Therefore, an interactive approach in which the user interacts with the explanations and poses further questions can tailor the explanations to better fit the users needs. In addition, the interaction of the user with the explanations for different models can be used to select more relevant models. Overall, the aim is that the interaction with the explanation framework will enhance the transparency and trustworthiness of the machine learning models. The overall flow of the interaction is illustrated in Figure 1.

In this deliverable D4.3, we report the results of the application of the counterfactual framework developed in WP3, named CoDiCE, to several use cases. In particular, we report on the interaction of the user with the explanations (counterfactuals provided for individual instances) in the form of the feedback received and further questions asked by the users. We argue that by completing such a learning loop the users can identify more meaningful explanations in specific contexts. This interaction helps in refining the explanations, making them more relevant, understandable, and actionable. Additionally, we also explored the idea of searching for relevant counterfactuals using the features combinations used by the Genetic Program models. **Unfortunately, the clinical case for healthcare had to be postponed due to the lack of sufficient data at the moment to conduct the explainability analysis and learning loop.**

Language Models (LLMs) streamline communication between humans and machine learning (ML) models, allowing individuals to interact directly with an ML model. This interaction lets users ask about data statistics, model explainability, hypothetical situations, and other related topics. In this context, we introduce our specialized adaptation of an LLM-based conversational interface for the energy use case. The interface enhances the interaction between humans and the trained GP models offering tailored explanations to the human users.

This deliverable is structured in the following way. First, we describe the methodology and tools used. In particular, in Section 2 we describe the framework for the counterfactual search (used to provide the counterfactual explanations), as well as the formalization of the two novel heuristics for the feasibility and directed coherence of explanations. Moreover, we also describe the transformed feature space used to search for counterfactuals in the space of feature

combinations extracted from the GP models. In Section 3, we report on the results of the application of the counterfactual explanations framework to the use cases of online retail and energy demand forecast. The feedback and insights from the experts upon the given explanations are also listed there. In Section 4, we detail how an interactive framework, built on LLM technology, is adapted to incorporate data and GP models specifically for the energy use case. We conclude with some collective insights about the use of counterfactuals and iterated dialog to validate and improve the models.

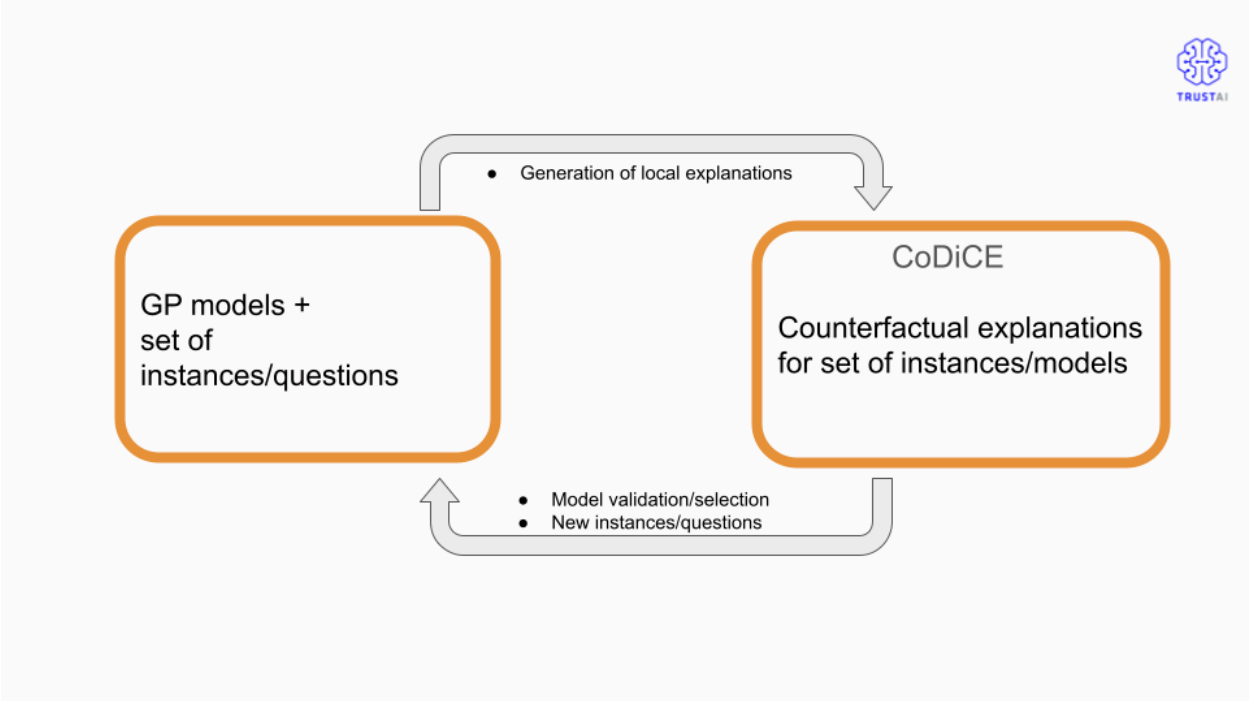

Figure 1. Overall diagram of the producing and refining counterfactual explanations by iterated feedback with users.

2. Counterfactual search with CoDiCE

As was identified in D2.1 counterfactual explanations were found effective for explaining and evaluating models for every use case. Counterfactual explanation can be thought of as the possible “smallest” change in input settings in order to get the desired model output that changes the model prediction. An illustrative example of the use of counterfactual explanations is that of a person applying for a loan and being rejected due to the model prediction. The company is willing to provide an explanation about its decision. However, giving a simple list of feature importance overview does not provide a local explanation nor guarantees that changing the most important feature would flip the prediction to a desired outcome. Instead, a counterfactual explanation highlighting the feasibility and actionability of the proposed alternative is usually found more effective. A novel interactive counterfactual framework was developed with use-cases requirements in mind. The flexibility of the counterfactual searching goals is achieved with the possibility to adjust parameters of different terms in objective function and the possibility to incorporate a set of constraints.

D3.1 reported the preliminary idea of counterfactual formalization. Here, we report how we enhanced it by accounting for human preferences for feasibility, actionability and coherence. In particular, these preferences are formalized with the help of the concepts of diffusion distance and directional coherence. The update objective counterfactual function is described next.
We denote f as a trained predictor function that maps input space to output space. Given a factual point or the original input point $x = (x_1, x_2, \ldots, x_n) \in \mathcal{X} \subseteq \mathbb{R}^n$, our objective is to identify a counterfactual point $c^* = (c_1^*, c_2^*, \ldots, c_n^*) \in \mathcal{X} \subseteq \mathbb{R}^n$ that yields the desired label y while minimizing a weighted sum of diffusion distance, sparsity and directional coherence penalties. The optimization problem is defined as follows:

$$c = \arg\min_{x^*} \Big(\text{loss}(f(x^*), y) + \lambda_1 \text{diffusion\_dist}(x^*, x) + \lambda_2 \text{sparsity}(x^*, x) + \lambda_3 (1 - \text{dcoherence}(x^*, x)) \Big), \quad (1)$$

where:

- $\text{loss}(f(x^*), y)$ is the loss term that checks if the counterfactual outcome is equal to the desired outcome, we utilize hinge loss for classification and mean squared error for regression.
- $\text{diffusion\_dist}(x^*, x)$ quantifies the diffusion distance between the original point x and counterfactual point x^*.
- $\text{sparsity}(x^*, x)$ computes l_0 distance to count the number of features that have been modified.

