SAFER: Semantic Assessment of Feasible Explanations for Algorithmic Recourse

Muhammed Ashrah[1], Ashwin Arun[2], Sahana Karthik[3] and Raghava Mutharaju[4]

[1]Jamia Millia Islamia, New Delhi, India
[2]Cochin University of Science and Technology, Kochi, India
[3]VIT University, Vellore, India
[4]Mehta Family School of Data Science and AI, IIT Palakkad, Kerala, India

Abstract

Algorithmic recourse aims to provide individuals with actionable paths to overturn unfavourable model decisions. Counterfactual explanations are the dominant means of operationalising this goal. However, existing generation methods lack formal semantics and any mechanism for post-hoc feasibility validation. Current generation methods handle immutability through binary feature level flags applied to tabular output. This approach is blind to cross-feature dependencies. A counterfactual may, for instance, extend an individual's employment tenure without a corresponding change in age. No existing generation method is designed to detect such inconsistencies. We address this gap by bridging Explainable AI and Semantic Web technologies through a post-hoc validation pipeline. Tabular counterfactuals are lifted into RDF Knowledge Graphs via a formal OWL ontology and declarative RML mappings. These are then validated against a four-layer SHACL constraint framework. The central technical contribution is a suite of SHACL-SPARQL cross-instance rules that enforce immutability, directional constraints, and jump limits. These constraint types are conditional and relational, and therefore cannot be evaluated at generation time when instances are processed independently. The validation framework is fully decoupled from the counterfactual generator. This makes it both extensible and generator-agnostic. Domain experts can introduce new constraints without modifying any generative model. The same framework applies uniformly across generators with differing internal assumptions. We evaluate SAFER on two datasets, German Credit and COMPAS, using two generation methods, DiCE and ActionRules. Across all four dataset and generator combinations, the pipeline produces structured and machine-readable SHACL validation reports. These reports establish a basis for formal cross-generator comparison and for audit trails suited to emerging regulatory requirements. Our implementation is publicly available at https://github.com/kracr/semantic-alg-recourse.

Keywords

Algorithmic Recourse, Counterfactual Explanation, SHACL, Knowledge Graph, Ontology, Explainable AI

1. Introduction

Automated decision making systems are increasingly used to determine access to housing, employment, and credit. The European Union's General Data Protection Regulation (GDPR)[1] grants individuals not only the right to an explanation for automated decisions but also the right to contest them [1]. Two related but distinct concepts have emerged in response to this requirement: counterfactual explanations and algorithmic recourse.

Counterfactual explanations answer the question: what would need to be different for the model to make a different decision? [1]. They describe the minimal set of feature changes required to flip a model's prediction, focusing on what would alter the outcome. Algorithmic recourse, by contrast, answers a different question: what can an individual actually do to achieve a desirable outcome? [2, 3]. Recourse is prescriptive. It focuses on the actionable steps an individual can take, within realistic constraints, to change their circumstances and obtain a favourable decision. Recourse must consider

XAI-KG@ISWC2026 - 2nd International Workshop on eXplainable AI and Knowledge Graphs, co-located with ISWC 2026, October 25-29, 2026, Bari, Italy

muhammed2306509@st.jmi.ac.in (M. Ashrah); ashwin.a.nair7@gmail.com (A. Arun); sahana@vit.ac.in (S. Karthik); raghava@iitpkd.ac.in (R. Mutharaju)
0009-0001-4442-3684 (M. Ashrah); 0009-0001-2587-7058 (A. Arun); 0009-0008-1855-8447 (S. Karthik); 0000-0003-2421-3935 (R. Mutharaju)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]https://gdpr-info.eu

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-07

CEUR-WS.org/Vol-4318/paper6.pdf

not only feature mutability but also effort, time horizon, feature dependencies, and individual agency. A counterfactual may be technically valid yet provide no practical recourse if the suggested changes are infeasible in the real world.

The foundational work of Wachter, Mittelstadt, and Russell [1] established counterfactual explanations as a legally viable approach to algorithmic recourse. Motivated by the GDPR right to explanation, they argued that meaningful explanations can be provided without revealing a model's internal workings by generating a counterfactual that flips the prediction while remaining close to the original instance.

A rich ecosystem of counterfactual generators now exists. DiCE [4] provides diverse counterfactual sets via genetic, random, and KD-tree methods. GenRE [5] introduces counterfactual reasoning using eBERT embeddings. ElliCE [6] provides provably robust algorithmic recourse via Rashomon set approximations. These methods generate multiple counterfactuals efficiently.

