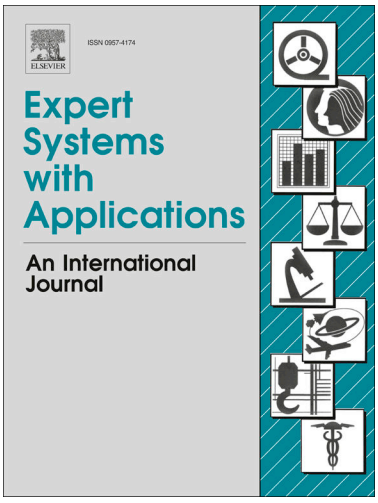
Journal Pre-proofs



Feature-Weighted Counterfactual-based Explanation for Bankruptcy Prediction

Soo Hyun Cho, Kyung-shik Shin

PII: S0957-4174(22)02408-3
DOI: https://doi.org/10.1016/j.eswa.2022.119390
Reference: ESWA 119390

To appear in: *Expert Systems with Applications*

Received Date: 2 September 2021
Revised Date: 8 July 2022
Accepted Date: 29 November 2022

Please cite this article as: Hyun Cho, S., Shin, K-s., Feature-Weighted Counterfactual-based Explanation for Bankruptcy Prediction, *Expert Systems with Applications* (2022), doi: https://doi.org/10.1016/j.eswa.2022.119390

Feature-Weighted Counterfactual-based Explanation for Bankruptcy Prediction

Soo Hyun Cho
Department of Big Data Analytics,
Ewha Womans University, 52 Ewhayeodae-gil,
Seodaemun-gu, Seoul 03760, South Korea
soohyuncho7117@gmail.com

Kyung-shik Shin
School of Business,
Ewha Womans University, 52 Ewhayeodae-gil,
Seodaemun-gu, Seoul 03760, South Korea
ksshin@ewha.ac.kr

Abstract

In recent years, there have been many studies on the application and implementation of machine learning techniques in the financial domain. Implementation of such state-of-the-art models inevitably requires interpretability for users to understand the result and trust. However, as most of the machine learning methods are "black-box," explainable AI, which aims to provide explanations to users, has become an important research issue. This paper focuses on explanation by counterfactual example for a bankruptcy-prediction model. Counterfactual-based explanation offers an alternative case for users in order for them to have a desired output from the model. This paper proposes a genetic algorithm(GA)-based counterfactual generation algorithm using feature importance whilst taking other key factors into account. Feature importance was derived from a prediction model, and key factors for counterfactuals include closeness to the original dataset and sparsity. The proposed method presented advantages over the nearest contrastive sample and a simple counterfactual generation algorithm in the experiment. Also, it provides relevant and compact explanations to enhance the interpretability of the bankruptcy prediction model.

Keywords: Bankruptcy prediction, Counterfactual-based explanation, Explainable artificial intelligence

1. Introduction

In the domain of finance, risk management tools such as bankruptcy prediction, credit scoring model and stock market prediction are critical. Failure to manage these risks will certainly cause damages to the company and, furthermore, to the related parties. With the development of current artificial intelligence techniques, numerous studies have utilized such techniques for financial prediction models (Alaka et al., 2018; He, Zhang, & Zhang, 2018; Ngai, Hu, Wong, Chen, & Sun, 2011; Son, Hyun, Phan, & Hwang, 2019). Despite their impressive performance, it is difficult to implement machine learning-based models due to their intrinsic trait of obscurity, especially when the field requires or values an explanation about the result obtained by the model, as is the case in the field of finance. Obscurity, or the model's being a "black-box," means that there is no explanation given to the user about the prediction made by the model. This well-known property of a machine learning technique's being a "black-box" has now become an interesting research topic called "Explainable Artificial Intelligence" (XAI).

Among other approaches in explanation generation, a counterfactual explanation is a type of an example-based explanation in terms of explanation format. A number of studies were carried out to provide an explanation on prediction models using counterfactuals. Counterfactual can be defined as a nearby instance of an actual observation but with a different prediction result from the model. Most of the studies on counterfactual example-based explanations focused on increasing the quality of the counterfactuals and paid little attention to the link between the counterfactuals and the prediction model. Past studies focused on increasing the diversity or the feasibility (or plausibility) of the generated counterfactual-based explanations. In Rodriguez et al. (2022), the proposed model utilizes a diversity-enforcing loss to discover diverse explanations when learning perturbations in a disentangled latent space. Mahajan, Tan, and Sharma (2019) proposed a method to offer feasible and actionable explanations by generating explanation that follows the underlying data distribution of the original data, and Poyiadzi, Sokol, Santos-rodriguez, Bie, and Flach (2020) adopted user labeling whether the generated explanation is feasible or not. In Kanamori, Takagi, Kobayashi, and Arimura (2020), the authors tried to generate explanations that fall under the distribution of the dataset by determining whether the counterfactual example is an outlier or not.

Other studies focused on increasing the relevance of the counterfactual explanation by taking feature importance into account. As the counterfactual example shows the feature changes required to have an opposite result from the model, not only the amount of the changes but also which features should change matters. In Le, Wang, and Lee (2020), an entropy-based forward feature ranking was used to obtain feature importance for each case. LEAFAGE trains a local linear surrogate model near the instance and uses the coefficients of the model as a measure of local feature importance (Adhikari, Tax, Satta, & Faeth, 2019). In Grath et al., (2018), the authors implemented two methods to include feature importance such as ANOVA for global feature importance and weight vectors by aggregating the relative changes of the k-nearest neighbors.

Although there have been some approaches considering feature importance in counterfactual explanation generation, most of the studies neglected to consider global feature importance in a counterfactual generation or used a model-independent statistical method such as ANOVA to obtain global feature importance.

This work introduces a counterfactual generation algorithm for a bankruptcy prediction model using a feature-weighted distance and multi-objective approach to produce a counterfactual example. We believe it is important to provide an explanation that is not only easy to understand but also it should hold the connection to the model it is explaining. The proposed method weighs important features discovered by the SHAP method in the black-box model when measuring the distance between the original sample and the contrastive sample. By incorporating the feature importance derived from the model into the explanation generation process, more relevant features can be used to offer the explanation. At the same time, instead of focusing on a specific trait of the counterfactual, we considered multiple traits that affect the quality of the explanation. Therefore, the key factors of counterfactual explanations, validity, proximity, and sparsity (Mothilal et al., 2020; Russell, 2019), were considered in the generation process. The proposed method can find counterfactuals instances with satisfactory validity and sparsity to provide the user with more compact and feature importance-based counterfactuals that are easier to understand for users.

In this article, we approach an example-based explanation by generating a counterfactual for a bankruptcy prediction model. Specifically, a feature-weighted counterfactual example for each observation will be generated using a genetic algorithm (GA) with multiple objectives. The proposed method aims to find proximal counterfactuals satisfying multiple objectives, including closeness to the original data with relevant features, the lowest possible number of changed features, and a generated sample that is close to the original dataset but is not an outlier albeit has an opposite output from the original instance.

The remainder of the paper is organized as follows: Section 2 covers related works on interpretable financial-prediction models and counterfactual-based explanation approaches for machine learning models. Section 3 introduces the proposed method for counterfactual generation, and Section 4 describes the data and experiment on the proposed method and discusses the results. Concluding remarks are given in Section 5.

2. Literature Review

2.1 Interpretability in Machine Learning

Technically speaking, there is no standard definition of the XAI (Berrada, Adadi, & Berrada, 2018). Also, the terms “interpretability” or “explainability” are used interchangeably in the field (Berrada et al., 2018; Carvalho, Pereira, & Cardoso, 2019). XAI is more of a trend and movement towards AI transparency and trust issues.

Beyond using the interpretable model for explainability, a hybrid approach utilizing symbolic and sub-symbolic representation together to achieve interpretability are under rigorous research. Sub-symbolic methods such as deep learning or ensemble models are inherently difficult to acquire explainability. Some studies proved the capability of learning and reasoning together using both symbolic and sub-symbolic representations and inferences (Barredo Arrieta et al., 2020). For example, such hybrid approaches include the use of the transparent model in twin with opaque models such as neural networks to obtain interpretability. This adds interpretability to the non-transparent model. For example, in Kenny and Keane (2021) and Kenny and Keane (2019), case-based reasoning was used to find neighbor instances by mapping feature weights to explain different types of ANN models.

Furthermore, the attention mechanism, in addition to improving the model performance, can also be utilized to explain perplexing neural architectural behavior (Niu, Zhong, & Yu, 2021) Attention mechanism distributes attention weights in the neural network model, yielding high attention weights to the relevant features by allowing the decoder of the network to use relevant parts from the input features. It is often used in natural language processing problems to represent feature importance (i.e., words or tokens), but recently attention mechanism has been adopted in other fields such as healthcare. For example, Kwon et al. (2019) developed an interactive temporal attention model called RetainEx to show feature contribution scores using an attention mechanism related to the model’s decision using medical treatment data.

The scope of interpretability in machine learning can be categorized into global and local interpretability. Global interpretability relates to how the parts of the model affect its predictions to understand the model at the modular level. Interpretability at the global level usually includes intrinsically transparent models such as decision trees, Naïve Bayes classifier, and k-nearest neighbor classifier as the trained model can be explained and interpreted. Local interpretability relates to why the model had a certain prediction in a particular instance, explaining the model's prediction at the instance level. There are several approaches to local interpretation, such as LIME (Ribeiro & Guestrin, 2016) and SHAP (Lundberg & Lee, 2017). LIME uses feature perturbation to generate local explanation, and SHAP measures the marginal contribution of the features to generate local interpretation, which can later offer global interpretation using collective values as well.

Depending on when the explanation operation occurs, the interpretability can be categorized as either ad-hoc or post-hoc. In general, ad-hoc methods consider generating explanations from the very beginning of the training. These methods are typically applied to transparent models like fuzzy models and tree-based models to produce explanations (Islam, Ahmed, Barua, & Begum, 2022). Post-hoc explanations on specific instances do not focus on model functions but on offering human-interpretable explanations on how and why the model behaved in a certain way (Mittelstadt, Russell, & Wachter, 2019). Most counterfactual-based explanation methods have also been applied post-hoc. As post-hoc explanations on specific instances do not focus on model functions but on offering human interpretable explanations on how and why the model behaved in a certain way, counterfactual example-based explanation attempts to deliver it by making the explanation as human interpretable as possible.

Based on the format of the explanation, interpretation approaches can be distinguished as well. For example, LIME and SHAP provide feature importance-based explanations. In Shimizu, Matsutani, and Goto (2022), an algorithm to offer explanations for recommendation systems using a graph attention network was introduced. There are numerous variables in the model, and to utilize side information for explanation and preserve the model accuracy, the authors used the knowledge graph attention network model. The proposed model can offer a behavior-based and attribute-based interpretation of the recommendation result with the attention weights of the network. Local rule-based models (Guidotti et al., 2019; Rajapaksha, Bergmeir, & Buntine, 2020) generate counterfactual rules for each case, assuming that local decision boundaries can be understood more easily and local instances behave similarly in the model. Anchor (Ribeiro, Singh, & Guestrin, 2018) and LORE (Guidotti et al., 2019) offer local interpretations by generating local rules for instances. Anchor provides if-then rules by finding, "anchors" representing feature conditions for instances where the prediction result stays the same. LORE generates rules using a synthetic local neighborhood and applying decision tree model to extract rules. Another format is an example-based explanation approach using an example to explain the model prediction, such as counterfactual or factual examples. Rule-based and example-based explanation approaches can deliver via contrastive explanation. Meaning that the explanatory rules and examples can counterfactually explain the decision (Waa et al. (2021). However, the two approaches differ as rule-based explanation extracts rules to explain the model, whereas example-based explanation uses examples to explain the model. This study also tries to provide an explanation at the instance level by producing counterfactuals. A representative type of example-based explanation is a counterfactual-based explanation, which is further explained in Section 2.3.