- $\mathrm{dcoherence}(x^*, x)$ assesses the directional coherence by aligning the joint direction of the counterfactual change with a set of specified or preferred marginals directions of change. Since we are interested in minimizing objective function, we take the penalty measure to be (1 − dcoherence).

The terms are weighted by hyperparameters λ_1, λ_2 and λ_3, which can be adjusted or set to 0 if a particular constraint or preference is not applicable.

2.1 Directional coherence

In D3.1 we identified several user-specified and inherited user constraints for each use case. In our approach we incorporate inherited user constraints by the inclusion of the directional coherence term.

Directional Coherence formulates a bias designed to maintain consistency between the marginal (one feature at a time) and joint (multiple features simultaneously) directions in feature space needed to flip the outcome of the model's prediction. This coherence facilitates the generation of counterfactual explanations that not only adhere to the model's predictions for individual feature alterations but also align with the overall direction of change necessary to shift to a desired counterfactual state. Such a term can be used to tune the importance of aligning counterfactual paths with intuitive human reasoning about a set of causal expectations when changes are produced in marginal directions (changing one feature at a time).

To illustrate this concept, consider again the scenario of applying for a home loan, where it is intuitively expected that an increase in income for either the applicant or co-applicant would improve the chances of loan approval. We would be shocked to learn that a bank advises increasing the applicant's income, but decreases a co-applicant income. This counterintuitive recommendation could arise from the specific nature of the data distribution, reflecting scenarios where other people with these factual scenarios in the past got loan approval. We argue that although observing such a point is possible, it would represent an undesirable direction for counterfactual explanation (although we have encountered such types of unintuitive explanations when using popular counterfactual search frameworks such as DICE). Figure 2.1 illustrates this conceptual situation. An input point highlighted with rectangular shape and belonging to Class 1, has two counterfactual candidates CF1 and CF2 , which are associated with the desired Class 2. The data spread indicates that increases in Feature 1 and Feature 2 are correlated with a higher likelihood of predicting Class 2. Consequently, the CF2 point is directionally coherent, as the joint increase in these features aligns with the marginal direction of probability of Class 2. On the other hand, CF1 is directionally incoherent, since the change in Feature 1 leads to decrease in the posterior probability of predicting Class 2.

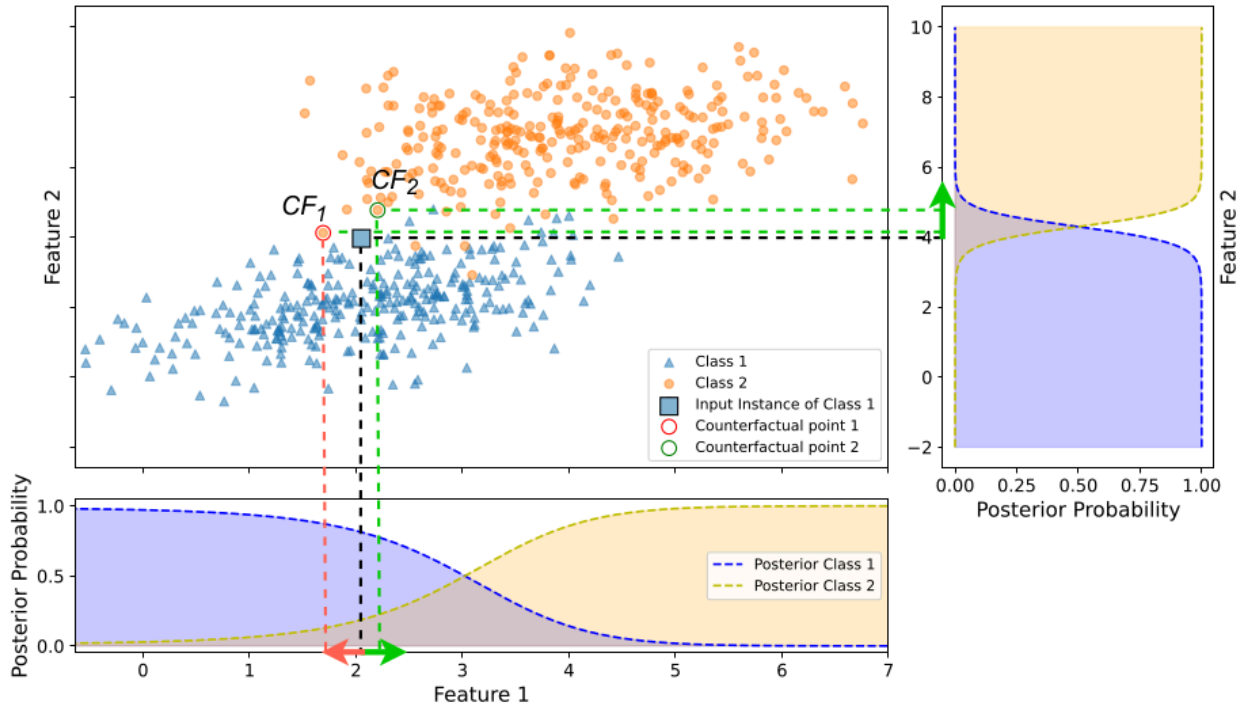

Figure 2.1. Illustration of Directional Coherence. The input point belongs to Class 1. Given counterfactual candidates CF1 and CF2 at equal distance from the original input point, we deem CF1 as incoherent with respect to the expected effect of changing Feature 1. Intuitively, CF1 suggests to decrease Feature 1, while the effect of increasing either Feature 1 or Feature 2 is to increase the posterior probability of predicting Class 2. For the other counterfactual (CF2), there is an agreement between the direction of marginal changes (changing one feature at a time) and the joint direction of changes resulting in a more coherent counterfactual point.

Mathematically, we formulate directional coherence as a term that quantifies the preference for alignment between joint and marginal directional changes in the feature space necessary to achieve a counterfactual outcome. Then, the Directional Coherence score counts the excess of features which have aligned marginal ($'$) and joint ($*$) directions to increase the model's prediction probability towards the desired outcome y:

$$dcoherence = \frac{1}{n}\sum_{i=1}^{n} \text{sgn}\left((x_i^* - x_i)\,(x_i' - x_i)\right)$$

The information about incoherent features can be leveraged to introduce new constraints or refine the model. In other words, users can specify constraints about how they expect that the change of a certain feature should affect the outcome of the model. The proposed directed coherence measure can be used to incorporate such constraints or expectations in the selection of explanations.

2.2 Using diffusion distance to search for more feasible transitions

A strategy for generating meaningful counterfactual explanations is to develop methodology that emphasizes the feasibility, coherence, and actionability of possible explanations. We propose the utilization of diffusion distance as a metric to assess the connectivity and potential actionability of counterfactual transitions. This approach brings points that are highly connected by numerous short paths into closer proximity, and hence highlighting points for which numerous short routes exist to transition from one point to the other while being on the data manifold. The concept of diffusion distance and its role in detecting counterfactual points that are more ”accessible” from the original instance (in the sense of the existence of numerous short distance routes between the points) is illustrated in Figure 2.2.