However, existing methods output counterfactuals as flat CSV or JSON objects with no formal semantics and no post-hoc mechanism to verify that the generated explanation is coherent. A counterfactual that flips the model's prediction is treated as valid by definition. This is a dangerously narrow view of validity. Consider a counterfactual that increases employment tenure from two to five years while keeping age unchanged at 25. The generator treats this as valid because age is marked immutable and employment tenure is marked mutable. But this explanation violates the basic laws of temporal causality. The generator cannot distinguish between what is mathematically possible and what is actually possible in the real world. Miller [7] argues that effective explanations are selected for relevance. A counterfactual that is temporally impossible fails this criterion. It is not a meaningful alternative for the individual seeking recourse.

Our pipeline, named SAFER, bridges the two perspectives. We generate counterfactuals and validate them against recourse criteria, moving from technical validity to practical actionability. We introduce a semantic validation pipeline grounded in the Semantic Web stack, with the following contributions.

1. **A declarative RDF conversion process** using RML mappings that transforms tabular counterfactual output into RDF Knowledge Graphs, enabling method-agnostic integration. The mappings are grounded in a lightweight OWL ontology that captures feature classifications and cross-instance relationships between original and counterfactual instances.
2. **A four-layer SHACL validation framework** comprising data integrity checks, hard immutability constraints, feasibility constraints including directional rules and jump limits, and risk warnings. The SHACL-SPARQL cross-instance rules enforce constraints that generation time checks cannot express, such as limiting employment tenure jumps and preventing unrealistic savings increases.

The framework is post-hoc and method-agnostic. We do not modify the generator. We validate its output. This separation of generation and validation is the foundation of trustworthy algorithmic recourse.

2. Related Work

2.1. Counterfactual Explanations

Counterfactual explanations describe the minimal changes required to flip a model prediction, answering the question: what would need to change for a different decision to be reached? [1]. This formulation was motivated by both legal and practical considerations. The GDPR grants individuals the right to an explanation for automated decisions, yet requiring full model transparency would conflict with trade secret protections. Counterfactual explanations offer a pragmatic middle ground: they provide meaningful recourse without requiring access to a model's internal parameters.

The foundational formulation of Wachter et al. [1] treats counterfactual generation as an optimisation problem that minimises distance to the original instance while ensuring the prediction is flipped. This closest possible world approach has become the dominant paradigm. However, as subsequent work has shown, minimality alone is insufficient for practical recourse.

Verma et al. [8] provide a comprehensive survey of desirable properties for counterfactual explanations, including validity, actionability, sparsity, plausibility, and causality. Validity ensures the counterfactual flips the prediction. Actionability requires that changes are under the individual's control. Sparsity favours explanations that change few features. Plausibility requires counterfactuals to remain realistic with respect to the data distribution. Causality requires respect for causal relationships between features. The survey identifies an inherent tension among these properties: improving diversity may reduce plausibility, and ensuring causality may require sacrificing sparsity.

DiCE [4] is one of the most widely used counterfactual generation methods. It argues that a single counterfactual is often insufficient, as individuals may face practical constraints that prevent them from adopting a specific recommendation. DiCE generates diverse sets of counterfactuals by balancing three objectives: validity, proximity, and diversity. It uses Determinantal Point Processes to encourage diversity. Users may specify immutable features and permitted ranges, but the constraint mechanism remains binary. A feature is either immutable or mutable, with no support for conditional or relational constraints.

ActionRules [9] offers a rule based approach to counterfactual generation through the Action Apriori algorithm. Unlike gradient based methods such as DiCE, ActionRules generates explicit action rules describing how to change stable attributes in order to transform a classifier prediction from an undesired class to a desired class. ElliCE [6] addresses model uncertainty by generating counterfactuals that remain valid across multiple models with similar predictive performance, though this emphasis on robustness comes at the expense of diversity and actionability. GenRE [5] uses generative models to produce counterfactuals through forward sampling, offering advantages in plausibility and computational efficiency, though ensuring actionability remains challenging.

Miller [7] provides a foundational perspective on explainable AI, arguing that explanations are fundamentally social processes involving communication between an explainer and an explainee. This framing reinforces the distinction between technical validity and practical actionability that motivates the present work.

2.2. Algorithmic Recourse

Algorithmic recourse extends the counterfactual paradigm by emphasising the individual's capacity to act. Where counterfactual explanations answer what would need to change, recourse answers what the individual can actually do to achieve a desirable outcome. This shift introduces additional considerations: feasibility, agency, and the temporal and effort costs of implementing changes.

Upadhyay et al. [2] formalise recourse as a decision making problem that accounts for individual preferences and constraints, introducing recourse-aware models that guide individuals toward actionable paths and emphasising that recourse must be personalised to the individual's situation.

The counterfactual literature has long emphasised actionability as a desirable property [8], yet the field has largely failed to operationalise this concept beyond binary mutability. A feature is treated as actionable if and only if it is mutable. True actionability requires considering time horizon, effort, cross-feature dependencies, and temporal coherence. Most generators assume that if a feature is mutable, it is equally actionable regardless of the time or effort involved or the dependencies it entails.