2.2 Interpretability in Financial Prediction Models

There have been many studies on improving the performance of bankruptcy prediction and credit scoring models (Du Jardin, 2016; Feng, Xiao, Zhong, Qiu, & Dong, 2018; He et al., 2018; Marqués, García, & Sánchez, 2012; Moscatelli, Parlapiano, Narizzano, & Viggiano, 2020). However, compared to studies focused on the performance of the prediction models, only a few studies have focused on the interpretability problem of the machine learning-based models (Dastile, Celik, & Potsane, 2020). However, a bankruptcy-prediction model involves an important type of financial prediction that requires both high accuracy and interpretability. Without any explanation on decisions such as bankruptcy prediction or credit scoring, it is very difficult to implement the system in real life. This aspect of the financial domain has interested some researchers in investigating the interpretability of financial-risk-prediction models.

For most interpretable credit risk management models, rule-based or example-based models have been extensively studied. Setiono and Liu (1996) proposed an automatic rule extraction method for a neural network using its symbolic representation called NeuroRule. To extract rules from a neural network model, the importance of the network's connections is reflected by weight decay in the backpropagation network. After removing remove irrelevant

connections and units, it discretizes the hidden unit activation values using a clustering technique to extract rules with discretized hidden unit activation values. Hayashi (2016) used a recursive rule extraction algorithm with a decision tree to extract interpretable rules for machine learning-based credit scoring models. The proposed rule extraction method can provide hierarchical and recursive consideration of discrete variables for continuous data and generate rules from a neural network model trained with both discrete and continuous features. Dong et al. (2021) proposed a two-stage rule-extraction method for a tree ensemble model with a local and a global stage. The local method simplifies each rule by removing its redundant constraints (Stage 1), while the global method optimizes the complete rule set based on the multi-objective optimization method (Stage 2). Other studies have approached rule generation for interpretability as an optimization problem (Soui, Gasmi, Smiti, & Ghédira, 2019). Using an evolutionary algorithm (EA), the study aimed to find the best combination of customer characteristics to generate appropriate classification rules.

For an example-based approach toward interpretability of the financial risk prediction model, Henley and Hand (1996) proposed a simple *k*-nearest neighbor for the credit scoring model. This method can easily provide information on why the model made a specific prediction using neighbors as examples. Similarly, but more recently, Grath et al. (2018) used another example-based approach using counterfactuals. The study implemented negative and positive counterfactuals as explanations for loan applicants and proposed two weighing strategies (i.e., feature importance and nearest neighbor) to generate more interpretable counterfactuals.

2.3. Counterfactual example-based explanation

A counterfactual explanation is an effective type of example-based explanation that offers changes needed to produce a different outcome from the model for a specific instance. In other words, it offers a local interpretation of the result obtained from the model by providing the user with a counterfactual instance. Counterfactual examples are supposed to be similar to the original dataset with only minor changes in features. This concept of the "closest possible world" is a key notion throughout the discussion of counterfactuals (Wachter, Mittelstadt, & Russell, 2018). The method itself is intuitive and similar to the human-thinking process (Mittelstadt et al., 2019) because people tend to ask why one decision is made instead of another (Miller, 2019), imagining how a decision could have been different if conditions were different (Byrne, 2019). This makes counterfactual explanation an attractive explanation option, as people are interested in factors that need to be changed to flip the decision. Recently, there has been active research on this topic. Some studies have dealt with image data for the counterfactual-based explanation (Goyal et al., 2019; Kenny, Ford, Quinn, & Keane, 2021; Mahajan et al., 2019; Poyiadzi et al., 2020), while other studies used counterfactual-based explanation for tabular data, as this paper (Guidotti et al., 2019; Hashemi & Fathi, 2020; Kanamori et al., 2020; Le et al., 2020).

In Kenny and Keane (2021b) the authors proposed a model for generating plausible counterfactuals and semi-factuals for the CNN model using image data. The study models the distribution of the latent features to detect and modify "exceptional" features to "normal" to increase the plausibility of the generated explanation. The model uses GAN on CNN classifier to generate counterfactual and identifies the exceptional features by examining their statistical probabilities in the training distributions of the desired class and then modifying them. In addition, a twin-system using a black-box deep learning model and white-box case-based reasoning (CBR) method was introduced to offer an example-based explanation (Kenny & Keane, 2021a). The study applied several feature weighting global and local methods on *k*-nearest neighbors to find the instance.

In Tsirtsis, De, and Gomez-Rodriguez (2021), the authors proposed a counterfactual generation algorithm for sequential decision-making problems, unlike other studies focusing on the counterfactual explanation for the non-sequential problem. A polynomial time algorithm was proposed to find a sequence of actions for a counterfactual generation. To find the counterfactuals, dynamic programming was applied.

Some studies focused on obtaining counterfactuals with feature importance (Grath et al., 2018; Keane & Smyth, 2020; Le et al., 2020). When generating counterfactuals, it is rather intuitive to expect changing features in the counterfactuals to be more relevant (Belkoura, Zanin, & Latorre, 2019). GRACE (Le et al., 2020) used an entropy-based forward feature ranking method for each instance to prioritize the features when generating contrastive samples. Two weighting methods that consider feature importance when generating counterfactuals by incorporating weight vector into the distance metric (Grath et al., 2018). Dandl et al. (2020) proposed to measure the feature importance for

each instance using the standard deviation of the individual conditional expectation curve and to promote important features. They used ANOVA and *k*-nearest neighbors to extract global and local feature importance to generate more compact counterfactuals.

In this study, SHAP-derived feature importance was used to incorporate feature importance into counterfactual generation. Unlike other statistical or heuristic-based methods to determine feature importance, SHAP-derived feature importance can give information on which features were found to be influential in a "black-box" model. At the same time, other criteria of the counterfactuals such as sparsity and feasibility are considered by formulating the generation algorithm as a multi-objective optimization problem using GA. paper aims to offer an explanation in counterfactual example format for a bankruptcy prediction model with a weighted feature and multi-objective approach for a counterfactual generation.

3. Proposed Model

Counterfactual generation can be framed as an optimization problem to find a solution (i.e., counterfactual sample) in the feature space by searching for perturbations that approximate the original but lead to a different result from the machine learning model (Mahajan et al., 2019; Wachter et al., 2018). The proposed method generates a counterfactual sample for each observation after training the "black-box" prediction model. In this paper, counterfactual samples for all instances, meaning for both positive (bankrupt) and negative (non-bankrupt) cases, were generated. Although most studies have focused on generating counterfactuals to explain how to obtain a desirable result, counterfactuals on negative cases can also ensure that people feel relieved and justified (Byrne, 2019). For the prediction model, an artificial neural network (ANN) and a support vector machine (SVM) were implemented as it is one of the commonly used ML-based bankruptcy prediction models with high accuracy (Alaka et al., 2018). In the proposed method, after training the model, feature-weighted counterfactuals were generated using GA with multiple objectives in the fitness function. As the method utilizes GA after training the prediction model, the proposed method can be applied model-agonistically. The brief flowchart of the method is presented in Figure 1.

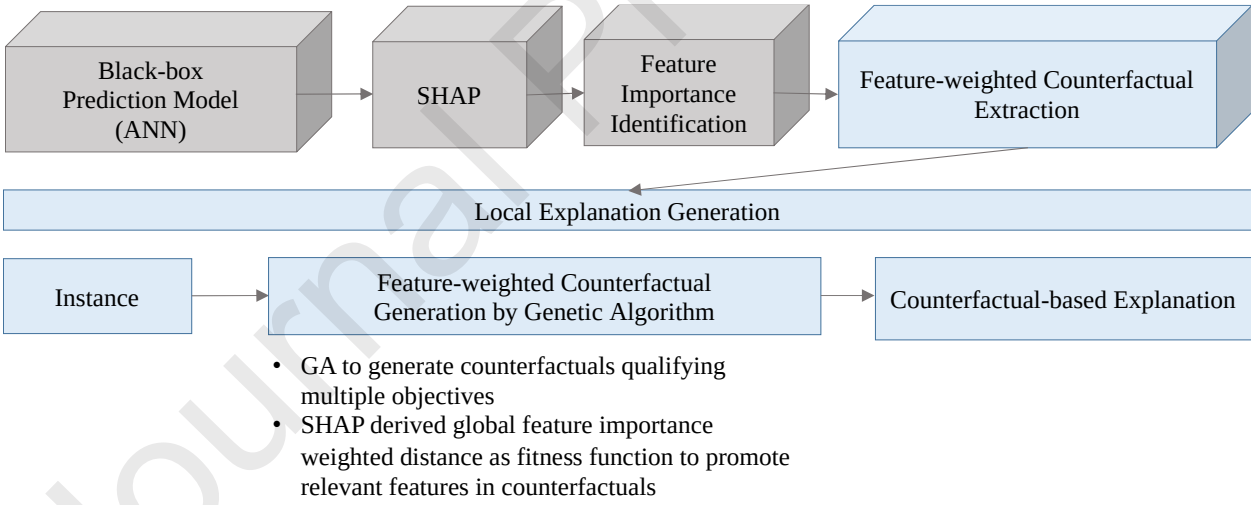


Figure 1 Proposed model flowchart

The main reason why we propose to generate feature-weighted counterfactuals with multiple objectives is that there are multiple trade-off criteria to consider in order to generate "good" counterfactuals, which is a key to enhancing the comprehensibility of the generated counterfactuals. First, we believe that it is important to promote important features when generating counterfactuals. Second, the number of criteria, such as the data distribution of the generated sample and the number of features changed, should be considered. Intuitively, it is more reasonable and understandable to provide a counterfactual explanation with relevant and important features than irrelevant ones. Numerous features are used in the model, and it makes less sense to explain a result to the user using unimportant features than using important features. For this reason, feature importance derived from the model using SHAP was used as feature weight in the

proposed method. SHAP is a model-agonistic explanation method based on Shapley values and game theory. The SHAP method computes the marginal contribution of the features in the model. Also, this method can be applied to any machine learning model, making it possible to find model-relevant features, as all models have different weights on features. In this paper, the SHAP method was applied to a trained black-box model to identify its important features. The absolute mean of the SHAP values was used as feature importance. At the same time, however, it is necessary to consider other qualities in counterfactuals. For example, sparsity and closeness to the original sample are desirable properties of a counterfactual (Fernández, Martín de Diego, Aceña, Fernández-Isabel, & Moguerza, 2020). Proximity, plausibility, and sparsity are the key elements in determining a good counterfactual (Kenny et al., 2021). Therefore, the proposed method employs multiple terms in the fitness function to consider various objectives in trade-off relations. Such objectives include numerous features changed in the counterfactual and realisticness of the generated counterfactual.

