Understanding algorithm bias in artificial intelligence-enabled ERP software customization

Sudhaman Parthasarathy and S.T. Padmapriya
Department of Applied Mathematics and Computational Science, Thiagarajar College of Engineering, Madurai, India

Received 24 April 2023
Revised 24 May 2023
Accepted 2 June 2023

Abstract

Purpose – Algorithm bias refers to repetitive computer program errors that give some users more weight than others. The aim of this article is to provide a deeper insight of algorithm bias in AI-enabled ERP software customization. Although algorithmic bias in machine learning models has uneven, unfair and unjust impacts, research on it is mostly anecdotal and scattered.

Design/methodology/approach – As guided by the previous research (Akter *et al.*, 2022), this study presents the possible design bias (model, data and method) one may experience with enterprise resource planning (ERP) software customization algorithm. This study then presents the artificial intelligence (AI) version of ERP customization algorithm using k-nearest neighbours algorithm.

Findings – This study illustrates the possible bias when the prioritized requirements customization estimation (PRCE) algorithm available in the ERP literature is executed without any AI. Then, the authors present their newly developed AI version of the PRCE algorithm that uses ML techniques. The authors then discuss its adjoining algorithmic bias with an illustration. Further, the authors also draw a roadmap for managing algorithmic bias during ERP customization in practice.

Originality/value – To the best of the authors' knowledge, no prior research has attempted to understand the algorithmic bias that occurs during the execution of the ERP customization algorithm (with or without AI).

Keywords Algorithms, Bias, Customization, Machine learning, ERP projects

Paper type Research paper

1. Introduction

To offer cross-functional services to a business, vendors of packaged software such as SAP deliver enterprise resource planning (ERP) solutions. How effectively the selected ERP package satisfies the business needs of the adopting organization will determine how customized the ERP deployment will be (Mahmood *et al.*, 2020). ERP, an integrated software system, has the important feature of being created for many enterprises with different requirements, located in different geographic areas and also incorporates best practices and processes. In conventional information systems, the product is often created for a single customer, using software development techniques to gather requirements at the beginning of the process with the idea that the finished product will satisfy those criteria (Febrianto and Soediantono, 2022).

Journal of Ethics in Entrepreneurship and Technology
Vol. 3 No. 2, 2023
pp. 79-93
Emerald Publishing Limited
e-ISSN: 2633-7444
p-ISSN: 2633-7436
DOI 10.1108/JEET-04-2023-0006

Most of ERP systems require some customization for both business processes and ERP systems. The effect of ERP customization on ERP deployment has been identified in several earlier studies (Wang *et al.*, 2022). For an implementation to be effective, it is critical to minimize the customization and maintain the current business processes. In practice, ERP software customization is carried out with the help of conceptual frameworks suggested by previous studies (Brehm *et al.*, 2001; Luo and Strong, 2004; Rothenberger and Srite, 2009; Zach *et al.*, 2014; Mahmood *et al.*, 2020; Febrianto and Soediantono, 2022; Yathiraju, 2022; Wang *et al.*, 2022). These frameworks are used by ERP vendors to develop an automated software tool grounded on an algorithm. To the best of our knowledge, the only prior research that offers an algorithm for ERP software customization in quantitative terms that can be used readily by ERP vendors is the one by Parthasarathy and Daneva (2016), namely, the prioritized requirements customization estimation (PRCE) algorithm. The PRCE algorithm did not use any AI techniques, but could still be biased by data inputs or altering the procedures specified in the algorithm at specific points during execution.