2.3. The Gap Between Generation and Validation

A common thread running through the literature is that generators enforce constraints during generation but provide no mechanism for post-hoc validation. DiCE allows users to specify immutable features, but once a counterfactual is generated there is no independent audit layer to verify that the output satisfies those constraints.

This is not a theoretical concern. The genetic search method in DiCE is stochastic and is not guaranteed to find a feasible solution. In practice, DiCE may return counterfactuals that violate user specified constraints, and the user has no way to detect this without manual inspection.

More fundamentally, the constraints that generators can enforce are limited to those expressible within their optimisation objective. A generator can enforce that a feature is immutable, but it cannot enforce that employment tenure can increase only if age increases proportionally. This is not a limitation of any specific method. It is a structural limitation of the generation first paradigm. The solution is a separation of concerns: generation and validation must be decoupled, and validation must be performed by an independent layer with a richer constraint language.

2.4. Semantic Web Technologies for Validation

The Semantic Web stack provides a principled foundation for validating structured data. RDF enables interoperable knowledge representation. OWL provides formal semantics for ontological reasoning. SHACL offers a declarative and W3C standard language for validating RDF graphs against constraint shapes [10, 11].

SHACL has been applied to validate machine learning pipelines [12] and to assess data quality [13]. A key strength of SHACL is its support for SPARQL based constraints, which can express complex conditions involving multiple properties and instances. This is essential for cross-instance validation, such as comparing feature values between an original instance and its counterfactuals.

To our knowledge, no prior work has applied SHACL to the validation of counterfactual explanations. No existing work introduces cross-instance SPARQL rules for temporal coherence in the context of algorithmic recourse. SAFER bridges Explainable AI and Semantic Web technologies, bringing the rigour of formal constraint validation to the problem of counterfactual recourse. The result is a framework that is formal, extensible, post-hoc, and method-agnostic.

3. Approach

Our approach decouples counterfactual generation from validation. Rather than modifying the generator to enforce additional constraints, we allow any generator to produce its outputs and validate them independently after generation. This separation of concerns has two benefits. First, it allows any counterfactual generator to be integrated into the pipeline without modification. Second, the validation layer provides a formal and machine-readable assessment of explanation quality that is grounded in semantic web standards.

3.1. Pipeline Overview

The pipeline consists of four stages, as illustrated in Figure 1. The input data is first used to train a machine learning model. The trained model is then passed to a counterfactual generator, which produces candidate counterfactual instances. These instances are lifted into a knowledge graph through a formal OWL ontology and declarative RML mappings. Finally, the knowledge graph is validated against a four-layer SHACL constraint framework, and the results are compiled into a structured validation report.

The pipeline is designed so that the first two stages, model training and counterfactual generation, are entirely standard and require no modification. The novel contribution of this work lies entirely in the final two stages: knowledge graph construction and SHACL validation.

3.2. Step 1: ML Model Training

The first stage trains a classifier on the input tabular data. The pipeline treats the model as a black box: the generator only calls its prediction function. A saved model can also be loaded directly instead of retraining. This stage is deliberately isolated so the framework remains model-agnostic in design.

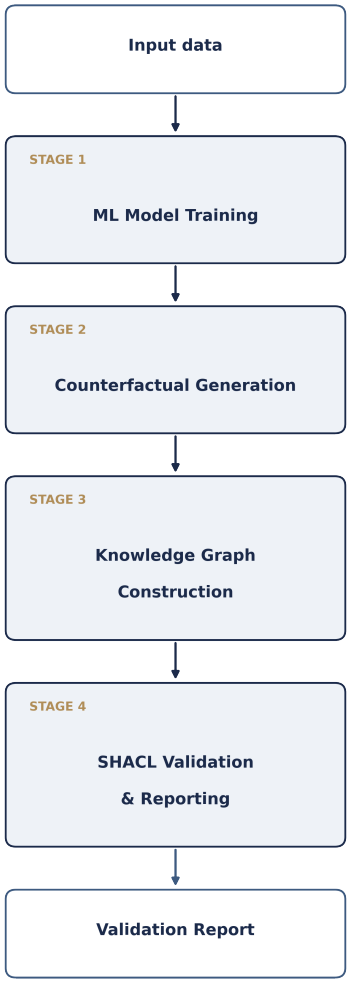


Figure 1: The SAFER pipeline. Input data is used to train a model and generate counterfactuals. The counterfactuals are lifted into a knowledge graph via RML mappings and validated against a four-layer SHACL constraint framework. The output is a structured validation report covering integrity, coherence, and counterfactual quality.