The optimization problem for counterfactual generation mostly consists of two terms: loss and distance. In Wachter et al. (2018), the authors suggested finding counterfactual samples considering loss from the prediction model and median absolute deviation (MAD) weighted Manhattan distance. Here, this paper aims to find feature-weighted counterfactuals satisfying multiple objectives to be a "good" counterfactual. Eq. (1) shows the fitness function used to find counterfactuals using GA, each representing different qualities required for counterfactuals. In the GA, the vector of features x represents an instance from a test set while $\hat{x}$ represents a possible counterfactual candidate. The GA compares the population with the instance to be explained (x) in terms of feature values and the model output based on the fitness function, and as the purpose of the model is to find the chromosome (i.e., a counterfactual or $\hat{x}$) that yields the opposite model output (i.e., presented as $f(\hat{x}) \neq f(x)$ in Eq.(3).) The role of the constraint is to ensure the validity of the generated sample, is to have the opposite output from the model compared to the original instance.

Eq. (1) is the objective function used for the counterfactual generation, and the objective function is subject to the case in which the output of the counterfactual is the opposite of the original instance. The fitness function was formulated as a minimization problem, and $\lambda(0 < \lambda)$ was used as a balancing parameter between the objective terms. In Eq. (1), $f(\cdot)$ is the trained prediction model, x being the original instance, $\hat{x}$ being a generated counterfactual, and $f(x)$ being the output of $f(\cdot)$ with instance x. It aims to find a solution that minimizes the distance between x (i.e., an instance to be explained) and $\hat{x}$ (i.e. a counterfactual explanation generated from GA), sparsity of the $\hat{x}$ and LOF of the $\hat{x}$ within the dataset. Each term in the fitness function represents some of the key factors for generating a counterfactual explanation such as distance, sparsity, and feasibility.

$$\arg min\, dist(x, \hat{x})\lambda_1 + lof(\hat{x})\lambda_2 + spr(\hat{x})\lambda_3,\ subject\ to\ f(\hat{x}) \neq f(x) \tag{1}$$

The first term pertains to calculating the weighted distance to induce changes of important features. To measure a distance between the original instance and a counterfactual, the weighted Euclidean distance was used, as shown Eq. (2) and Eq. (3). We incorporated the feature importance of the model into the distance measure to generate more relevant counterfactual examples as counterfactuals that lacks the link to the model are insufficient as an explanation. As important as it is to generate explanations in an understandable format, it should be related to the prediction model as well. Therefore, feature importance was considered in the proposed model using SHAP values.

SHAP aims to explain the prediction at the instance level by computing the contribution of each feature in the model. The SHAP explanation method obtains Shapley values using coalitional game theory. The feature values of an instance act as players in a coalition and Shapley values indicate how the "payout", meaning the output of the model, among the features can be distributed. For tabular data, a player indicates a feature value. One of the important points of the SHAP method is that the method is represented as an additive feature attribution method, a linear model. From the SHAP method, an explanation function can be computed as Eq. (2).

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i \quad (2)$$

From Eq. (2), g is a surrogate explanation model, $z' \in \{0,1\}^M$ is the coalition vector, M is the maximum coalition size (i.e., number of features in original instance x) and $\phi_i \in \mathbb{R}$ is the feature attribution for feature i, the SHAP value. The coalition vector represents either presence or absence of each feature, 1 indicating presence and 0 indicating absence. Assuming that $f(x)$ is the prediction from the original model, the explanation model $g(z')$ approximates $f(x)$ by summing the feature attributions. These feature attributions, called the SHAP value, can be calculated as Eq. (3). The SHAP value of feature i of instance x in the model $f(\cdot)$ is denoted as $\phi_i(f, x)$. M is the maximum coalition size (i.e., number of features in original instance x) and $|z|$ is the number of non-zero entries in input x. As the equation shows, it uses the average of all possible permutations of the features, and to simply put, the contribution of the feature is obtained by subtracting the contribution of the features excluding the ith feature from the total contribution. Calculating SHAP values is time consuming as it requires permutations of all the features. In this experiment, a Python library SHAP was used.

$$\phi_i(f, x) = \sum\nolimits_{z \subseteq x} \frac{|z|!\,(M - |z| - 1)!}{M!} [f(z) - f(z \backslash i)] \quad (3)$$

For feature importance, min-max scaled mean of absolute SHAP values was used. In the classification problem, SHAP values are ranged between -1 and 1, and the value can be either negative or positive depending on the impact of the feature on the output. Meaning that if the feature has a negative impact on the output of the model, the value will be negative, and the magnitude of the impact will be the absolute value. In this paper, only the magnitude of the impact is considered as feature importance in the model. Therefore, regardless of the direction of the impact, whether it is negative or positive, only the magnitude of the feature importance was considered by taking the absolute SHAP value of the ith feature as in Eq. (4). Also, to inversely weigh the distance measure, the feature importance of the feature i subtracted from its maximum value 1 and then 0.5 was added to ensure all the values stay positive as shown in Eq. (5). In Eq. (6), w_i is the ith weight vector created from feature importance, while j indicates the total number of features. Note that the weight vector w was created to promote important features by weighting features inversely when calculating weighted distance, and the addition of 0.5 was to ensure all the weight vectors stay positive.

$$feature\ importance_i = |SHAP\ value|_i \quad (4)$$

$$w_i = (1 - feature\ importance_i) + 0.5 \quad (5)$$

$$dist(x, \hat{x}) = \sum_{i=1}^{j} \sqrt{w_i (x_i - \hat{x}_i)^2} \quad (6)$$

The second term of the objective function represents another factor to consider when generating counterfactuals is whether the generated sample is close to the original dataset. Actionability and realisticness of counterfactuals can be achieved by grounding them in the input dataset distribution (Verma, Dickerson, & Hines, 2020). This leads to the second term, taking the feasibility of the counterfactual into consideration by minimizing the LOF score of the given instance, first suggested in Breunig, Kriegel, Ng, and Sander, (2000). It is important to have the counterfactuals within the feasibility, meaning that the counterfactual should fairly represent the distribution of the real dataset. Otherwise, the counterfactual will just be an outlier or a case that is impossible in real life. LOF was used in this study because

unlike conventional outlier detection methods that simply label an instance in terms of whether it is an outlier in a given dataset, LOF shows the degree of the instance being an outlier in its local area. Additionally, one of the use cases of a counterfactual is to detect outliers and so-called bugs, but since the counterfactual in this paper aims to provide an actionable, feasible explanation, the generated counterfactuals should be close to the distribution of the original dataset. LOF calculates its score based on local density, and the locality is defined by a k neighbors. DACE utilizes LOF to produce distribution-aware counterfactuals as well by comparing local characteristics of its single nearest neighbor. However, comparing the LOF with its single nearest neighbor delivers less accurate information on whether the new instance is truly an outlier. To consider broader local traits existing in the dataset, k is set to 20 in this study. To obtain the LOF as in Eq. (9), reachability distance as in Eq. (7) and local reachability density as in Eq. (8) were calculated. Usually, inliers have a LOF near 1(LOF ~1), and instances have higher chances of being an outlier when they have a LOF of higher than 1 because they would have a lower density than their neighbors.

$$rd_k(o,p) = \max\{k-distance(o), d(o,p)\}, \quad k-distance = mean\ distance\ of\ p\ and\ its\ k-closest\ neighbors \tag{7}$$

$$lrd_k(o) = \frac{|N_k(o)|}{\sum_{o \in N_k(p)} rd_k(o,p)}, \quad N_k(o) = number\ of\ instances\ near\ datapoint\ o\ within\ k-distance \tag{8}$$

$$lof_k(o) = \frac{\sum_{o \in N_k(p)} \frac{lrd_k(p)}{lrd_k(o)}}{|N_k(o)|} \tag{9}$$

The third objective, as reflected in Eq. (10), concerns the number of features changed. The fewer the features are changed, the more concise and easier the counterfactual becomes. Also, it becomes a more feasible and actionable explanation if fewer features are changed.

$$spr(\hat{x}) = number\ of\ features\ changed \tag{10}$$

To find counterfactuals, GA was used. GA is a kind of evolutionary algorithm widely used for generating near-optimal solutions for optimization problems. GA has some advantages compared to other optimization techniques. It uses both an exploratory and exhaustive approach to find a solution, and it finds a near-optimal solution while avoiding falling to local minima. GA can also handle the multi-objective optimization problem. GA is a gradient-free method that makes the proposed approach applicable to any model, which makes the proposed model model-agonistic approach. LORE also uses a GA to generate the local neighborhood of the instance, which is later used to make a set of rules that can explain the model's local prediction. Similarly, in Dandl et al. (2020), the authors used a type of evolutionary algorithm called the Nondominated Sorting Genetic Algorithm II (NSGA-II) as a counterfactual generation algorithm. To incorporate feature importance into the explanation, we used SHAP values to weigh important features in the process of generating an explanation using GA that can be applied model agonistically.

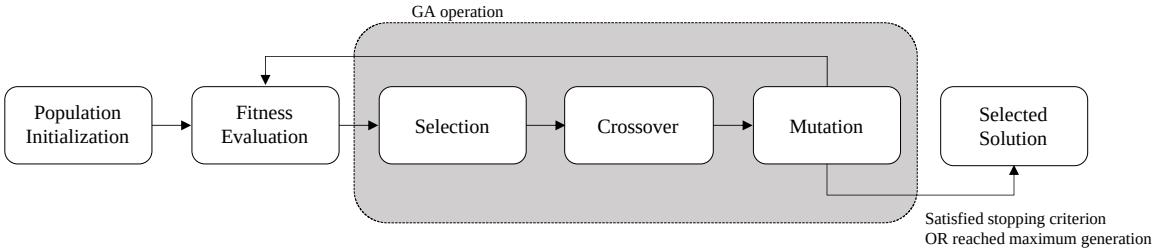


Figure 2 GA flowchart

To briefly explain the process of GA, it starts with a set of chromosomes containing a certain number of genes. GA operates on the population of the set in an "evolutionary" way to search for the best chromosome. Each chromosome represents a solution, and therefore a set of chromosomes—a population—indicates a set of possible solutions. In this case, each chromosome indicates a possible counterfactual candidate. The solutions are evaluated based on the fitness function declared above**.** GA operation consists of selection, mutation, and crossover to generate generations of population, which makes the algorithm search using both the exploitation and exploration methods. Figure 2 demonstrates how GA works.

In the population initialization phase, GA generates an initial population to begin a search. The initial population is generated by randomly making a certain number of chromosomes consisting of a certain number of genes declared before the GA operation. A large number of populations introduces more diversity by enlarging the search space, but it converges slowly, whereas small number of populations converges faster but the search space of the problem may not be enough to find the near-optimal solution. Figure 3 shows the encoding scheme of the chromosome in the experiment. Each gene denotes a counterfactual's feature value and is designed to have a real value between 0 and 1. The genes of the chromosome are the feature values of the counterfactual example. Therefore, the length of the chromosomes is equal to the number of features. In this case, the length of the chromosome is 17, the same as the number of input features of the prediction model. In the experiment, the size of the population was set to 1,000 and remained static during the process. To initiate the population, chromosomes were generated by randomly setting new values for the genes (i.e., features) within the scaled range of 0 to 1. Each feature had a 50% chance of having the value changed from the original instance. After having designed the population initialization, crossover and mutation operators were used to maintain the diversity of the population.