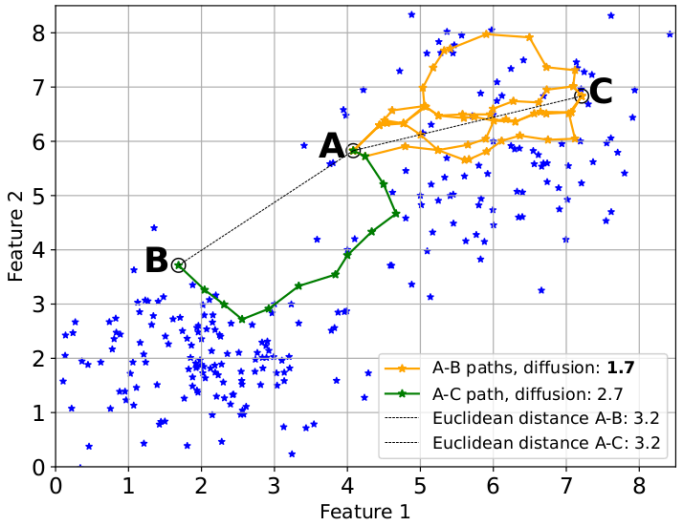

Figure 2.2: Illustration of the concept of diffusion distance and its use for counterfactual search. Points connected by numerous short distance paths (A-C) exhibit a shorter diffusion distance than pairs of points whose connections pass through a bottleneck or low density region (A-B). Note that evaluated by Euclidean distance the pairwise distance A-C and A-B would be exactly the same.

The formal definition of diffusion distance between two points x and y at time t is given by:

$$D_{\text{diff}}(x, y, t)^2 = \sum_z \frac{(p_t(x|z) - p_t(y|z))^2}{\phi_0(z)},$$

Where $p_t(x|z)$ represents the probability of transitioning from point z to x in t steps following a diffusion process (random walk on the graph), and $\phi_0(z)$ is the stationary distribution of the diffusion process at point z. This formula highlights the diffusion distance’s capacity to account for the data’s intrinsic geometry through probabilistic transitions.

2.3. User specified constraints for enhanced actionability

To tailor counterfactual search to end user liking, the framework supports a set of constraints that tailors the search of counterfactual explanation. Supported constraints include:

- Immutability - specified feature will not change its value in counterfactual search.
- Permitted range - the search of counterfactual points will happen for limited range specified by the user.
- Monotonicity - allows to specify the direction of desired counterfactual change.
- Causal dependency - given a pair of features A and B, changing feature A should invoke a change in feature B
- Monotonic dependence - given a pair of features A and B, changing feature A proposes the direction of a change in feature B (increasing or decreasing).
- Rule-based dependency - given a pair of features A and B, changing feature A creates a specific rule by which feature B should change (i.e. B = A-7).

These sets of constraints improved the overall satisfaction and usability of explanations by use-cases.

2.4 Transformed space counterfactual analysis

To enrich the understanding of model behavior and potentially to introduce new combined features derived from genetic programming training, we propose a transformed space counterfactual analysis. By aggregating specific features to create higher-dimensional constructs, we explore the semantic meanings of learned relationships within the model. A concrete example is illustrated in section 3.1.2. While this method does not ensure a seamless transition of suggestions from the transformed space back to the original space, it still provides a significant simplification in explaining the complexities of the original multi-dimensional feature space. This approach allows for more manageable interpretations of the model's behaviour, despite the challenges in direct applicability.

3. Counterfactual analysis for GP model validation and improvement

3.1 Online retail GP model analysis

For Online retail use-case we investigate the probabilistic classification problem of predicting the probability of selecting a time slot. “Given a combination of customer and time slot, will the customer select the slot? And with which probability?”

The training data has 3529 instances and 13 features. The features used in prediction of selection probability of a time slot are the following:

- x_0 – "slotcost";
- x_1 – "slot_start";
- x_2 – "exact_selection_customer_perc";
- x_3 – "rank_cost";
- x_4 – "median_cost";
- x_5 – "partial_selection_customer_perc";
- x_6 – "expanding_avg_days_to_delivery";
- x_7 – "days_since_first_purchase";
- x_8 – "q1_cost";
- x_9 – "max_cost";
- x_{10} – "min_cost";
- x_{11} – "slot_width";

The model studied was trained as a genetic programming expression and was provided by partners:

$$f(x_0, x_1, x_2, x_3, x_4, x_8, x_9) = ((x_2 \cdot x_9) - \mathrm{aq}(x_1, -7645.436000)) + ((x_8 \cdot x_3) \cdot (x_4 - x_0)) ,$$

where $aq(,)$ - is the protected division operator.

3.1.1 Interaction with explanation module on original space features

We started model validation with a counterfactual search on unconstrained original feature space. Below is an example of one instance of counterfactual suggestion to flip the probability of selecting a particular time slot from 0 (or 0.01 in probability) to 1 (or 0.98).

Table 3.1. Example of counterfactual explanation with our counterfactual framework (CoDiCE) for one instance without imposing any constraints.

	slotcost	slot_start	exact_selection_customer_perc	rank_cost	median_cost	partial_selection_customer_perc	expanding_avg_days_to_delivery	days_since_first_purchase	q1_cost	max_cost	min_cost	slot_width	**Output with threshold 0.5**
Input	8.0	4440.0	0	0.96	5	0	2.1	173	3	8	2	150	**0**
CoDiCE	3.36	5120	0.134	0.95	4.9	0.005	-	-	3.41	7.78	-	-	**1**

The suggested counterfactual here is implausible as it does not capture how features are interacting.

For the first iteration of improvements we kept only immutability constraints for max_cost, min_cost, rank_cost and median_cost as these features are computed on the panel level and we can not compute them without access to panel data. The results for the new exploration are listed in Table 3.2.

Table 3.2. Example of counterfactual explanation with our counterfactual framework (CoDiCE) for one instance with immutability constraints for max_cost, min_cost, rank_cost, median_cost.

	slotcost	slot_start	exact_selection_customer_perc	rank_cost	median_cost	partial_selection_customer_perc	expanding_avg_days_to_delivery	days_since_first_purchase	q1_cost	max_cost	min_cost	slot_width	**Output**
Input	6.0	1110	0	0.673913	4	0.0208	2.6	176	3	8	2	120	**0.015**
CF	2.09	2747	0.06	-	-	0.067	-	-	-	-	-	-	**0.981**

This counterfactual can indeed be a possible data point (within data distribution), thus is more useful to validate the model behavior. It was reported that slot_start direction does not align with experts' intuition about the problem. However, the suggested change is in agreement with symbolic expression trained by models, where `slot_start` appears with minus sign and negative coefficient $-\operatorname{aq}(x_1, -7645.436000))$which results in a term contributing positively to the selection probability of the time slot. In this way, counterfactual analysis helped to identify inconsistency of model expression with expert intuition about the problem.

As the next step, we designed a set of constraints that align with use case specifics for counterfactual search to explore slotcost effect in the panel overview.

The list of constraints used for the counterfactual analysis for the panel is:

```
{
   "features": {
       "max_cost":  {
           "type": "immutable"
       },
       "min_cost": {
           "type": "immutable"
       },
       "rank_cost": {
           "type": "immutable"
       },
       "median_cost": {
           "type": "immutable"
       },
       "q1_cost": {
           "type": "immutable"
       },
       "slotcost": {
           "type": "monotonic",
           "direction": "decreasing"
       },
       "slot_start": {
           "type": "monotonic",
           "direction": "decreasing"
       },
       "partial_selection_customer_perc": {
           "type": "monotonic",
           "direction": "increasing"
       },
       "exact_selection_customer_perc": {
           "type": "monotonic",
           "direction": "increasing"
       },
       "slot_width": {
           "type": "monotonic",
           "direction": "increasing"
       }
   }
}
```

Two examples of counterfactual instances aligned with these constraints are presented in Table 3.3.