3.3. Step 2: Counterfactual Generation

The second stage produces candidate counterfactual instances using any off-the-shelf counterfactual generator. The generator takes the original instance and the trained model as input and produces a set of counterfactual instances that flip the model prediction. In this work we evaluate two generators, DiCE and ActionRules, but the framework places no constraints on which generator is used. The output of this stage is a set of tabular instances that are passed to the Knowledge Graph construction stage.

3.4. Step 3: Knowledge Graph Construction

The third stage of the validation pipeline transforms tabular counterfactual output into a formal RDF knowledge graph. Dataset features are first classified into three categories: immutable features that cannot change across any counterfactual, mutable features that may change freely, and ordinal features whose changes must respect a defined direction or magnitude.

This classification is encoded in a formal OWL ontology that captures the semantic relationships between features, including cross-feature dependencies such as the relationship between age and employment tenure. RML [14] mappings are then used to declaratively convert tabular instances into RDF triples. The mappings specify classes, properties, datatypes, and the relationships between original and counterfactual instances. Because the mappings are declarative, they can be adapted to new datasets

without changes to the validation logic.

The ontology, SHACL shape graph, and RML mapping are authored separately for each dataset rather than sharing a single common ontology across them. Each dataset has its own configuration specifying dataset-specific feature classifications, immutability sets, and constraint thresholds. This allows the framework to adapt to a new domain by supplying a new configuration rather than modifying the validation logic itself

3.5. Step 4: SHACL Validation and Reporting

The RDF knowledge graph is validated against a four-layer SHACL constraint framework [10]. The four layers are designed to check progressively richer properties of the generated counterfactuals.

- **Layer 1: Data Integrity.** Validates structural correctness, including datatypes, value ranges, and the presence of required properties. This layer catches malformed or incomplete counterfactual instances before deeper validation is attempted.
- **Layer 2: Hard Immutability.** Verifies that features designated as immutable remain unchanged between the original instance and the counterfactual. Violations at this layer indicate that the generator has produced a counterfactual that is invalid by construction.
- **Layer 3: Feasibility Constraints.** Enforces directional constraints, jump limits, and cross-feature dependencies using SHACL-SPARQL rules. This is the most technically novel layer. Because these rules operate across both the original and counterfactual instances simultaneously, they can express constraints that no generation time check can enforce. For example, a rule can detect that employment tenure has increased while age has remained constant, a temporally impossible combination.
- **Layer 4: Counterfactual Quality.** Assesses higher level properties of the counterfactual set, including sparsity, proximity to the original instance, and plausibility relative to the training data distribution.

After validation, the pipeline produces a structured and machine-readable SHACL validation report. The report records which constraints were satisfied and which were violated, along with the specific feature values responsible for each violation. This report serves two purposes. For individual users, it identifies which counterfactuals represent genuinely feasible recourse actions. For auditors and regulators, it provides a formal and reproducible audit trail of the explanation quality produced by a given generator on a given dataset.

3.6. Worked Example

We illustrate the pipeline for one feature, `employment`, which records an applicant's employment tenure as a categorical band. The dataset German Credit uses five bands: unemployed, less than one year, one to four years, four to seven years, and seven years or more. To compare an original value with its counterfactual value, we encode these bands as integers 1 through 5. The ontology therefore defines two paired properties on `foaf:Person`: `ex:originalEmploymentOrdinal` and `ex:newEmploymentOrdinal`. An RML mapping reads the tabular columns `employment`, `original_employment_ordinal`, and `new_employment_ordinal` and writes them to these properties.

The rule below, `ex:EmploymentAgeCohesionRule`, then checks a temporal constraint: if employment tenure increases, age must also increase. The rule is written as a SPARQL query over the knowledge graph. For each counterfactual applicant node `?p`, it reads the original and new employment ordinals and the original and new ages. It selects the node as a violation whenever the employment ordinal rose while the age did not.

```
ex:EmploymentAgeCohesionRule
    a sh:NodeShape ; sh:targetClass ex:CounterfactualApplication ;
    sh:sparql [ a sh:SPARQLConstraint ;
```

```
sh:message "Employment tenure increase requires age increase." ;
sh:select '''
  SELECT $this WHERE {
    $this schema:borrower ?p .
 ?p ex:originalEmploymentOrdinal ?oldEmp ; ex:newEmploymentOrdinal ?newEmp .
    FILTER(?newEmp > ?oldEmp)
    ?p foaf:age ?newAge ; ex:originalAge ?oldAge .
    FILTER(?newAge <= ?oldAge)
  }''' ] .
```

A binary immutability flag on age alone cannot catch this. The problem is the relation between the two features, which only a cross-instance rule can express.

4. Evaluation

We evaluated the SAFER pipeline across two datasets, German Credit [15] and COMPAS [16], using two counterfactual generation methods, DiCE and ActionRules. In all four dataset and generator combinations, a substantial proportion of generated counterfactuals were rejected by the SHACL validation framework. This confirms that semantic filtering is essential for producing actionable and feasible explanations.