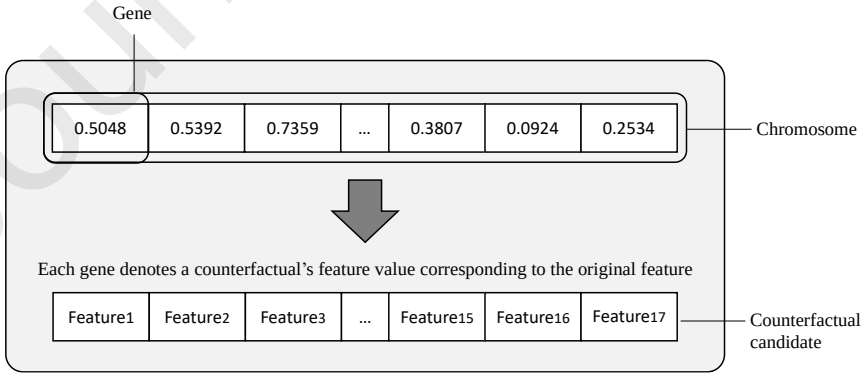


Figure 3 Chromosome encoding for counterfactual generation

After evaluating the chromosomes using the fitness function, it now evolves to the next generation using GA operators. In the selection process, chromosomes are selected based on the fitness of the chromosomes using methods such as the roulette method, tournament selection, and rank selection. For instance, in the tournament method, several tournaments are held with randomly chosen chromosomes, and the winner is selected based on the fitness value of each chromosome, which is similar to the tournament match. The tournament size indicates a number of chromosomes (i.e., players) for the tournament, and several tournament matches are held afterward to select the chromosome with the highest fitness value. In the experiment, the tournament method was used as it can be easily implemented and adjusted (Sharma, Wadhwa, and Komal 2014). The size of the tournament was set to four in the experiment.

In the crossover phase, the selected chromosomes go pairwise to generate new chromosomes. It is performed on two chromosomes (i.e., parent chromosomes) to create new chromosome(offspring), so it is performed 2/N times. It can be performed using the k-point crossover method that stochastically pins k points to parent chromosomes, and then the genes located between the points are swapped between the parent chromosomes. In this study, a two-point crossover was used. In the experiment, the crossover probability was set to 0.1, and it remained static during the course of the operation. In the case where there is no crossover operation, the same chromosomes are passed on to the next generation, and the offspring can replace the population when it exceeds the parents' fitness value.

In the mutation phase, the algorithm randomly mutates the genes in the chromosomes following a given probability. In this experiment, chromosomes were mutated by randomly resetting the values of the genes. It is an exploratory approach in GA operation to prevent falling in local optima. High mutation probability may delay convergence, but it may prevent the falling to local optima. Low mutation probability may lead to premature convergence. The process returns to the fitness evaluation step and repeats until satisfying the GA's stopping criterion. In the experiment, mutation probability was set to 0.7, and it remained static during the operation.

In this paper, GA was set to start the population by randomly generating new values for features within the scaled range of 0 to 1, and each feature had a 50% chance of having the value changed from the original instance. The genes of the chromosome are the feature values of the counterfactual example. Therefore, the length of the chromosomes is equal to the number of the features. In this case, the length of the chromosome is 17, the same as the number of input features of the prediction model.

As a very small difference in financial ratio carries only a little significance in reality, it makes more sense to disregard those small differences in financial variables during the process. For example, debt service coverage ratios (DSCRs) of 82% and 85% convey little difference in real life. Therefore, during the course of experiments to find counterfactuals, instead of discretizing all financial ratios, values within the range of ±10% of the original value were considered to have the same value as the original value. In the case of GA, each gene having ±10% of the original value was considered to have the original value during the operation.

4. Experiments

4.1 Data and Variables

The data used for the experiment contains 4,838 bankrupt cases and non-bankrupt cases. The dataset is balanced, and the observations consist of small and medium-sized firms in Korea over five years, between 2003–and 2007. The dataset consists of small and medium-sized firms in Korea over five years, between 2003–and 2007. Specifically, the firms were limited to manufacturing companies, not subject to external audits in Korea. Bankrupt firms in the dataset are the companies officially filed for bankruptcy. Initial features comprised 35 financial ratios. As this study focuses on the implementation of the explanation approach to the bankruptcy prediction model, a balanced dataset was used to avoid difficulty in model training due to data imbalance, although it is more common to have imbalanced bankruptcy data. Features with more than 50% of missing values, and financial variables with redundant meanings, were eliminated. Final input features were chosen using a stepwise feature selection method. As a result, 17 variables were selected. Features used as input variables can be categorized into five groups: activity, growth, productivity, profitability, and stability. Detailed information on the feature set used in the experiment is shown in Table 1. The features were scaled using a min-max standardization method via Eq (11) since the features are not on the same scale, such as ratios, days, and KRW.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{11}$$

Table 1 Selected input variables (KRW: Korean Won)

#	Var.	Variable Name	Category	#	Var.	Variable Name	Category
1	v11	Inventory turnover	activity	10	v415	Retained earnings to sales	profitability
2	v110	Working capital requirement (KRW)	activity	11	v418	CGS to sales	profitability
3	v13	Account payable turnover	activity	12	v423	Debt service coverage ratio (DSCR)	profitability
4	v17	Non-current asset turnover	activity	13	v47	Net income on shareholder's equity	profitability
5	v19	Working capital cycle (days)	activity	14	v51	Cash ratio	stability
6	v26	Owner's capital growth	growth	15	v56	Financial cost to sales	stability
7	v34	Labor income share rate	productivity	16	v513	Non-current assets to shareholders' equity	stability
8	v37	Gross value-added to machinery (KRW)	productivity	17	v515	Current liability ratio	stability
9	v39	Gross value added per capita (KRW)	profitability				

To generate counterfactual samples, the bankruptcy prediction model was trained on the training data. We split the data to 7:3 and used 70% of the data as a training set and 30% as a validation set. The detailed parameters settings for the bankruptcy prediction models are given in Table 2. A grid-search approach was used to find the parameters of the ANN and SVM. The parameter grid used for the ANN is as follows: neurons: [6,8,10,12,14], layers: [1,2,3], learning rate: [0.0001, 0.001, 0.01, 0.1], optimizer: [Adam, SGD]. As a result, the model has two hidden layers and 12 and 6 neurons for each hidden layer. Figure 4 shows the graphical representation of the ANN-based prediction model. As shown in the figure, the input layer has the number of neurons the same as the number of input features i, which in this case is 17. The first hidden layer has 12 neurons ($j = 12$) and the second hidden layer has 6 neurons ($k = 6$). We demonstrated the output layer with a single node ($o = 1$) using a sigmoid function for classification. When training the model, an early stopping mode with a patience of 20 was used to prevent overfitting. Early stopping mode uses a holdout set within the training set and stops model training if there are no performance improvements on the holdout set for a certain number of iterations. The model uses the sigmoid function for activation as it is a classification problem. For SVM, we used the parameter grid as: kernel: ['linear', 'rbf', 'poly'], gamma: [0.5, 0.1, 0.05, 0.01], C: [1, 10, 100, 1000]. As a result, we used a radial basis function (RBF) kernel with 0.5 and 1,000 for gamma and C, respectively.

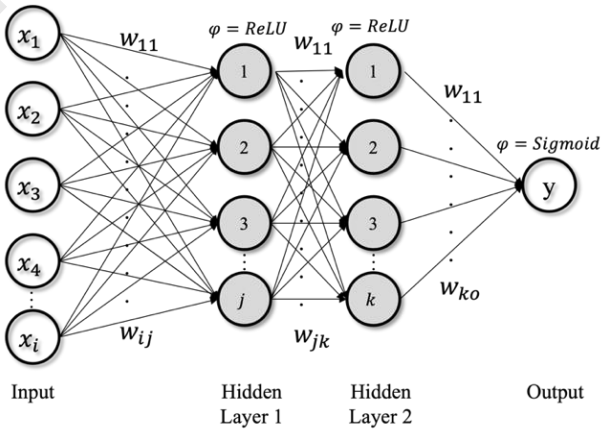


Figure 4 Graphical representation of the ANN model

Table 2 Model parameters for bankruptcy prediction models

Model	Parameters	Details
ANN	Number of hidden layers	2
	Number of neurons in the hidden layer	12, 6
	Transfer function	ReLU
	Transfer function (output layer)	sigmoid
	Optimizer	SGD
	Learning rate	0.01
	Epoch	max. 10,000
SVM	kernel	RBF
	C	1,000
	gamma	0.50

Grid-search method was used to find parameters

Table 3 Performance of the bankruptcy prediction models

Metrics	ANN	SVM
Accuracy	0.7672	0.8429
Precision	0.7407	0.8263
Recall	0.8223	0.8683
F-1 score	0.7794	0.8468
ROC AUC	0.8566	0.8429

For prediction evaluation, accuracy, precision, recall, F-1 score, and AUC-ROC was used. As the bankruptcy prediction model was designed as a binary classification problem in this paper, we adopted commonly used evaluation metrics for the classification model (i.e., Accuracy, Precision, Recall, F-1 score, and AUC). Accuracy indicates the percentage of samples that are correctly classified by the model and it is one of the most commonly used metrics to evaluate classification models. Precision measures the number of true positives over the number of true positives and false positives. Recall measures the number of true positives over the number of true positives and false negatives. F-1 score is the harmonic mean of precision and recall. ROC AUC indicates the area under the curve of the ROC plot. ROC curve shows the performance of the classification model at various decision thresholds. ROC AUC presents how much a model can distinguish between the class given an instance. The performance of the trained models on the validation set is shown in Table 3. The accuracy of the ANN-based bankruptcy prediction model was 76.72% and the accuracy of the SVM-based bankruptcy prediction model was 84.29%.

$$\arg\min_{x'} \max_{\lambda} \lambda(f_w(x') - y')^2 + d(x_i, x') \tag{12}$$

To compare the results, the nearest unlike sample, also called nearest unlike neighbor, and simple counterfactual generation method was used as benchmark models. Nearest contrastive sample approach locates an instance that is closest to the instance yet yielding a different result from the model. The nearest unlike sample, as the name suggests, does not offer a counterfactual explanation by generating one but uses a preexisting datapoint already in the dataset for a counterfactual explanation. It borrows a concept of case-based reasoning (CBR) directly but applies it to provide a contrastive sample as an explanation for a trained model. Therefore, it profoundly relates to the basic concept of the counterfactual-based explanation, except that the result of this approach is a real case world (Keane & Smyth, 2020; Kenny & Keane, 2021a; Nugent, Doyle, & Cunningham, 2009). We believe it is important to compare with the actual

case to justify the use of a synthetic counterfactual-based explanation. The simple counterfactual generation algorithm was used as another benchmark since the concept of counterfactual-based explanation for the machine learning model was first introduced (Wachter et al., 2018). The objective function defined in by the authors is presented in Eq. (12). Here, x_i is the original data point , and x' is the counterfactual. $d(x_i, x')$ measures the distance between the two. This method uses the Manhattan distance measure, and the distance is weighted using MAD. Maximization of λ is reached by iteratively solving for $x'$and increasing λ until the solution is found. In this experiment, a Nelder-Mead optimization technique was used to solve the optimization. We used Nelder-Mead as it is also a gradient-free method like GA and iterated the algorithm until finding the solution satisfying the validity. The experiment was conducted under Python 3.7 environment with customized algorithm for GA. To train the prediction model, Keras and Scikit-learn library were used.