Table 3.3. Example of counterfactual explanation with our counterfactual framework (CoDiCE) for one instance with immutability constraints for max_cost, min_cost, rank_cost, median_cost and directional constraints for slotcost, slot_start, partial_selection_customer_perc, exact_selection_customer_perc and slot_width.

	slotcost	slot_start	exact_selection_customer_perc	rank_cost	median_cost	partial_selection_customer_perc	expanding_avg_days_to_delivery	days_since_first_purchase	q1_cost	max_cost	min_cost	slot_width	**Output**
Input	8.0	4440	0	0.96	5	0	2.1	173	3	8	2	150	**0**
CF1	4.2	-	0.40	-	-	0.007	-	-	-	-	-	-	**1 (0.99)**
CF2	4.9	-	0.41	-	-	0.0028	-	-	-	-	-	-	**1 (0.97)**

For such a setting we also generate counterfactuals for every slot in a price panel to see what slot should be selected within the panel. Results are shown in Figure 3.1 focusing on the effect of the slot-cost in the selection of the different time slots across the day.

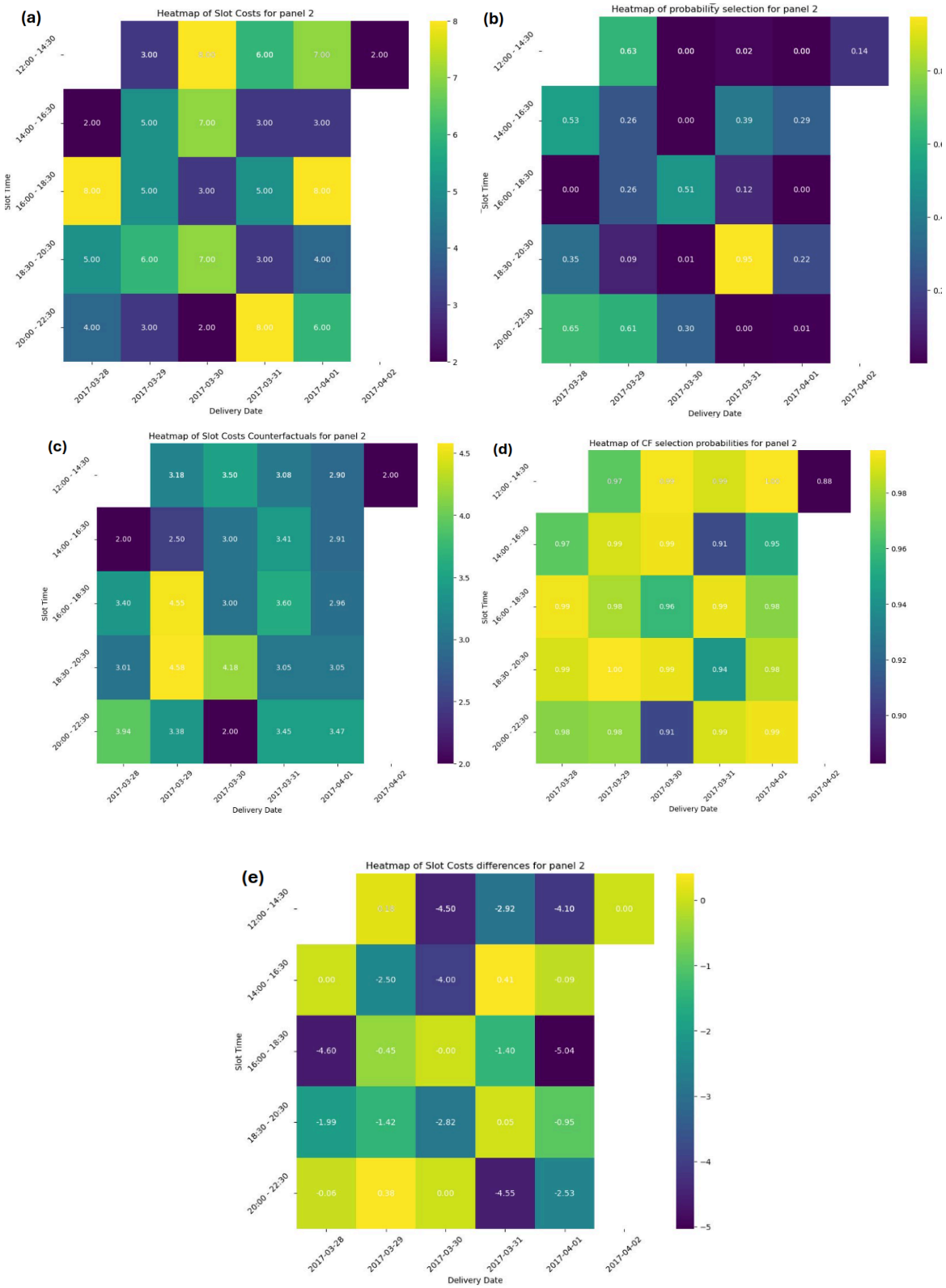

Figure 3.1. Counterfactual analysis for price panel. (a) - original slotcost of every time slot; (b) - selection probability for original panel; (c) - panel with counterfactual slotcost that would make

every slot more attractive; (d) - selection probability of every slot; (e) - difference between original slotcost and counterfactual cost.

The analysis helped to visualize the model behavior under different settings and revealed the inconsistency between intuition and model behavior. The expectation was that slots far off distanced into the future (high slot start) require larger price shifts to steer demand, than slots that are closer to the customer (low slot start), which are typically more preferred by the customer. Therefore, another iteration of the model training is required to adhere to this intuition.

Overall, we aim to continue the learning loop by comparing the explanations across different models to the same input. In the case of having multiple models, evaluating the feasibility, actionability and coherence of counterfactuals for the different models can be used for the selection/preference among competing models.

3.1.2 Interaction with explanation module on transformed space features

Furthermore, we investigate counterfactual explanations using a space of combined features as suggested by the GP model expressions. Such transformation from the original features can be used ro reduce the dimensionality of the original search space or to find semantic meaning of learned relations.

In the original feature space the model has the following expression:

$$f(x_0, x_1, x_2, x_3, x_4, x_8, x_9) = ((x_2 \cdot x_9) - \mathrm{aq}(x_1, -7645.436000)) + ((x_8 \cdot x_3) \cdot (x_4 - x_0)) \,,$$

where we can aggregate some features that appear together to look for a combined semantic meaning. Given the original model f defined on original space X, let's denote model g on transformed space T, where

- $t_0 = x_4 - x_0$ ("median_cost" - "slotcost"), which we call "shifted_cost";
- $t_1 = x_1$ ("slot_start");
- $t_2 = x_2 * x_9$ ("exact_selection_customer_perc"*"max_cost"), which we call "max_exact_selection"
- $t_3 = x_8 * x_3$ ("q1_cost" * "rank_cost"), which we call "negative_attractiveness"

Then the model on the transformed space will take the following expression:

$$g(t_0, t_1, t_2, t_3) = (t_2 - \mathrm{aq}(t_1, -7645.436000)) + (t_3 \cdot t_0)$$

Using this model we run the counterfactual search in the transformed space. Given the instance from original space (see Table 3.4), we generate the corresponding transformed input and its corresponding counterfactual point (see Table 3.5).