4.1. Cost Metric

We quantify the cost of a counterfactual, x', as the sum of manually assigned weights over the features that differ from the original instance, x:

$$\text{cost}(x, x') = \sum_{i: x_i \neq x'_i} w_i,$$

where w_i denotes the cost weight of feature i. These weights encode domain intuition about the relative difficulty of altering each feature in practice. For example, we assign employment status and savings status a weight of 5.0, reflecting that they are substantially harder to change than the loan amount, which we weight at 2.0. Assigning differential costs to features to reflect their relative difficulty of change is supported in the literature. Ustun et al. [17] note that their default cost functions assume changing each feature is equally difficult, but that this assumption can be relaxed by having a domain expert specify relative difficulty, which is the approach we adopt. We adopt this metric as a practical proxy for real-world actionability. The specific weights have not been empirically validated against actual user effort or stated preferences.

4.2. German Credit with DiCE

We generated 690 counterfactual candidates for 46 rejected applicants using DiCE. After applying the SHACL constraint framework, only 262 candidates (37.97%) were fully valid. The remaining 428 candidates (62.03%) were rejected.

Table 1 compares valid and invalid counterfactuals across three metrics. Valid counterfactuals are substantially closer to the original instance and have lower cost than invalid ones. They also tend to propose longer loan durations. This suggests that realistic recourse often requires more conservative and temporally plausible adjustments rather than large feature changes.

Table 2 shows the most frequent violation types. Three rules account for the majority of rejections. The most common violation is that employment tenure cannot decrease, occurring 99 times. The second most common is that an increase in employment tenure must be accompanied by a corresponding increase in age, occurring 76 times. The third is that the loan amount was reduced by more than 70%, occurring 38 times. These violations reveal a consistent pattern: DiCE frequently proposes temporally

Table 1
Comparison of Valid and Invalid Counterfactuals (German Credit, DiCE)

Metric	**Valid** (n=262)	**Invalid** (n=428)
Distance (L1)	970.95	2353.83
Cost	21.98	24.11
Duration (months)	19.01	15.83

inconsistent changes, such as increasing employment tenure without aging the individual, or reducing the loan amount beyond a realistic threshold. At the applicant level, 45 out of 46 individuals had at least one valid counterfactual, meaning that feasible recourse was available to almost all applicants under semantic constraints.

Table 2
Most Frequent Violation Types (German Credit, DiCE)

Violation Rule	**Frequency**
Employment tenure cannot decrease	99
Employment increase requires age increase	76
Loan amount reduced by >70%	38
Checking status jump from <0 to ≥ 200 DM	23
Duration reduction exceeds 36 months	21

4.3. German Credit with ActionRules

We evaluated ActionRules on the same German Credit dataset, producing 675 counterfactual candidates. The validity rate dropped dramatically compared to DiCE. Only 52 candidates (7.70%) were valid, while 623 candidates (92.30%) were rejected.

Table 3 compares valid and invalid counterfactuals. Unlike the DiCE results, the distance and cost differences between valid and invalid counterfactuals are small. This indicates that the ActionRules violations are not driven by extreme feature changes but rather by systematic directional constraint violations.

Table 3
Comparison of Valid and Invalid Counterfactuals (German Credit, ActionRules)

Metric	**Valid** (n=52)	**Invalid** (n=623)
Distance (L1)	687.79	654.59
Cost	4.88	5.28
Duration (months)	11.77	21.87

Table 4 confirms this hypothesis. The two dominant violations are savings status cannot decrease, occurring 553 times, and employment tenure cannot decrease, occurring 429 times. ActionRules systematically proposes changes that violate these directional constraints. This contrasts with DiCE, which primarily violated cohesion and magnitude constraints. The two generators therefore fail in qualitatively different ways, a distinction that only a semantic validation framework can surface.

4.4. COMPAS

We evaluated both generators on the COMPAS dataset. ActionRules produced 932 candidates, of which 267 (28.65%) were valid. DiCE produced 495 candidates, with a higher validity rate of 271 (54.75%).

Table 5 compares valid and invalid counterfactuals for both methods. In all cases, valid counterfactuals are closer to the original instance and have lower cost than invalid ones. This is consistent with the

Table 4
Most Frequent Violation Types (German Credit, ActionRules)

Violation Rule	Frequency
Savings status cannot decrease	553
Employment tenure cannot decrease	429
Duration reduction exceeds 36 months	80
Loan amount reduced by >70%	28
Employment increase requires age increase	6

German Credit results and supports a general finding: counterfactuals that are rejected by the SHACL framework tend to be more extreme and further from the original instance.