4.2 Experiments and Results

For evaluation, three evaluation metrics were used: validity, sparsity, and LOF. Validity, sparsity, and proximity were considered critical factors in the counterfactual-based explanations (Mothilal, Sharma, & Tan, 2020; Russell, 2019). Proximity was excluded from the evaluation because the comparing methods use different distance measures. The proposed method uses the weighted Euclidean distance, while the nearest contrastive sample uses the simple Euclidean distance, and the simple counterfactual generation method uses the inverse MAD weighted Manhattan distance. LOF was included as an evaluation metric to measure how the produced counterfactuals follow the distribution of the dataset (Kanamori et al., 2020) and the paper uses the same metric to evaluate the counterfactuals. The performance measure for counterfactuals differs by the focus of the study, and despite an increasingly expanding use of counterfactual explanations, no uniform set of evaluation methods has been adopted so far (Stepin, Alonso, & Catala, 2021). Therefore, the most common and adequate quantitative measures were selected for evaluation.

$$Validity = \frac{Generated\ instances\ subject\ to\ desired\ f(\hat{x})}{S} \tag{13}$$

$$Sparsity = \frac{1}{S}\sum_{\hat{x}\in S} N\ ,\ N = number\ of\ features\ changed \tag{14}$$

$$LOF_{mean} = \frac{1}{S}\sum_{\hat{x}\in S} LOF_k \tag{15}$$

Validity is the total number of counterfactuals with the target class over the total number of counterfactuals generated, as presented in Eq. (13). S is the total number of instances in the dataset and the desired class of $f(\hat{x})$ is the prediction output we target to have with counterfactual, which is the opposite class of the original instance. Validity measures how the counterfactual generation algorithm produces counterfactuals with target output from the model. Sparsity, as shown in Eq. (14), measures an average number of features changed across the counterfactual sample set compared to the original inputs. A small number of changes in features is more compact and concise and, therefore, easier to understand as it guides the required changes to have a different result from the model. For this reason, a small value in sparsity is desired. To measure whether the generated counterfactuals fall under the distribution of the original dataset, the mean LOF score of the obtained counterfactuals was used, as shown in Eq. (15).

After training the models, the SHAP method was applied to acquire feature importance. The obtained feature importance of the ANN is shown in Figure 5 and the obtained feature importance of the SVM is shown in Figure 6. According to the SHAP result for ANN model, the most influential features in the model were identified as DSCR (v423) and Working capital cycle (v19). The feature importance of the SVM model was different compared to the ANN model. The DSCR (v423) had the largest impact on the prediction of the model as ANN. Financial cost to sales

(v56) and Working capital requirement (v110) were found relatively important in both models as well. However, Non-current asset turnover (v17) was found more important in SVM than in ANN while Current liability ratio (v515) was found less important in SVM and more important in ANN.

To generate counterfactuals, the GA was set to initiate a population size of 1,000 with a generation size of 10. The crossover rate was set to 0.7, and the mutation rate was set to 0.1. λ was used as a trade-off parameter between the objective terms. The detailed parameter setting for GA is presented in Table 4. It should be noted that to guarantee the validity of the counterfactuals, we employed a death penalty approach to penalize and eliminate the chromosomes that do not satisfy the target class.

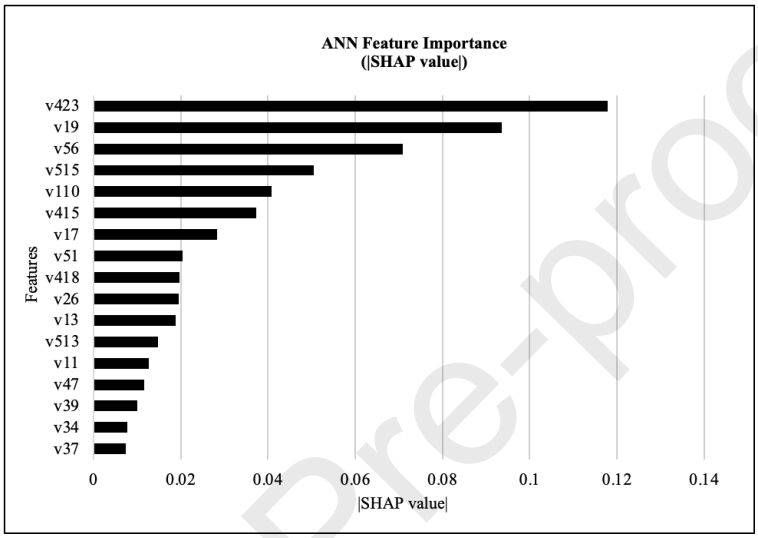


Figure 5 Feature importance obtained by SHAP for ANN model

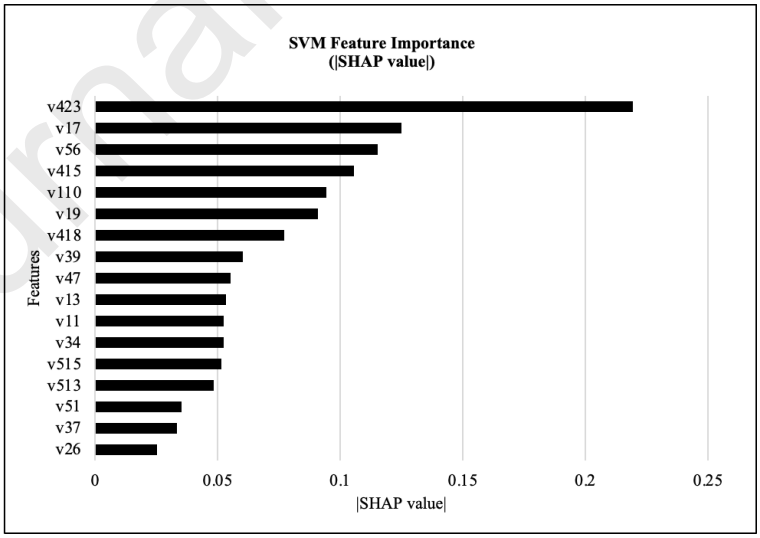


Figure 6 Feature importance obtained by SHAP for SVM model

Table 4 GA parameters

Parameters	Details
Fitness function	arg min $dist(x,\hat{x})\lambda_1 + lof(\hat{x})\lambda_2 + spr(\hat{x})\lambda_3$
Population	1,000
Max. generation	10
Selection	Tournament (size=4)
Crossover rate	0.7 (two-point crossover)
Mutation rate	0.1

$\lambda_1 = 0.6$ $\lambda_2 = 0.5$, $\lambda_3 = 0.6$

Table 5 Counterfactual generation algorithm performance

Eval. Metrics	Nearest Contrastive Sample		Simple Counterfactual*		Weighted Counterfactual (Proposed)	
	ANN	SVM	ANN	SVM	ANN	SVM
Validity	1.00	1.00	1.00	1.00	1.00	1.00
Sparsity	15.65	16.63	3.62	4.46	1.06	1.07
LOF (k= 20)	1.16	1.17	1.87	1.35	1.20	1.13

*Simple Counterfactual refers to the method proposed by Wachter et al. (2018)

Table 5 shows the performance of the proposed counterfactual generation algorithm and other baseline counterfactual algorithms applied to the validation set. Regarding the validity of the generated explanation, all methods in the experiment were guaranteed to qualify validity. The first base model nearest contrastive sample naturally guarantees the validity of the explanation as it is an instance found in the dataset. For this reason, the method is very intuitive and only involves real cases. The simple counterfactual method, the second base model, also had a guaranteed validity as the change in prediction was set as a constraint.

The proposed model presented the best results in sparsity in both ANN and SVM models. The proposed model had the smallest number of features changed for counterfactuals with 1.06 for ANN and 1.07 for SVM. The simple counterfactual algorithm showed a sparsity of 3.04 for ANN and had an increased sparsity of 4.46 in SVM. The nearest contrastive sample method had 15.65 in ANN and 16.63 in SVM. Regarding the sparsity of the nearest contrastive sample method, considering there are 17 input features, most of the features had to change to flip the decision using a real case.

The proposed model presented the LOF score of 1.20 in ANN and 1.13 in SVM. The simple counterfactual method showed LOF score of 1.87 and 1.35 for ANN and SVM model, respectively. The nearest contrastive sample approach had LOF score with 1.16 (ANN) and 1.17 (SVM), which is the lowest among all the other models. A low LOF score indicates that the instance is close to the distribution of the dataset, and it is not an outlier. It is deemed to be an inlier if the LOF score is near 1 and the higher the value is than 1 the more chances that the case is an outlier as it indicates lower density than its neighbors. Compared to the proposed model and the second base model, the nearest contrastive sample method showed the best results in terms of LOF score but showed the worst result in sparsity. As the result of the nearest contrastive sample suggests, finding a contrastive sample from the given dataset has some advantages over generating a synthetic one (i.e., counterfactuals) since the sample is a real instance, meaning that it is real (i.e., not synthetic) and avoids the risk of the sample being an infeasible case. However, as an explanation, this approach is difficult to understand and recommends actions that is very unlikely to follow as most of the features in the instance should be altered. On the other hand, the proposed model can offer explanations that are comprehensible with a small

number of features changed while maintaining a high chance of being the inlier, which indicates the feasibility of the explanation.

Figures 7 and Figure 8 show examples of generated counterfactual example from ANN. Figure 7 shows an example of counterfactuals for a bankrupt prediction case. Here, NCS refers to the nearest contrastive sample, and feature values changed from the original values are highlighted. As shown in the figure, counterfactuals provide an explanation as to what should have been different or what should be changed in order to flip the model prediction by changing certain features. For instance, Figure 8 demonstrates an example of the proposed counterfactual example from Figure 7 can be interpreted. The result can be easily interpreted and understood by the users. As shown in the figure, an explanation can be provided using the following text: Had the company (instance) managed to have a debt service coverage ratio of 63.49% instead of 19.78%, the model would not have predicted "bankrupt." This type of explanation can be further refined as not only an explanation but as a recommendation as well.

X_original

(Bankrupt)

LOF = 1.0123

Debt service coverage ratio(DSCR) (v423)	Working capital cycle (days) (v19)	Financial cost to sales (v56)	Cash ratio (v51)	Working capital requirement (KRW) (v110)	Retained earnings to sales (v415)
19.78	71.18	0.97	11.44	398973510	11.22
Non-current asset turnover (v17)	**Gross value added per capita (KRW) (v39)**	**Current liability ratio (v515)**	**Non-current assets to shareholders' equity (v513)**	**Gross value-added to machinary (KRW) (v37)**	**Owner's capital growth (v26)**
16.36	52026000	294.94	35.69	2192.16	2.59
Inventory turnover (v11)	**Accout payable turnover (v13)**	**Labor income share rate (v34)**	**Net income on shareholder's equity (v47)**	**CGS to sales (418)**	
2.28	2.94	76.55	4.54	84.96	

Proposed CF

(Non-bankrupt)

LOF= 1.0163

Debt service coverage ratio(DSCR) (v423)	Working capital cycle (days) (v19)	Financial cost to sales (v56)	Cash ratio (v51)	Working capital requirement (KRW) (v110)	Retained earnings to sales (v415)
63.49	71.18	0.97	11.44	398973510	11.22
Non-current asset turnover (v17)	**Gross value added per capita (KRW) (v39)**	**Current liability ratio (v515)**	**Non-current assets to shareholders' equity (v513)**	**Gross value-added to machinary (KRW) (v37)**	**Owner's capital growth (v26)**
16.36	52026000	294.94	35.69	2192.16	2.59
Inventory turnover (v11)	**Accout payable turnover (v13)**	**Labor income share rate (v34)**	**Net income on shareholder's equity (v47)**	**CGS to sales (418)**	
2.28	2.94	76.55	4.54	84.96	

Simple CF

(Non-bankrupt)