Table 3.4. Original instance of interest.

	slotcost	slot_start	exact_selection_customer_perc	rank_cost	median_cost	partial_selection_customer_perc	expanding_avg_days_to_delivery	days_since_first_purchase	q1_cost	max_cost	min_cost	slot_width	**Output**
Input	8.0	4440	0	0.96	5	0	2.1	173	3	8	2	150	**0**

Table 3.5. Input feature in transformed space and its respective counterfactual explanation.

	shifted_cost	max_exact_selection	negative_attractiveness	slot_start	**Output**
Transformed_Input	4.295	0.163	0.91	10814	**0**
CoDiCE	0.764	0.00017	0.437	4908	**1 (0.99)**

The experts reported that the counterfactual explanations in the transformed space were interesting for the exercise of thinking about the meaning of transformed features. The most insightful was the combination of q1_cost and rank_cost which after the transformation was named as “negative attractiveness”. This intuition gave a very clear way on how to evaluate counterfactuals and models, as in order to increase the probability of selecting the slot, the overall negative attractiveness should decrease. Such an approach helped to understand model behavior better and to suggest incorporating negative attractiveness as a combined feature to simplify the GP expression.

3.1.3 Learning loop summary

The counterfactual analysis conducted for the online retail scenario has served as an important learning exercise, illustrating the interplay between advanced data models and domain expertise. Initially, the analysis focused on validating a genetic programming model that predicts the likelihood of customers selecting specific time slots. This phase used counterfactual analysis in an unrestricted feature space, revealing discrepancies between the model's predictions and

expert intuition, particularly regarding the effect of the slot_start feature. This discrepancy highlighted the need for iterative model refinement, leading to adjustments such as applying immutability constraints to certain cost features derived at the panel level. These adjustments made the counterfactuals more plausible and insightful, particularly by showcasing unexpected findings about the influence of time slot positioning and pricing on customer choices. Further exploration using transformed feature space deepened the understanding of the model's behavior, especially the insights from analyzing the "negative attractiveness" feature, which suggested ways to simplify and improve the model.

Feedback from partners played a crucial role in directing the analysis towards practical applications, like optimizing pricing strategies for different customer segments and delivery times. This feedback emphasized the use of counterfactual analysis not only in model validation but also in its potential for guiding business decisions to modify customer behavior. Moreover, this process showed the importance of the balance between using data-driven insights and incorporating human judgment, particularly in cases where domain knowledge is rich and implicit. Overall, the iterative process of analysis, refinement, and partner feedback led to a more refined understanding of the predictive model, highlighting the value of selecting models that align with domain expertise and expectations. Through this process, counterfactual analysis has proven to be important not only for model validation but also to emphasize the importance of an iterative, feedback-driven approach in model development and validation in real-world scenarios.

3.2 Energy case

The next use case problem is that of predicting the energy consumption of a residential building. The model under investigation is trained on autumn data. The original symbolic expression obtained by GP is a large complex tree:

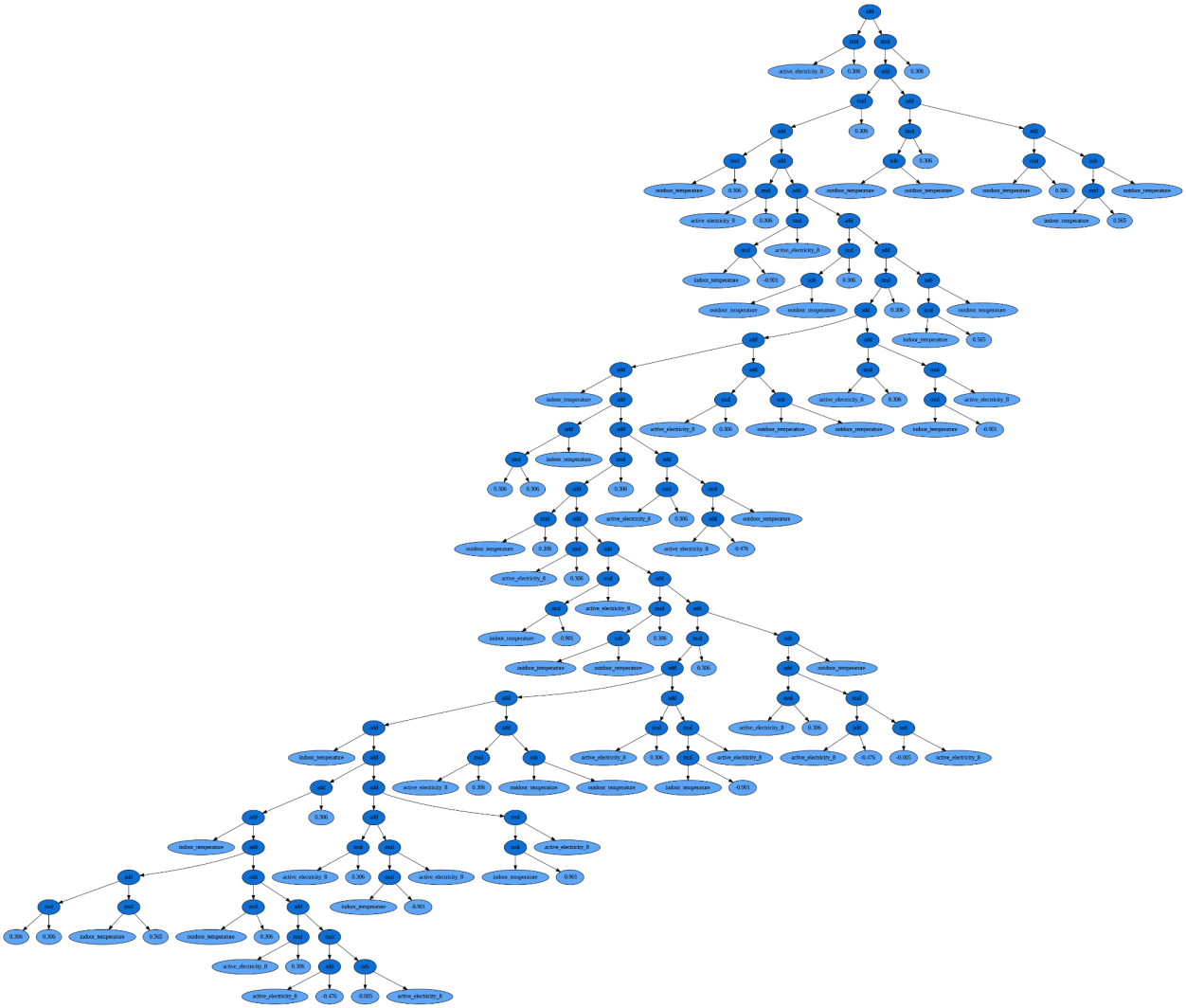

Figure 3.2. Trained model for autumn data for energy use case.

The notation is as follows: a - active_electricity_8, i - indoor temperature and o - outdoor temperature. The found model can be written in mathematical notation in the following way:

$$f(a,i,o) = 0.306 * a + 0.306 * (A_1 + 0.306 * (o - o) + 0.306 * o + 0.565 * i - o)$$

, where

$$A_1 = 0.306*(0.306*o+0.306*a+(-0.901*i*a+0.306*(o-o)+A_2+0.306*a+(a-0.476)*(a-0.005)-o)$$

, where

$$A_2 = 0.306*(0.306*a-0.901*a*i)+((o-o)+0.306*a)+i-0.901*i*a+0.306*a-0.901*i*a+0.306+i+A_3)$$

, where

$$A_3 = 0.306 * 0.306 + 0.525 * i + 0.306 * o + 0.306 * a + (a - 0.476) * (a - 0.005)$$

After the first stage of simplification we obtained the following expression:

$$f(a,i,o) = a \times (0.306 \times 1.302) + o \times (0.306^3 \times 1.306) + a \times i \times (-0.901 \times 0.306 \times 1.918) + (a - 0.476) \times (a - 0.005) \times 0.306 \times 1.306 + (o - o) \times 0.306^2 \times 1.612$$

It can be simplified even further to:

$$f(a,i,o) = 0.399636 \times a^2 - 0.5288 \times a \times i + 0.206187 \times a + 0.03742 \times o + 0.0009$$

This expression is considerably more intuitive and can help to evaluate the intuition about model behavior. We see from expression that increase of outdoor temperature leads to increase in consumption. Indoor temperature has reverse relations with outdoor temperature, meaning decrease of indoor temperature leads to increase of consumption.