Table 5
Comparison of Valid and Invalid Counterfactuals (COMPAS)

Method	Status	Distance	Cost
ActionRules	Invalid (n=665)	43.56	6.03
ActionRules	Valid (n=267)	34.93	5.00
DiCE	Invalid (n=224)	22.99	7.25
DiCE	Valid (n=271)	8.06	5.35

The violation patterns in Table 6 differ markedly between the two generators. ActionRules predominantly violates magnitude constraints. The most frequent violation is an unrealistic change in prior count exceeding five, occurring 317 times, followed by age changes exceeding 15 years, occurring 189 times. DiCE, by contrast, most frequently violates directional constraints. The most common DiCE violation is proposing that the defendant becomes younger, occurring 112 times. This again illustrates that the two generators fail in qualitatively different ways, and that different layers of the SHACL framework are needed to catch each type of failure.

Table 6
Most Frequent Violation Types (COMPAS)

Violation Rule	ActionRules	DiCE
Unrealistic priors change (>5)	317	32
Age change >15 years	189	5
Age cannot decrease	29	112
priors_count cannot increase	–	28
Combinations of above	89	28

4.5. Case Studies

To illustrate the practical behaviour of the pipeline, we examine two representative counterfactuals generated by DiCE on the German Credit dataset.

4.5.1. Case Study 1: A Valid Counterfactual

Applicant #136 was originally rejected. Table 7 shows the original profile alongside a DiCE generated counterfactual that successfully flips the prediction and passes all SHACL constraints.

This counterfactual changes age from 22 to 24, employment tenure from less than one year to one to four years, credit amount from 1409 to 1800 DM, and loan duration from 10 to 18 months. The checking status improves from overdrawn to no checking account, a reduction of one ordinal level and well within the permitted three level drop.

Table 7
Valid Counterfactual for Applicant #136

Feature	Original	Counterfactual
Age	22	24
Employment tenure	<1 year	1–4 years
Credit amount (DM)	1409	1800
Duration (months)	10	18
Savings status	<100 DM	<100 DM
Checking status	<0 DM	No checking

The key feature of this counterfactual is temporal coherence. The increase in employment tenure is accompanied by a two year increase in age. This reflects a realistic scenario: an applicant who was six months into a job at age 22 could plausibly have two and a half years of tenure at age 24. The loan amount and duration both increase, avoiding the reduction caps. Warnings are not triggered. This is a semantically valid and practically actionable recourse suggestion.

4.5.2. Case Study 2: An Invalid Counterfactual

Applicant #11 was originally rejected due to poor checking status and short employment tenure. Table 8 shows a DiCE generated counterfactual that flips the prediction but fails SHACL validation.

Table 8
Invalid Counterfactual for Applicant #11

Feature	Original	Counterfactual
Age	19	19
Employment tenure	<1 year	4–7 years
Credit amount (DM)	500	2611
Duration (months)	18	6
Savings status	<100 DM	<100 DM
Checking status	<0 DM	≥200 DM

This counterfactual is rejected for two violations. First, the checking status jumps from overdrawn to salary assignment level in a single step. This is an unrealistically large ordinal jump and violates the jump limit constraint. Second, employment tenure jumps from less than one year to four to seven years while age remains unchanged at 19. This is temporally impossible and violates the employment age cohesion rule.

Under a standard binary immutability model where only age is locked, this counterfactual would be considered valid because age is unchanged. Our semantic validation correctly flags it as infeasible. This demonstrates precisely the limitation of generation time checks: they cannot express the conditional relationship between employment tenure and age.

These two case studies highlight the two most important strengths of the framework. Temporal cohesion rules prevent logically impossible scenarios where employment tenure increases without the passage of time. Jump limit rules prevent unrealistic step changes in financial status. Together, they show that semantic validation provides a more honest and actionable assessment of counterfactual quality than generation time checks alone.

4.6. Discussion

Our results expose a fundamental gap between algorithmic validity and semantic feasibility. Across both datasets and both generators, a substantial proportion of counterfactuals that successfully flipped the model prediction were rejected by the SHACL validation framework. On German Credit, the rejection

rate was 92.30% for ActionRules and 62.03% for DiCE. On COMPAS, the rejection rates were 71.35% for ActionRules and 45.25% for DiCE. These figures are not marginal. They indicate that the majority of counterfactuals produced by current generators would be meaningless or harmful if presented to individuals seeking recourse.

The cost of ignoring semantic constraints. Valid counterfactuals are cheaper than invalid ones across all four dataset and generator combinations, and are closer to the original instance in three of the four combinations. This pattern is consistent and meaningful. Counterfactuals that propose extreme or unrealistic feature changes are more likely to violate semantic constraints. Proximity to the original instance is therefore not only a desirable property in its own right but also a proxy for feasibility.