The information technology (IT) landscape of all the enterprises, irrespective of scale, has been steadily undergoing a digital transformation, especially since the advent of big data and business processes required in the majority of ERP systems. Today, as big data sets have been amassed by firms, ERP vendors are increasingly using artificial intelligence (AI), specifically machine learning (ML), to recommend ERP customization choices based on the requirements of their customer. This aids the ERP vendor's functional and technical consultants by rapidly determining the degree of ERP software customization required for the client organization in quantitative terms. Hence, the use of AI-based algorithms such as ML has become inevitable during ERP implementation. Though the AI algorithms are useful for the ERP consultants to better manage ERP software customization, they also carry some adjoining risks, such as algorithm bias. If unaddressed, such bias may have profound effect on the outcome of such algorithm, leading to the failure of a customized ERP software system. To detect and mitigate bias, one needs to use a comprehensive and systematic approach that involves evaluating the data and algorithms used in AI systems to incorporate fairness considerations into the creation and implementation.

1.1 Motivation

An algorithm is a process or group of rules that must be adhered to when calculations or other problem-solving tasks are performed, especially by a computer. The word "bias" generally refers to prejudice that is held against or in favour of one object, person, or group when compared to another, typically in a manner that is deemed unfair. The term "algorithmic bias" often refers to systematic, recurrent errors in computer algorithms that produce unjust results, such as giving one group of users more weight than another group of users. Furthermore, it occurs when a ML algorithm produces findings that are routinely biased as a result of exaggerated claims made throughout the learning process. Similar to this, "data bias" happens when specific data sources or types are purposefully or accidentally handled differently than others (Lomborg and Kapsch, 2020).

A finite, abstract, efficient, compound control structure that fulfils a particular objective according to a set of rules is what is known as an algorithm (Hill, 2016). In a ML model, the term "bias" should not be confused with algorithmic bias. Algorithmic biases may be influenced by three elements, including design bias (model, data and method), contextual bias (cultural, social and personal) and application bias (product, pricing, place and promotion) (Akter *et al.*, 2022). A biased algorithm design frequently misses causality, finds meaningless correlations or patterns and yields ambiguous results (Tsamados *et al.*, 2021). Due to the widespread use of ML-based apps to support decision-making and carry out

various ERP customization tasks, algorithmic bias has become a major concern for ERP vendors.

When ML models are used in analytics applications, a phenomenon known as "model bias" occurs that causes findings to be biased. Instead of being programmed directly, ML models are mathematical frameworks that correlate variables or features in a training data set by using statistical principles and standards (Paulus and Kent, 2020; Rozado, 2020). A ML application may produce biased results if inaccurate modelling fails to account for correlations between input features and output variables. This could have a negative effect on protected or unprotected groups (Rozado, 2020; Tsamados *et al.*, 2021). Data science models may contain a bias that is recorded in the data used to build them, which would systematically disfavour a social subgroup (Crawford, 2013).

When it comes to ML applications, method bias is defined as sources of bias caused by methodological approaches and methods used at various stages of the application's lifecycle, starting with the conceptual framework of the ML problem and proceeding through the implementation and ongoing maintenance (Walsh *et al.*, 2020). Many ML applications have poorly designed problem definitions as a result of the ML model developers' lack of familiarity with ML application development techniques, which results in unintentionally discriminating effects (Lorenzoni *et al.*, 2021). When using fabricated data, data inflating bias could be caused by the practice of balancing strong results with methodological correction (Baumgartner and Thiem, 2020).

Although algorithmic bias in ML-based models has uneven, unjust and unfair effects, research in this area is primarily anecdotal and scattered and has not yet evolved into an integrated conceptualization. In the context of ERP software projects, none of the previous studies have addressed bias in algorithms used for reasoning ERP software customization. Hence, in this paper, we strive to help practitioners involved in ERP implementation understand the algorithmic bias when they use algorithms for reasoning the degree of ERP customization. These algorithms may or may not use AI in practice, it depends on the ERP vendors and may vary from one project to another. Hence, we designed our study to address the algorithmic bias for both cases of ERP customization algorithms – one that does not use AI and one that is aided by AI.