3.1.1 Counterfactual analysis for energy use case

While the simplicity of the reduced model might allow for different strategies, we tested the full counterfactual optimization framework (including the biases towards feasibility and coherence, and possible constraints) on a set of test instances. In particular, we searched for counterfactual scenarios that decrease electricity consumption by at least 5%. The desired range of decrease was set to [10%,5%]. We applied two searches of our counterfactual framework CoDiCE with diffusion distance and with weighted L1 distance and compared their results with DiCE. CoDiCE offers more directionally coherent explanations, meaning the direction of change respects the marginal direction of prediction. For instance, if fixing all other variables, indoor temperature drop decreases electricity consumption, then a counterfactual explanation should respect this relation even if other features are changing.

Results for the evaluation metrics (comparing with DICE) of the counterfactuals for this case are shown in Tables 3.6.

Table 3.6. Evaluation metrics comparison for our counterfactual framework (CoDiCE) with diffusion distance and with L1 norm and with DiCE framework. Standard deviation is calculated over 20 samples.

Dataset	Metric	Validity ↑	Diffusion ↓	L1 continuous ↓	Coherence ↑
Energy	$\text{CoDiCE}_{\text{diff}}$	100%	0.0049 ± 0.03	0.52 ± 0.33	$\mathbf{0.67 \pm 0.04}$
	$\text{CoDiCE}_{\text{L1}}$	100%	$\mathbf{0.0026 \pm 0.02}$	$\mathbf{0.41 \pm 0.26}$	0.63 ± 0.11
	DiCE	100%	1.64 ± 2.21	0.51 ± 0.47	0.62 ± 0.11

The validity in percentages shows how many counterfactuals reached the desired outcome with a target of interest. We see that all methods succeeded in finding counterfactual explanations for these settings, although CoDiCE reported a more coherent explanation with smaller distance than DiCE for both versions of distances. CoDiCE offers an interactive way to adjust preferences for different objectives and incorporate user constraints. We also report ablation experiments in Table 3.7 under different weight parameters for the objective function (1).

Table 3.7. Evaluation metrics tinder various ablations of diffusion, sparsity, and directional coherence terms for CoDiCE on the energy use case for 20 test instances.

Inactive terms	Validity	Diffusion	L1 continuous	Sparsity	Coherence
$\lambda_1 = 0$ - diffusion	100%	0.015 ± 0.04	1.03 ± 0.51	1	0.64 ± 0.05
$\lambda_2 = 0$ - sparsity	100%	0.0012 ± 0.004	0.76 ± 0.46	1	0.64 ± 0.05
$\lambda_3 = 0$ - dcoherence	100%	0.0011 ± 0.003	0.61 ± 0.45	1	0.50 ± 0.16

We note that when the diffusion distance term is inactive we have the highest average distance, while when the directional coherence term is 0 the coherence metric drops. However, we do not see any effect with sparsity, since we have only 3 features for model prediction, while the sparsity metric has quite high sensitivity, which means that even small change in feature will put sparsity to 1.

The example of the original instance and respective counterfactual scenario without constraints is shown in Table 3.8.

Table 3.8. Input instance and corresponding counterfactual explanation with CoDiCE and DiCE without any constraints invoked.

	outdoor_temperature	indoor_temperature	active_electricity_8
Input instance	22.8	23.5	2351.2
CoDiCE	24.9	**23.7**	**2374.4**
DiCE	**22.8**	13.8	2815.2

Overall CoDiCE produces a closer counterfactual explanation. Nevertheless, it is difficult for both methods to follow the model direction, since the importance and scale of active_electricity_8 outweighs the directions of temperatures. We see that CoDiCE produced a closer counterfactual point in terms of all features, however, both points are incoherent with respect to signs of the model. Knowing that experts reported that it is unfeasible to change active_electricity_8, since it is historical observation data, and hence we imposed constraints for this feature (see Table 3.9).

Table 3.9. Input instance and corresponding counterfactual explanation with CoDiCE and DiCE with immutability constraints for active_electricity_8.

	outdoor_temperature	indoor_temperature	active_electricity_8
Input instance	22.8	23.8	2697.5
CoDiCE	**22.6**	**20.2**	2697.5
DiCE	23.4	19.8	2697.5

Given that the current GP model has only 3 features, we did not explore transformed space counterfactuals for this use case.

3.1.2 Learning loop summary

The counterfactual analysis, facilitated by the CoDiCE framework, aimed to explore scenarios for reducing electricity consumption by at least 5%, utilizing both diffusion distance and weighted L1 distance for comparison against the DiCE framework. CoDiCE's strength lay in generating directionally coherent explanations, aligning changes in features with expected impacts on consumption, thereby offering more reliable and closer counterfactual points than DiCE. This coherence underscores the framework's capability to respect the intrinsic relationships within the data, such as the decrease in indoor temperature leading to reduced electricity usage, even when other variables shift.

Overall CoDiCE enhanced the analysis with its interactive features, allowing for the adjustment of preferences and the incorporation of user-defined constraints, enriching the exploratory process. The evaluation of CoDiCE under various configurations showed the relative impact of

diffusion distance, sparsity, and directional coherence on the counterfactual explanations. More importantly, the counterfactuals generated by the framework were satisfactory to the user.

Furthermore, the energy case was tested with the natural language interactive chat platform which is to integrate CoDiCE for counterfactual explanations, as described in Section 4.

4. Iterated dialog interface based on LLMs

4.1 Introduction

Large Language Models (LLMs) like OpenAI's GPT series have revolutionized dialogue applications, making virtual conversations more intuitive and human-like. These models enable chatbots and virtual assistants to provide highly personalized and context-aware interactions across various sectors. LLM-powered chatbots deliver instant and accurate responses in customer service, enhancing customer satisfaction. Virtual assistants, including Google Assistant and Siri, have become more adept at handling complex conversations, offering users a seamless interaction experience. Educational tools, such as language learning apps, leverage LLMs to simulate natural dialogues, making learning more engaging. Additionally, mental health platforms use these models to provide therapeutic conversations, offering support and companionship to users in need.

The impact of LLMs on dialogue applications demonstrates their potential to improve and personalize human-computer communication, promising even more sophisticated conversational agents in the future [4], [3]. Recent research shows practitioners struggle to understand and choose among model explanations, particularly without deep data science knowledge [1]. Grasping how models work, including identifying and correcting errors, is a major challenge. This issue hinders the broader adoption of machine learning models, especially where they are expected to enhance decision-making. A solution to this problem could be the adoption of LLMs.

LLMs streamline communication between humans and ML models, allowing individuals to interact directly with an ML model. This interaction lets users ask about data statistics, model explainability, hypothetical situations, and other related topics.