LOF= 1.1327

Debt service coverage ratio(DSCR) (v423)	Working capital cycle (days) (v19)	Financial cost to sales (v56)	Cash ratio (v51)	Working capital requirement (KRW) (v110)	Retained earnings to sales (v415)
19.78	*64.17*	*0.91*	11.44	398973510	11.22
Non-current asset turnover (v17)	**Gross value added per capita (KRW) (v39)**	**Current liability ratio (v515)**	**Non-current assets to shareholders' equity (v513)**	**Gross value-added to machinary (KRW) (v37)**	**Owner's capital growth (v26)**
16.36	52026000	294.94	35.69	2192.16	2.59
Inventory turnover (v11)	**Accout payable turnover (v13)**	**Labor income share rate (v34)**	**Net income on shareholder's equity (v47)**	**CGS to sales (418)**	
2.28	2.94	76.55	4.54	*52.09*	

NCS

(Non-bankrupt)

LOF= 0.9813

Debt service coverage ratio(DSCR) (v423)	Working capital cycle (days) (v19)	Financial cost to sales (v56)	Cash ratio (v51)	Working capital requirement (KRW) (v110)	Retained earnings to sales (v415)
34.10	*67.06*	*0.99*	*1.42*	381573260	7.06
Non-current asset turnover (v17)	**Gross value added per capita (KRW) (v39)**	**Current liability ratio (v515)**	**Non-current assets to shareholders' equity (v513)**	**Gross value-added to machinary (KRW) (v37)**	**Owner's capital growth (v26)**
12.57	*38100000*	*211.48*	*42.35*	*793.75*	*12.27*
Inventory turnover (v11)	**Accout payable turnover (v13)**	**Labor income share rate (v34)**	**Net income on shareholder's equity (v47)**	**CGS to sales (418)**	
4.50	*14.17*	76.29	*16.00*	*89.11*	

Figure 7 Generated counterfactual example(Bankrupt ➔ Non- Bankrupt)

Counterfactual Explanation Example

Model Prediction: Non-bankrupt (Original instance prediction: bankrupt)

Debt service coverage ratio(DSCR)	Working capital cycle (days)	Financial cost to sales	Cash ratio	Working capital requirement (KRW)	Retained earnings to sales
63.49	71.18	0.97	11.44	398,973,510	11.22
Non-current asset turnover	Gross value added per capita (KRW)	Current liability ratio	Non-current assets to shareholders' equity	Gross value-added to machinary (KRW)	Owner's capital growth
16.36	52,026,000	294.94	35.69	2192.16	2.59
Inventory turnover	Accout payable turnover	Labor income share rate	Net income on shareholder's equity	CGS to sales	
2.28	2.94	76.55	4.54	84.96	

Explanation Text: If the case were to have 'Debt service coverage ratio(DSCR)' of 63.49% instead of 19.78%, the company would have been predicted to be a non-bankrupt case rather than bankrupt case.

Figure 8 Counterfactual explanation example

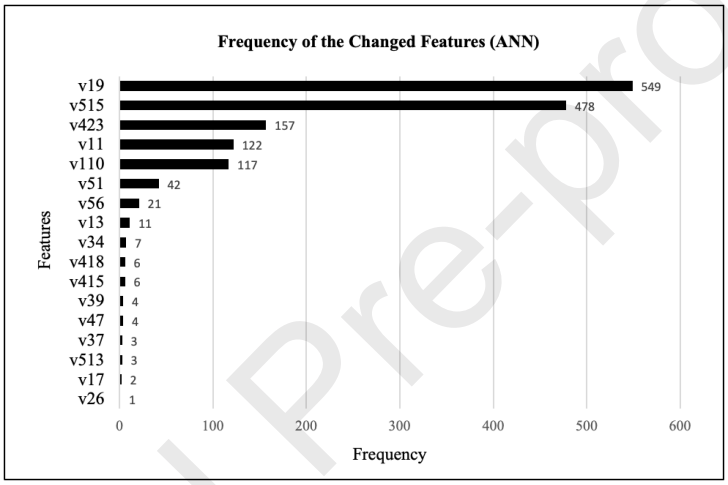


Figure 9 Frequency of the changes features (ANN)

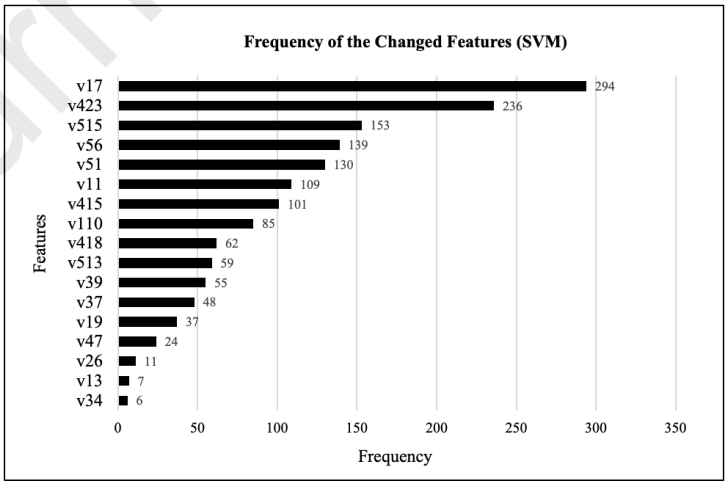


Figure 10 Frequency of the changed features (SVM)

The proposed model incorporated feature importance using the SHAP values that computes the marginal contribution of the feature to the prediction model. Figure 9 and Figure 10 show the frequency of the altered features in counterfactuals from ANN and SVM to compare the result of the feature importance and the altered features in the counterfactual. The reason why the proposed model used feature importance to inversely weigh the distance is to generate counterfactuals that are more relevant to the model by inducing changes in influential features rather than any other features with minimal changes. The results shows that most of the counterfactuals were produced with changes in important features. However, some features did not follow the order of the feature importance but not exactly the same. As shown in Figure 9, working capital cycle (v19), current liability ratio (v515), and debt service coverage ratio (v423) were found to be both high in the rank of feature importance and altered features. On the contrary, Inventory turnover (v11) was found to be changed often to generate counterfactuals, whereas it was not highly ranked in feature importance. In Figure 10 shows the frequency of the altered features in SVM model. The top two most frequently changed features (v17 and v423) are consistent with the feature importance obtained by SHAP. However, Current liability ratio (v515) was frequently used in generated counterfactuals although it presented relatively low feature importance by SHAP. Also, the results show that the frequency of the altered features is more dispersed in SVM model than ANN with the given set of trade-off parameters in the GA.

In sum, the results show that the strengths of the nearest contrastive sample are its guaranteed validity and low LOF score, while its high sparsity is its critical disadvantage in its use as a counterfactual explanation. Incapability to ensure the low sparsity of the counterfactuals is the critical drawback of using an actual case. On the other hand, the proposed method has the lowest number of features altered in generated counterfactuals. It shows that the proposed method can provide the most compact explanation with only a few features changed. Considering that the algorithm also promotes features by using a weighted-distance measure, the explanations generated by the proposed method are highly relevant. It also shows a lower LOF score than the simple counterfactual-generation algorithm without the term considering the distribution of the original data.

Also, a Wilcoxon signed-rank test statistical significance test was performed on the proposed model and the benchmark models to verify that the performance of the proposed method is statistically significant compared to other models. A non-parametric Wilcoxon signed-rank test method employed as the Shapiro–Wilks test showed that the results are not normally distributed. Therefore, a non-parametric Wilcoxon test was employed for each evaluation metric to compare the proposed model with two different models. Since the validity of the models is equal, validity was excluded from the test. The results of the significance test on ANN and SVM models are presented in the Table 6. The empirical test suggests that the performance of the proposed model is statistically significant in terms of both sparsity and LOF at the significance level of 0.00005 compared to other models.

Table 6 Wilcoxon signed-rank test results

Model	Eval. Metric	Merhods	Mean	Median	Std Dev	p-value
ANN	Sparsity	Proposed Model (Weighted Counterfactual)	1.06	1.00	0.29	-
		Nearest Contrsative Sample	15.65	16.00	1.83	< .00005
		Simple Counterfactual(Nelder-Mead)	3.62	3.00	1.70	< .00005
	LOF (k=20)	Proposed Model (Weighted Counterfactual)	1.20	1.18	0.27	-
		Nearest Contrsative Sample	1.16	1.17	0.31	< .00005
		Simple Counterfactual (Nelder-Mead)	1.87	1.89	0.52	< .00005
SVM	Sparsity	Proposed Model (Weighted Counterfactual)	1.07	1.00	0.32	-
		Nearest Contrsative Sample	16.63	17.00	1.11	< .00005
		Simple Counterfactual (Nelder-Mead)	4.46	4.00	2.46	< .00005
	LOF (k=20)	Proposed Model (Weighted Counterfactual)	1.13	1.08	0.25	-
		Nearest Contrsative Sample	1.17	1.18	0.32	< .00005
		Simple Counterfactual (Nelder-Mead)	1.35	1.35	0.43	< .00005

4.3 GA Performance Analysis

In this section, we conducted a performance analysis of the proposed model by a different number of generations. Figure 9 shows the average sparsity and LOF of the model per generation. Setting all other parameters of the GA fixed as in the experiment, the maximum number of generations in the GA was changed to see how it converges and affects the performance of the model. The prediction model used for the performance analysis is ANN. As presented in Figure 9, both Sparsity and LOF tend to decrease as the generation increases after some fluctuations. Sparsity reached convergence early compared to the LOF score. The minimum sparsity is one and the model almost reached its minimum at 3rd generation. LOF steadily decreased with some fluctuation as the generation increased. However, as mentioned above, the experiment had maximum 10 generations and it is mainly due to computation time cost. Therefore, we also added an average computation time by generation in Figure 10. As shown in Figure 10, the computation time increases as maximum generation increases. It begins with 12.69 seconds in the 1st generation and increases up to 114.56 seconds in the 10th generation. In the 20th generation, it takes over 324seconds. It is essential to consider the trade-off between the performance and computational cost when deciding on the number of generations in the GA operation.