Generators fail in qualitatively different ways. A key finding of this evaluation is that DiCE and ActionRules violate different types of constraints. On German Credit, DiCE primarily violates temporal cohesion and magnitude constraints. The most frequent violations involve increasing employment tenure without a corresponding increase in age, and reducing the loan amount beyond a realistic threshold. ActionRules, by contrast, systematically violates directional constraints for savings and employment status. On COMPAS, ActionRules predominantly violates magnitude constraints, proposing unrealistic changes in prior count and age. DiCE on COMPAS primarily violates directional constraints, most commonly proposing that a defendant becomes younger.

This qualitative difference between generators is a finding that no existing evaluation framework can surface. Standard metrics such as proximity, sparsity, and diversity measure properties of the counterfactual set in isolation. They cannot distinguish between a generator that proposes feasible but expensive changes and one that proposes cheap but temporally impossible changes. The SHACL validation framework makes this distinction explicit and formally grounded.

Binary immutability is insufficient. The majority of rejections across all four combinations stem from constraints that involve conditional relationships between features rather than simple immutability. The age cohesion rule alone eliminates scenarios such as a 19 year old applicant with four to seven years of employment tenure. This kind of violation cannot be prevented by marking age as immutable, because age is not immutable in the counterfactual. The violation arises from the relationship between age and employment tenure, a dependency that binary immutability flags cannot express. This confirms that richer constraint languages are needed for actionable recourse, and that SHACL-SPARQL cross-instance rules represent a principled and expressive solution.

Correctness of hard immutability enforcement. The immutability constraints for personal status, foreign worker status, and number of dependents produced zero violations across all generated counterfactuals. This confirms that both generators correctly respect hard immutability flags for these features. It also confirms that the SHACL framework does not produce false positives for these constraints, which is an important baseline for trust in the validation results.

Limitations. Several limitations of the current work should be acknowledged. The SHACL constraints and RML mappings were authored manually for the German Credit and COMPAS datasets. Extending the framework to a new domain requires domain expertise and manual effort to define the constraint layers and feature mappings. Automating this process, for example by deriving constraints from dataset metadata or existing ontologies, is an important direction for future work. The current framework provides post-hoc validation. It filters counterfactuals after generation but does not guide the generator toward feasible solutions. Future work can involve proposing an approach that provides validation during the process of counterfactual explanation generation. The cost model used in this work is heuristic and has not been empirically calibrated against real user preferences. User studies are needed to validate whether the cost metric reflects the actual effort required for individuals to achieve recourse. Finally, the SHACL constraints encode observed correlations between features rather than causal relationships. Integrating causal graphs into the constraint framework would enable richer and more principled feasibility checks. The evaluation is limited to two datasets and two generators, and broader validation is needed to establish the generality of the findings.

5. Conclusion

Counterfactual explanations are the dominant mechanism for algorithmic recourse, yet existing generators optimise for proximity and diversity while ignoring the temporal and causal dependencies that determine true actionability. A counterfactual that flips a model prediction is not necessarily a counterfactual that a real individual can act upon. We argue that semantic validation is not an optional post-processing step but an essential component of any trustworthy recourse pipeline.

We introduced SAFER, a post-hoc validation pipeline that bridges Explainable AI and Semantic Web technologies. Tabular counterfactuals are lifted into RDF knowledge graphs via declarative RML mappings and validated against a four-layer SHACL constraint framework. The framework enforces data integrity, hard immutability, feasibility constraints including temporal cohesion and jump limits, and counterfactual quality checks. The SHACL-SPARQL cross-instance rules at the heart of the framework can express conditional and relational constraints that no generation time check can enforce.

The evaluation results are striking. On German Credit, DiCE produced 62.03% invalid counterfactuals and ActionRules produced 92.30% invalid counterfactuals. On COMPAS, the invalid rates were 45.25% for DiCE and 71.35% for ActionRules. These are not edge cases. The majority of counterfactuals produced by current generators would be meaningless or misleading if presented to individuals seeking recourse. The temporal cohesion rules alone eliminate scenarios such as a 19 year old applicant with four to seven years of employment tenure. The jump limit rules prevent unrealistic step changes in financial status that no individual could plausibly achieve.

A further finding is that the two generators fail in qualitatively different ways. ActionRules predominantly violates magnitude and minimality constraints while DiCE primarily violates directional and temporal constraints. This distinction is invisible to standard evaluation metrics such as proximity and sparsity. It only becomes apparent through semantic validation. This suggests that SAFER can serve not only as a filter for infeasible counterfactuals but also as a diagnostic tool for understanding the systematic failure modes of individual generators.

Since the framework is fully decoupled from the generator, it is both method-agnostic and extensible. Domain experts can introduce new constraints without modifying any generative model, and the same pipeline applies uniformly across generators with different internal assumptions. The structured and machine-readable SHACL validation reports provide a formal basis for cross-generator comparison and for audit trails suited to emerging regulatory requirements such as those introduced by the EU AI Act.