In this context, we introduce our specialized adaptation of an LLM-based conversational interface for the energy use case.

4.2 TalkToModel Framework

TalkToModel is a dynamic conversational platform powered by LLMs designed to field inquiries related to ML binary classification algorithms. It acts as an intelligent dialogue system capable of understanding natural language and producing insightful replies. Within its framework, it features an operational component that builds explanations, such as counterfactuals and feature importance, which are then incorporated into chats with the system users.

TalkToModel Framework parses the user's unstructured text in a structured text following the next steps:

1. **Grammar Construction** A grammar that outlines the acceptable parsing options is constructed based on the dataset and model provided by the user.
2. **Parse Generation** It then generates pairs of utterances and their corresponding parses for the given dataset and model.
3. **LLM fine-tuning** TalkToModel uses a fine-tuned LLM to convert user utterances into their corresponding parses.
4. **Answer generation** The system answers to the user, incorporating the outcomes of the parsed commands. These responses contextualize the results and suggest avenues for further queries.

The interface was also re-engineered since the TalkToModel algorithm is designed for binary classification, and the TRUST-AI project focuses on different classification and regression problems.

4.3 Integration of Energy Use case Data

TalkToModel [2] is a research prototype of a system created to demonstrate interactive language conversations' capabilities for explainable machine learning. The original publication included three datasets, and in all of these cases, a machine-learning model was applied to solve a binary classification task.

The energy use case dataset exemplifies a regression problem, requiring the machine learning model to forecast a singular, continuous value from multiple numerical inputs. Specifically, a Genetic Programming ML model is employed for this task, offering high predictive accuracy and optimal transparency.

Given these requirements, we first adjusted the TalkToModel code to support regression problems and genetic programming ML models **Figure 4.1**. Secondly, we narrowed down the research scope of the system, including simplifying explanation selection, to optimize the overall performance and make the system production-ready.

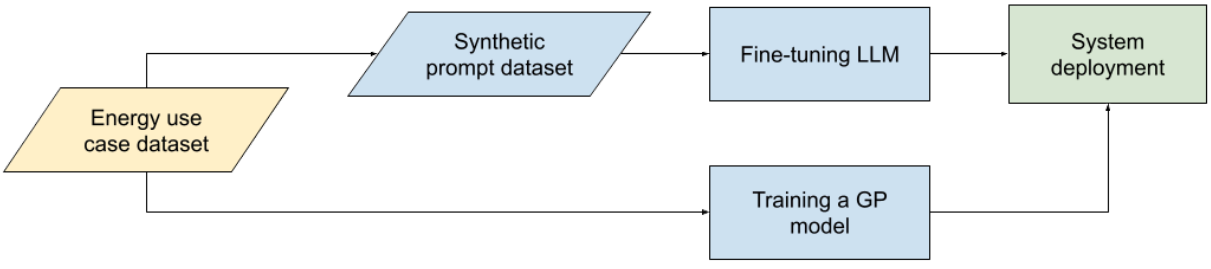

Figure 4.1. TalkToModel integration and deployment

TalkToModel's main feature is its interactive ability, powered by the real-time execution of specific predefined actions. For instance, users can instruct the model to categorize instances within the test dataset according to particular criteria and then request predictions and explanations or pose questions in natural language. A LLM interprets the user's input as a command from a predetermined collection. Upon successful parsing, the system carries out this command and delivers a clear, user-friendly response via a chat message based on the action's result.

Adapting the TalkToModel system to support regression problems involved making changes across the entire code base, as the system relies heavily on the prediction method of the given ML model. Apart from refactoring out the references that conflicted directly with the specifics of a regression model, such as calling the classification-specific methods on the model instance, it was also necessary to locate and rectify all indirect references to prediction outcomes of a classification model, such as reliance on class labels.

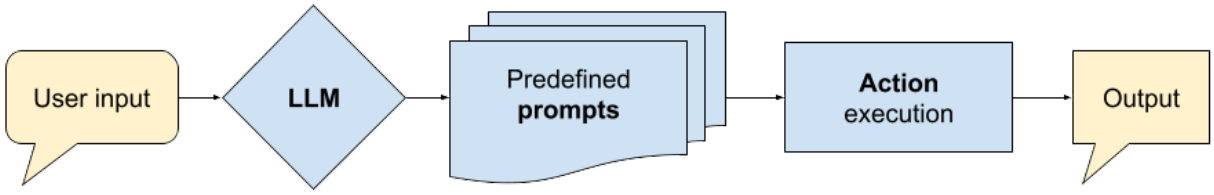

Figure 4.2. TalkToModel NLP pipeline

The modifications primarily targeted the code for **actions** depicted in **Figure 4.2**, altering the execution process of interactive commands. Additionally, adjustments were necessary in the overall system configuration and caching mechanisms. For instance, the setup for generating counterfactual explanations with DiCE and selecting feature importance with SHAP needed revisions. The original study highlighted detailed comparisons between various explanation methods, notably LIME and SHAP, producing results for further analysis and automatic evaluation to determine the most suitable explanation for the user. We streamlined this process by opting for SHAP explanations exclusively, aiming to enhance the system's performance and code maintainability.

Our subsequent phase involved refining the **prompt** dynamic dataset, crucial for enhancing the accuracy with which the LLM interprets user commands into actionable instructions. Specifically, this entailed eliminating irrelevant prompts, such as those concerning class labels and assessments of classification model precision. We also updated and introduced new prompts to accommodate novel functionalities, like inquiries regarding the regression model’s specifics or accuracy.

Within TalkToModel, this dynamic prompt dataset, when used in conjunction with the energy use case dataset, facilitates the creation of an extensive synthetic dataset for fine-tuning the LLM.

This dataset pairs "user utterances" with their corresponding "parsed actions," enabling the LLM to translate user inputs into executable commands accurately. Following the approach in the original TalkToModel study, we assessed T5 LLMs of various sizes. We opted for the T5 large model, which strikes an optimal balance between accuracy and performance. We fine-tuned this model using a single Nvidia V100 Tensor Core GPU at the University of Tartu HPC Center (UTHPC).

We made necessary system adjustments to allow all stakeholders to assess and engage with the chat interface. These modifications included enhancements to logging and implementing a straightforward method for tracking user sessions. This setup aids in collecting feedback and monitoring the system for any issues. We refer to figures **4.3**, **4.3** and **4.5** for illustrative examples of interactions.

The system operates in a virtual machine with 4 virtual CPUs and 16 GB of RAM It is accessible at http://nlpxai.xyz. Remarkably, the deployment does not necessitate a GPU for real-time, instant inference using a large T5 model, rendering the setup cost-effective and adaptable to various applications. In our forthcoming research, we plan to investigate the potential and benefits of employing other LLMs, especially Llama2, for analogous tasks within the same hardware limitations.