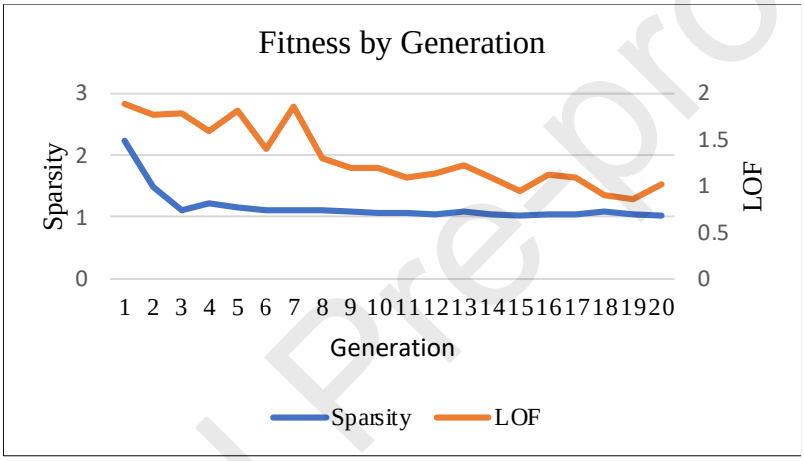


Figure 11 Proposed model performance by generations

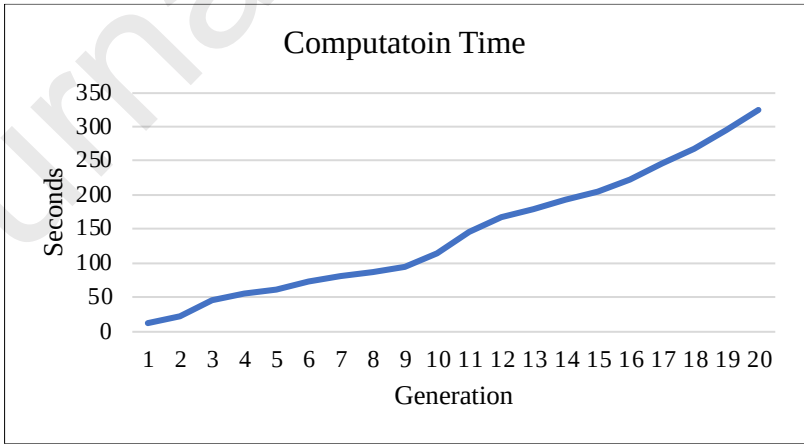


Figure 12 Computation time by generations

4.4 Experiment with other bankruptcy datasets

To further verify the proposed method, we conducted experiments using other publicly available datasets. We used the Taiwanese bankruptcy dataset from UCI depository and HMEQ (Home Equity Loan Default) dataset from Kaggle. The Taiwanese dataset was collected from the Taiwan Economic Journal between 1999 and 2009 and the bankruptcy of the companies was defined by the business regulations of the Taiwan Stock Exchange. As in the previous study (Liang, Lu, Tsai, & Shih, 2016) using the Taiwanese dataset, we used a balanced set of bankrupt and non-bankrupt cases in a total of 440 (bankrupt: 220, non-bankrupt: 220). The original data has 95 numerical variables and we used 24 variables using stepwise feature selection. HMEQ dataset consists of the loan for 5,960 home equity loans, and we used balanced set of 1,189 'bad' and 1,189 'good' cases using stratified sampling. The bad observations are the ones who eventually defaulted or was seriously delinquent. There are 12 input variables and we used 10 numerical variables among them. We trained the prediction model with ANN and SVM using grid-search to find the parameters. The performance of the prediction models on the validation set is shown in Table 7. For Taiwanese dataset, the model had 84.85% accuracy with ANN and 87.12% with SVM. For HMEQ dataset, the model achieved 75.91% accuracy with ANN and 78.31% with SVM.

Table 7 Performance of the bankruptcy prediction models (Taiwanese and HMEQ dataset)

Metrics	Taiwanese		HMEQ	
	ANN	SVM	ANN	SVM
Accuracy	0.8485	0.8712	0.7591	0.7831
Precision	0.8194	0.8889	0.8571	0.8195
Recall	0.8939	0.8485	0.6218	0.7260
F-1 score	0.8551	0.8682	0.7208	0.7699
ROC AUC	0.8485	0.8712	0.7591	0.7831

Table 8 Feature importance and the frequency of the altered features from the Taiwanese dataset

Features	ANN		SVM		Features	ANN		SVM	
	SHAP	Freq.	SHAP	Freq.		SHAP	Freq.	SHAP	Freq.
X19	0.0183	-	0.0416	3	X58	0.0026	-	0.0043	-
X38	0.0714	87	0.0809	99	X29	0.0063	-	0.0061	-
X81	0.0026	-	0.0001	-	X53	0.0103	-	0.0154	-
X57	0.0471	1	0.0290	1	X64	0.0114	-	0.0073	-
X79	0.0032	-	0.0023	-	X28	0.0022	-	0.0037	-
X47	0.0008	-	0.0067	-	X26	0.0035	-	0.0023	-
X51	0.0048	-	0.0034	1	X18	0.0333	-	0.0022	-
X91	0.0027	-	0.0001	-	X16	0.0527	22	0.0026	-
X52	0.0127	-	0.0071	3	X1	0.0411	15	0.0049	-
X68	0.0373	13	0.0543	25	X27	0.0041	1	0.0024	-
X34	0.0154	2	0.0088	12	X46	0.0010	-	0.0003	-
X83	0.0030	-	0.0020	-	X10	0.0028	-	0.0038	-

Table 9 Feature importance and the frequency of the altered features from the HMEQ dataset

Features	ANN		SVM	
	SHAP	Freq.	SHAP	Freq.
X1	0.0515	196	0.0474	44
X2	0.0490	9	0.0241	20
X3	0.0353	4	0.0229	11
X6	0.0318	3	0.0441	68
X7	0.0552	42	0.0613	64
X8	0.1281	319	0.1118	243
X9	0.0802	76	0.0611	57
X10	0.0450	7	0.0484	102
X11	0.0522	114	0.0484	121
X12	0.0404	4	0.0400	42

We then applied the nearest contrastive sample, simple counterfactual, and the proposed weighted counterfactual generation method. The derived SHAP values and the frequency of the changed features in the counterfactual examples are shown in the Table 8 and Table 9. Table 8 shows the result from the Taiwanese dataset and Table 9 shows the results of the HMEQ dataset. The SHAP values are the mean of the absolute SHAP values, and the frequency shows the sum of the changed features from the generated counterfactual examples. As the Taiwanese dataset has relatively small number of observations compared to the number of input features, most of the features altered in the counterfactuals are concentrated mainly in the features with higher SHAP values. With HMEQ dataset, similar with other models, the features altered in the counterfactuals generally follow the order of feature importance obtained by SHAP. Also, as HMEQ dataset has smaller number of input features, more diverse features were changed compared to the Taiwanese dataset.

The result of the counterfactual example generation is shown in Table 10. To generate the counterfactuals, we used the same parameters of the GA used for the Korean bankruptcy dataset. Similar to the previous experiment with the Korean bankruptcy prediction dataset, the proposed method demonstrated superior performance especially in sparsity compared to other models. Also, the nearest contrastive sample method showed its strength in LOF score as it is using real data. Regardless of the base models and the dataset, the proposed method maintained to have only few number of features changed on the counterfactuals compared to other models as well as LOF score. On the other hand, the sparsity and LOF score of the simple counterfactual model varied depending on the dataset. For Taiwanese dataset, it used 5.62 and 5.39 features in the counterfactual and for HMEQ dataset, it used 2.72 and 2.43 features for counterfactuals.

Table 10 Counterfactual generation algorithm performance in Taiwanese and HMEQ dataset

Model	Data	Taiwanese		HMEQ	
	Eval. Metrics	ANN	SVM	ANN	SVM
Nearest Contrstive Sample	Validity	1.00	1.00	1.00	1.00
	LOF	1.38	1.20	1.12	1.20
	Sparsity	23.78	23.75	9.41	9.46
Simple Counterfactual*	Validity	1.00	1.00	1.00	1.00
	LOF	2.81	1.72	1.28	1.89
	Sparsity	5.62	5.39	2.72	2.43
Weighted Counterfactual (Proposed)	Validity	1.00	1.00	1.00	1.00
	LOF	1.56	1.33	1.19	1.18
	Sparsity	1.07	1.09	1.08	1.06

*Simple Counterfactual refers to the method proposed by Wachter et al. (2018)

5. Conclusion

In this paper, we proposed a model-agonistic feature-weighted counterfactual generation method using multi-objective GA for bankruptcy prediction models. The proposed method has several advantages over other methods as it considers feature importance and multiple criteria. Using feature importance derived from the SHAP method makes it possible to apply to any trained "black-box" model and to take important model-relevant features into account. Approaching the problem with multiple objectives can provide counterfactuals that satisfy key features to offer a more comprehensible explanation. We conducted experiments using various bankruptcy datasets and machine learning techniques such as ANN and SVM. The empirical experiment shows that the overall performance of the proposed method is better than a simple counterfactual-generation algorithm. Although it is evident that the nearest contrastive sample provides a realistic explanation, it is hardly a compact and effective counterfactual-based explanation.

Not only in bankruptcy prediction models, but there are other approaches to handle interpretability issues focusing on the recipients of the explanations in other domains such as in medicine (Dragoni, Donadello, & Eccher, 2020) and manufacturing industry including automotive and machine-tool (Bilbao-Ubillos, Camino-Beldarrain, & Intxaurburu, 2020). As the proposed method employs counterfactual example-based explanations using relevant features for more comprehensible and acceptable explanations for the users, the method can potentially be used in other fields of the industry than finance.

In this study, we trained the prediction model using a balanced bankruptcy dataset. However, it is more likely that we can enhance the prediction model by adopting techniques to handle the imbalance problem. Although we did not focus on training the base models for better performance using imbalance techniques, future study should investigate how the performance of the model affects the quality of the counterfactuals, as counterfactual-based explanation is based on generating a synthetic sample that flips the model output. This study used GA for the explanation generation method, however, the study did not use the optimized set of parameters in GA operation for neighborhood generation. Also, the trade-off parameters between the objective terms were not optimized but chosen after trial and error. This paper uses a LOF score as a measure to encourage the generated sample to remain as inliers in the dataset by preventing it from being an outlier. However, it is difficult to find the appropriate threshold for a LOF score to identify a data point as an outlier as it depends on the particular dataset. Future works should consider using the diverse dataset from the financial domain and optimization of the model parameters. Furthermore, qualitative analysis with a user study and optimization of the weights between the objective terms to generate explanations.

References

Adhikari, A., Tax, D. M. J. J., Satta, R., & Faeth, M. (2019). LEAFAGE: Example-based and Feature importance-based Explanations for Black-box ML models. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (pp. 0–6). IEEE. https://doi.org/10.1109/FUZZ-IEEE.2019.8858846

Alaka, H. A., Oyedele, L. O., Owolabi, H. A., Kumar, V., Ajayi, S. O., Akinade, O. O., & Bilal, M. (2018). Systematic review of bankruptcy prediction models: Towards a framework for tool selection. *Expert Systems with Applications*, *94*, 164–184. https://doi.org/10.1016/j.eswa.2017.10.040

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., … Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*(December 2019), 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

Belkoura, S., Zanin, M., & Latorre, A. (2019). Fostering interpretability of data mining models through data perturbation. *Expert Systems With Applications*, *137*, 191–201. https://doi.org/10.1016/j.eswa.2019.07.001

Berrada, M., Adadi, A., & Berrada, M. (2018). Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, *6*, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

Bilbao-Ubillos, J., Camino-Beldarrain, V., & Intxaurburu, G. (2020). A technology-based explanation of industrial output processes: the automotive, machine-tool and "other transport material" industries. *Journal of Knowledge Management*, *25*(6), 1640–1661. https://doi.org/10.1108/JKM-07-2020-0582

Breunig, M. M., Kriegel, H., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. In *ACM SIGMOD 2000 Int. Conf. On Management of Data, Dalles, Texas* (Vol. 9, pp. 4–23). Retrieved from http://allman.rhon.itam.mx/~mendoza/Foresight.pdf%0Ahttps://pdfs.semanticscholar.org/e390/c5d56ddcc8e9f6f27264ee7196539d0e7f78.pdf%0Ahttps://doi.org/10.1016/j.ijforecast.2018.07.006%0Ahttp://arxiv.org/abs/1903.05440

Byrne, R. M. J. (2019). Counterfactuals in Explainable Artificial Intelligence (XAI): Evidence from Human Reasoning. In *International Joint Conference on Artificial Intelligence (IJCAI-19)* (pp. 6276–6282).