Declaration of the use of Generative AI

We used Generative AI to help polish sentences for clarity. All technical content, implementation, validation, and analysis were developed and verified by the authors, who take full responsibility for the work.

References

[1] S. Wachter, B. Mittelstadt, C. Russell, Counterfactual explanations without opening the black box: Automated decisions and the GDPR, SSRN Electronic Journal (2017). URL: https://ssrn.com/abstract=3063289. doi:10.2139/ssrn.3063289.

[2] S. Upadhyay, S. Joshi, H. Lakkaraju, Towards robust and reliable algorithmic recourse, arXiv abs/2102.12646 (2021). URL: https://arxiv.org/abs/2102.12646. doi:10.48550/arXiv.2102.12646.

[3] A.-H. Karimi, G. Barthe, B. Schölkopf, I. Valera, A survey of algorithmic recourse: Definitions, formulations, and solutions, ACM Computing Surveys 55 (2022) 1–36. URL: https://dl.acm.org/doi/10.1145/3527848. doi:10.1145/3527848.

[4] R. K. Mothilal, A. Sharma, C. Tan, DiCE: Explaining machine learning classifiers through diverse

counterfactual explanations, arXiv abs/1905.07697 (2019). URL: https://arxiv.org/abs/1905.07697. doi:10.48550/arXiv.1905.07697.

[5] P. Garg, L. Nagalapatti, S. Sarawagi, From search to sampling: Generative models for robust algorithmic recourse, in: International Conference on Learning Representations (ICLR), 2025. URL: https://proceedings.iclr.cc/paper_files/paper/2025/hash/117308c91189385fd47cfc36d7e6e0e5-Abstract-Conference.html.

[6] B. Turbal, I. Voitsitska, L. Semenova, ElliCE: Efficient and provably robust algorithmic recourse via the rashomon sets, in: Advances in Neural Information Processing Systems (NeurIPS), 2025. The Thirty-ninth Annual Conference on Neural Information Processing Systems.

[7] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artificial Intelligence 267 (2019) 1–38. URL: https://www.sciencedirect.com/science/article/pii/S0004370218305988. doi:10.1016/j.artint.2018.07.007.

[8] S. Verma, V. Boonsanong, M. Hoang, K. Hines, J. Dickerson, C. Shah, Counterfactual explanations and algorithmic recourses for machine learning: A review, arXiv abs/2010.10596 (2020). URL: https://arxiv.org/abs/2010.10596. doi:10.48550/arXiv.2010.10596.

[9] L. Sýkora, T. Kliegr, action-rules: Gpu-accelerated python package for counterfactual explanations and recommendations, SoftwareX 29 (2025) 102000. URL: https://www.sciencedirect.com/science/article/pii/S2352711024003704. doi:10.1016/j.softx.2024.102000.

[10] H. Knublauch, D. Kontokostas, Shapes Constraint Language (SHACL), W3C Recommendation, World Wide Web Consortium (W3C), 2017. URL: https://www.w3.org/TR/shacl/.

[11] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, T. Pellissier, J. F. Sequeda, S. Staab, A. Zimmermann, Knowledge Graphs, Synthesis Lectures on Data, Semantics, and Knowledge, Morgan & Claypool, 2021. doi:10.2200/S01125ED1V01Y202109DSK022.

[12] A. Klironomos, B. Zhou, Z. Tan, Z. Zheng, M. H. Gad-Elrab, H. Paulheim, E. Kharlamov, ExeKGLib: A platform for machine learning analytics based on knowledge graphs, in: Proceedings of the 24th International Semantic Web Conference (ISWC 2025), Springer, Cham, 2026, pp. 57–76. doi:10.1007/978-3-032-09530-5_4.

[13] C. Cortés, L. Ehrlinger, L. Etcheverry, F. Naumann, Is SHACL suitable for data quality assessment?, 2025. URL: https://arxiv.org/abs/2507.22305. arXiv:2507.22305.

[14] A. Dimou, M. V. Sande, P. Colpaert, R. Verborgh, E. Mannens, R. V. de Walle, RML: A generic language for integrated RDF mappings of heterogeneous data, in: Proceedings of the 7th Workshop on Linked Data on the Web (LDOW), 2014.

[15] H. Hofmann, Statlog (german credit data), UCI Machine Learning Repository, 1994. URL: https://doi.org/10.24432/C5NC77.

[16] ProPublica, Compas recidivism risk score data, 2016. URL: https://github.com/propublica/compas-analysis.

[17] B. Ustun, A. Spangher, Y. Liu, Actionable recourse in linear classification, in: Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT* '19), ACM, New York, NY, USA, 2019, pp. 10–19. doi:10.1145/3287560.3287566.