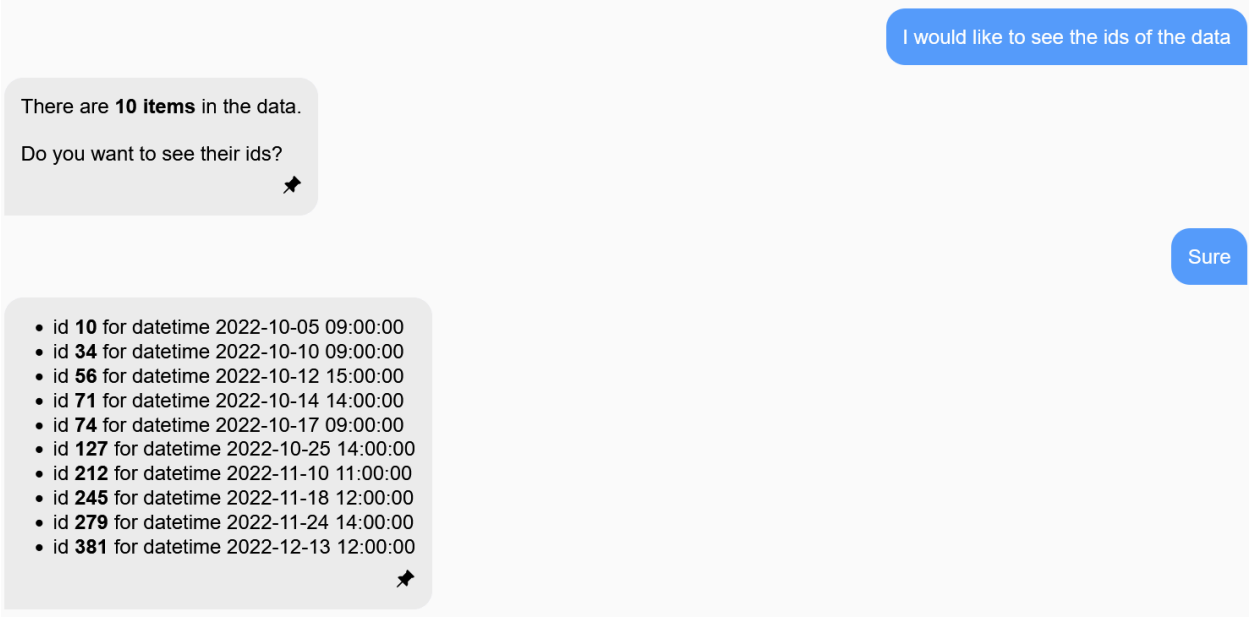

Figure 4.3. TalkToModel chat example

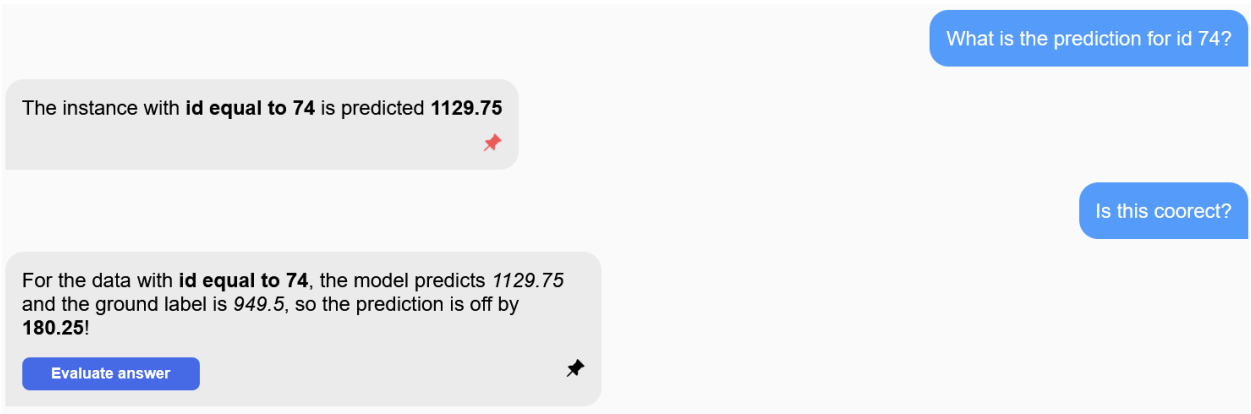

Figure 4.4. TalkToModel robust utterance parsing

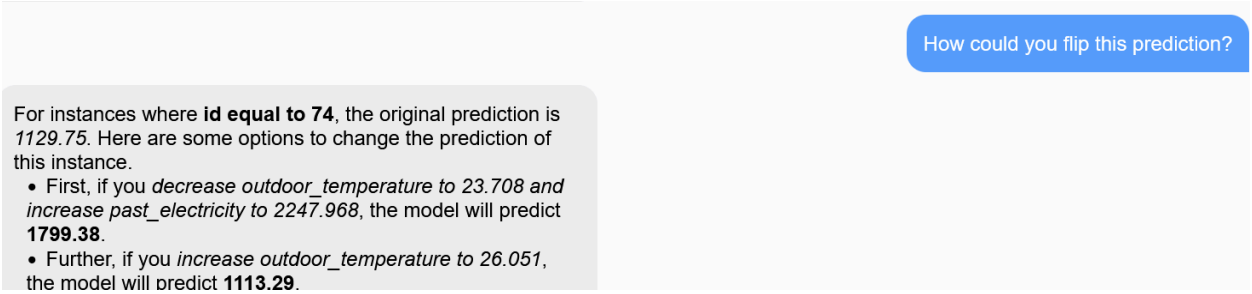

Figure 4.5. TalkToModel functionality example

4.4. Future work

We are advancing along three paths in response to the re-engineered interactive interface. Firstly, we are partnering with Apintech to design a scenario that involves interactions between human users and Genetic Programming (GP) models, aiming to evaluate the interface's performance with input from Apintech's experts. The feedback from this evaluation will guide a reengineering process to refine the interface.

Simultaneously, another focal point of our research is incorporating cutting-edge Large Language Models (LLMs) into the interface. An example of this effort is integrating the Llama model, an open-source LLM introduced last year by Meta. This initiative seeks to enhance the interface by integrating the latest advancements in LLM technology.

Finally, the counterfactual module (CoDiCE) that incorporates diffusion distance developed for this project, presented in the previous sections of our deliverable, is to be incorporated into the tool.

Conclusions

We reported on the generation of counterfactual explanations by a novel framework and how it assisted in the learning loop with users to validate and obtain insights from GP models. The novel framework (CoDiCE) incorporates two terms that formalize the preference for feasible and coherent counterfactual explanations for a model's outcome. The iterated loop between the counterfactual explanations and the domain experts was critical to identify unexpected behavior of the model (e.g. against deeply held expectations of the role of certain features). The iterative process of the analysis through counterfactuals and user feedback led to practical model validation and selection criteria. This was especially useful for the online retail case. In conclusion, the energy case counterfactuals found by the framework are satisfactory and the online retail case requires enhanced model training. Current work aims to apply the iterated learning loop to a diversity of GP models. In addition, we have specialized a LLM-based iterated dialog interface for users to interact in natural language with a explainer module (including the counterfactual framework) for the energy use case. This has resulted in improved human interactions with GP models by providing customized explanations.

References

[1]. Lakkaraju, H., Slack, D., Chen, Y., Tan, C., Singh, S.: Rethinking explainability as a dialogue: A practitioner's perspective (2022)

[2]. Slack, D., Krishna, S., Lakkaraju, H., Singh, S.: Explaining machine learning models with interactive natural language conversations using TalktoModel. Nature Machine Intelligence (Jul 2023). https://doi.org/10.1038/s42256-023-00692-8, https://doi.org/10.1038/S42256-023-00692-8

[3]. Ward, N.G., DeVault, D.: Ten challenges in highly-interactive dialog systems. In: AAAI Conference on Artificial Intelligence (2015), https://api.semanticscholar.org/CorpusID:18795232

[4]. Zhang, Z., Takanobu, R., Zhu, Q., Huang, M., Zhu, X.: Recent advances and challenges in task-oriented dialog system (2020)