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*(8), 1–34. https://doi.org/10.3390/electronics8080832

Dandl, S., Molnar, C., Binder, M., & Bischl, B. (2020). Multi-Objective Counterfactual Explanations. In *Parallel Problem Solving from Nature – PPSN XVI. PPSN 2020. Lecture Notes in Computer Science* (pp. 448–469). Springer International Publishing. https://doi.org/10.1007/978-3-030-58112-1

Dastile, X., Celik, T., & Potsane, M. (2020). Statistical and machine learning models in credit scoring: A systematic literature survey. *Applied Soft Computing Journal*, *91*(106263), 106263. https://doi.org/10.1016/j.asoc.2020.106263

Dong, L., Ye, X., & Yang, G. (2021). Two-stage rule extraction method based on tree ensemble model for interpretable loan evaluation. *Information Sciences*, *573*, 46–64. https://doi.org/10.1016/j.ins.2021.05.063

Dragoni, M., Donadello, I., & Eccher, C. (2020). Explainable AI meets persuasiveness: Translating reasoning results into behavioral change advice. *Artificial Intelligence in Medicine*, *105*(January), 101840. https://doi.org/10.1016/j.artmed.2020.101840

Du Jardin, P. (2016). A two-stage classification technique for bankruptcy prediction. *European Journal of Operational Research*, *254*(1), 236–252. https://doi.org/10.1016/j.ejor.2016.03.008

Feng, X., Xiao, Z., Zhong, B., Qiu, J., & Dong, Y. (2018). Dynamic ensemble classification for credit scoring using

soft probability. *Applied Soft Computing Journal*, *65*, 139–151. https://doi.org/10.1016/j.asoc.2018.01.021

Fernández, R. R., Martín de Diego, I., Aceña, V., Fernández-Isabel, A., & Moguerza, J. M. (2020). Random forest explainability using counterfactual sets. *Information Fusion*, *63*, 196–207. https://doi.org/10.1016/j.inffus.2020.07.001

Goyal, Y., Wu, Z., Ernst, J., Batra, D., Parikh, D., & Lee, S. (2019). Counterfactual Visual Explanations. In *36th International Conference on Machine Learning(ICML)* (Vol. 2019-June, pp. 4254–4262).

Grath, R. M., Costabello, L., Le Van, C., Sweeney, P., Kamiab, F., Shen, Z., & Lécué, F. (2018). Interpretable credit application predictions with counterfactual explanations. In *NIPS 2018 Workshop on Challenges and Opportunities for AI inFinancial Services: the Impact of Fairness,Explainability, Accuracy, and Privacy*. Montreal, Canada. https://doi.org/https://doi.org/10.48550/arXiv.1811.05245

Guidotti, R., Monreale, A., Ruggieri, S., Giannotti, F., Pedreschi, D., & Turini, F. (2019). Factual and Counterfactual Explanations for Black Box Decision Making. *IEEE Intelligent Systems*, *November/D*, 14–23.

Hashemi, M., & Fathi, A. (2020). PermuteAttack: Counterfactual explanation of machine learning credit scorecards. *ArXiv*.

Hayashi, Y. (2016). Application of a rule extraction algorithm family based on the Re-RX algorithm to financial credit risk assessment from a Pareto optimal perspective. *Operations Research Perspectives*, *3*, 32–42. https://doi.org/10.1016/j.orp.2016.08.001

He, H., Zhang, W., & Zhang, S. (2018). A novel ensemble method for credit scoring: Adaption of different imbalance ratios. *Expert Systems with Applications*, *98*, 105–117. https://doi.org/10.1016/j.eswa.2018.01.012

Henley, W. E., & Hand, D. J. (1996). A k-Nearest-Neighbour Classifier for Assessing Consumer Credit Risk. *The Statistician*, *45*(1), 77–95. Retrieved from https://www.jstor.org/stable/2348414

Islam, M. R., Ahmed, M. U., Barua, S., & Begum, S. (2022). A Systematic Review of Explainable Artificial Intelligence in Terms of Different Application Domains and Tasks. *Applied Sciences*, *12*(3). https://doi.org/10.3390/app12031353

Kanamori, K., Takagi, T., Kobayashi, K., & Arimura, H. (2020). DACE: Distribution-aware counterfactual explanation by mixed-integer linear optimization. *IJCAI International Joint Conference on Artificial Intelligence*, 2855–2862. https://doi.org/10.24963/ijcai.2020/395

Keane, M. T., & Smyth, B. (2020). Good Counterfactuals and Where to Find Them: A Case-Based Technique for Generating Counterfactuals for Explainable AI (XAI). *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12311 LNAI*, 163–178. https://doi.org/10.1007/978-3-030-58342-2_11

Kenny, E. M., Ford, C., Quinn, M., & Keane, M. T. (2021). Explaining black-box classifiers using post-hoc explanations-by-example: The effect of explanations and error-rates in XAI user studies. *Artificial Intelligence*, *294*, 103459. https://doi.org/10.1016/j.artint.2021.103459

Kenny, E. M., & Keane, M. T. (2019). Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In *IJCAI International Joint Conference on Artificial Intelligence* (pp. 2708–2715). https://doi.org/10.24963/ijcai.2019/376

Kenny, E. M., & Keane, M. T. (2021a). Explaining Deep Learning using examples: Optimal feature weighting methods for twin systems using post-hoc, explanation-by-example in XAI. *Knowledge-Based Systems*, *233*, 107530. https://doi.org/10.1016/j.knosys.2021.107530

Kenny, E. M., & Keane, M. T. (2021b). On Generating Plausible Counterfactual and Semi-Factual Explanations for Deep Learning. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)* (pp. 11575–11585). Retrieved from http://arxiv.org/abs/2009.06399

Kwon, B. C., Choi, M. J., Kim, J. T., Choi, E., Kim, Y. Bin, Kwon, S., … Choo, J. (2019). RetainVis: Visual Analytics with Interpretable and Interactive Recurrent Neural Networks on Electronic Medical Records. *IEEE Transactions on Visualization and Computer Graphics*, *25*(1), 299–309. https://doi.org/10.1109/TVCG.2018.2865027

Le, T., Wang, S., & Lee, D. (2020). GRACE : Generating Concise and Informative Contrastive Sample to Explain Neural Network Model ' s Prediction. In *KDD '20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 238–248). Association for Computing MachineryNew YorkNYUnited States. https://doi.org/https://doi.org/10.1145/3394486.3403066

Liang, D., Lu, C. C., Tsai, C. F., & Shih, G. A. (2016). Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study. *European Journal of Operational Research*, *252*(2), 561–572. https://doi.org/10.1016/j.ejor.2016.01.012

Lundberg, S. M., & Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. In *31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA* (pp. 1–10).

Mahajan, D., Tan, C., & Sharma, A. (2019). Preserving causal constraints in counterfactual explanations for machine learning classifiers. In *33rd Conferenceon Neural Information Processing Systems*.

Marqués, A. I., García, V., & Sánchez, J. S. (2012). Two-level classifier ensembles for credit risk assessment. *Expert Systems with Applications*, *39*(12), 10916–10922. https://doi.org/10.1016/j.eswa.2012.03.033

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38. https://doi.org/10.1016/j.artint.2018.07.007

Mittelstadt, B., Russell, C., & Wachter, S. (2019). Explaining explanations in AI. *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, 279–288. https://doi.org/10.1145/3287560.3287574

Moscatelli, M., Parlapiano, F., Narizzano, S., & Viggiano, G. (2020). Corporate default forecasting with machine learning. *Expert Systems with Applications*, *161*, 113567. https://doi.org/10.1016/j.eswa.2020.113567

Mothilal, R. K., Sharma, A., & Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. *Conference on Fairness, Accountability, and Transparency*, 607–617. https://doi.org/10.1145/3351095.3372850

Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, *50*(3), 559–569. https://doi.org/10.1016/j.dss.2010.08.006

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, *452*, 48–62. https://doi.org/10.1016/j.neucom.2021.03.091

Nugent, C., Doyle, D., & Cunningham, P. (2009). Gaining insight through case-based explanation. *Journal of Intelligent Information Systems*, *32*(3), 267–295. https://doi.org/10.1007/s10844-008-0069-0

Poyiadzi, R., Sokol, K., Santos-rodriguez, R., Bie, T. De, & Flach, P. (2020). FACE : Feasible and Actionable Counterfactual Explanations. In *AAAI/ACMConference on AI, Ethics, and Society (AIES)*. New York: ACM. https://doi.org/https://doi.org/10.1145/3375627.3375850 1

Rajapaksha, D., Bergmeir, C., & Buntine, W. (2020). LoRMIkA: Local rule-based model interpretability with k-optimal associations. *Information Sciences*, *540*, 221–241. https://doi.org/10.1016/j.ins.2020.05.126

Ribeiro, M. T., & Guestrin, C. (2016). " Why Should I Trust You ?" Explaining the Predictions of Any Classifier. In *KDD 2016 San Francisco, CA, USA*. ACM. https://doi.org/http://dx.doi.org/10.1145/2939672.2939778

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors : High-Precision Model-Agnostic Explanations. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)* (pp. 1527–1535).

Rodriguez, P., Caccia, M., Lacoste, A., Zamparo, L., Laradji, I., Charlin, L., & Vazquez, D. (2022). Beyond Trivial Counterfactual Explanations with Diverse Valuable Explanations. In *International Conference on Computer Vision (ICCV)* (pp. 1036–1045). https://doi.org/10.1109/iccv48922.2021.00109

Russell, C. (2019). Efficient search for diverse coherent explanations. *Conference on Fairness, Accountability, and Transparency*, (January), 20–28. https://doi.org/10.1145/3287560.3287569

Setiono, R., & Liu, H. (1996). Symbolic Representation of Neural Networks. *Computer*, (March), 71–77.

Shimizu, R., Matsutani, M., & Goto, M. (2022). An explainable recommendation framework based on an improved knowledge graph attention network with massive volumes of side information. *Knowledge-Based Systems*, *239*, 107970. https://doi.org/10.1016/j.knosys.2021.107970

Son, H., Hyun, C., Phan, D., & Hwang, H. J. J. (2019). Data analytic approach for bankruptcy prediction. *Expert Systems with Applications*, *138*, 112816. https://doi.org/10.1016/j.eswa.2019.07.033

Soui, M., Gasmi, I., Smiti, S., & Ghédira, K. (2019). Rule-based credit risk assessment model using multi-objective evolutionary algorithms. *Expert Systems with Applications*, *126*, 144–157. https://doi.org/10.1016/j.eswa.2019.01.078

Stepin, I., Alonso, J. M., & Catala, A. (2021). A Survey of Contrastive and Counterfactual Explanation Generation Methods for Explainable Artificial Intelligence. *IEEE Access*, *9*. https://doi.org/10.1109/ACCESS.2021.3051315

Tsirtsis, S., De, A., & Gomez-Rodriguez, M. (2021). Counterfactual Explanations in Sequential Decision Making Under Uncertainty. In *35th Conference on Neural Information Processing Systems (NeurIPS)*. Retrieved from http://arxiv.org/abs/2107.02776

Verma, S., Dickerson, J., & Hines, K. (2020). Counterfactual Explanations for Machine Learning : A Review. In *arXiv* (pp. 1–13).

Wachter, S., Mittelstadt, B., & Russell, C. (2018). COUNTERFACTUAL EXPLANATIONS WITHOUT OPENING THE BLACK BOX : AUTOMATED DECISIONS AND THE GDPR. *Harvard Journal of Law & Technology*, *31*(2), 842–887.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Highlights

- Counterfactual example-based explanation
- Explainable bankruptcy prediction model
- Feature-weighted multi-objective counterfactuals
- GA-based counterfactual generation algorithm