The paper is organized as follows: In Section 2, we present the relevance of AI to ERP software solutions from a customization perspective. In Section 3, we provide a brief overview of ERP software customization, followed by our research statement in Section 4. Section 5 discusses the algorithmic bias during ERP software customization. This section is divided into two parts: Section 5.1 illustrates the possible bias when the PRCE algorithm (Parthasarathy and Daneva, 2016) available in the ERP literature is executed without any AI, whereas Section 5.2 present our newly developed AI version of the PRCE algorithm that uses ML techniques, which will be followed by an illustration. In Section 6, we draw a roadmap for managing algorithmic bias during ERP customization in practice. Section 7 concludes our research work, followed by a brief outline about the scope for future research.

2. Artificial intelligence in enterprise resource planning software solutions

Due to the evolution of AI and ML, companies are reinventing their old ERP software (Yathiraju, 2022). In this era of digital competition, customized systems are failing to deliver the desired outcomes. To scale and expand their business models, companies are searching for improved ERP software solutions. Achieving company goals does not make use of the production consistency gained from running ERP services. This happens due to rapid changes in the customer's requirements during ERP implementation. The platform will provide businesses with new opportunities and solutions in these situations by integrating

AI and ML. New ERP systems pick up knowledge progressively by observing behaviours (patterns) and by changing the default algorithm. We anticipate that the ERP models will be highly customizable and flexible. By doing this, various divisions in an organization will perform better and have better usability.

The top players in the software development industry will soon face competition from algorithms that can learn on their own. It is necessary to perform inventive programming to connect big data, or the data generated by out-dated ERP systems, with AI and ML. This would facilitate the rapid knowledge and information accumulation of the wisdom algorithms. The strategic integration of AI into management practices of executives, especially in ERP systems, is still largely unknown.

To offer the real-time data collected from production for monitoring, judgmental algorithms with tracking and traceability capabilities are needed. When creating a Web-based cloud ERP platform or a customized software solution with a self-learning system, there are significant criteria to meet. As a result, the ERP industry is seeing an increase in AI-enabled software solutions (Goundar *et al.*, 2021; Wei and Pardo, 2022; Wei *et al.*, 2022). The ERP vendors have also begun investigating the potential of using AI to choose ERP customization options during ERP implementation, going beyond just the integration of AI into ERP software solutions. As a result, they will spend less time and money trying to close the gap between the ERP software and the customer organization's business requirements.

3. Enterprise resource planning software customization

"ERP customization" is a broad term for the changes the implementation team makes to the ERP software to meet all customer needs that are not directly or indirectly met by the ERP system (Brehm, 2001; Light, 2005). The "degree of customization" refers to the amount of customer requirements (CR) that are not readily available in the chosen ERP package as a standard feature and require modification to the ERP product to meet (Parthasarathy and Sharma, 2016). Even though ERP vendors have said they will make package releases with interfaces that work for all businesses that use an ERP system, some customization work is always needed to get business processes and IT aligned at a satisfactory level. Knowing that customization is inevitable, ERP adopters must strike a delicate balance between company value and risk when determining how much customization is necessary. If customization is in line with strategy and aids the adopter in achieving its business objectives, it is regarded in ERP literature as a value-adding activity in a project.

We are aware of and grateful for the research on business information systems, specifically ERP that has been done. These studies have proposed conceptual frameworks for thinking about customization from the perspective of benefits realization and methods for both qualitatively and quantitatively estimating the benefits of customization (Luo and Strong, 2004; Ashurst *et al.*, 2008; Parthasarathy, 2008; Rothenberger and Srite, 2009; Staehr, 2010; Daneva and Wieringa, 2010; Aslam *et al.*, 2012; Eckarz *et al.*, 2012; Norton *et al.*, 2013; Doherty, 2014; Parthasarathy and Daneva, 2014; Parthasarathy and Sharma, 2014; Parthasarathy and Daneva, 2016; Parthasarathy and Sharma, 2016; Parthasarathy and Sharma, 2017; Ibrahim *et al.*, 2019; Parthasarathy *et al.*, 2020; Mahmood *et al.*, 2020; Febrianto and Soediantono, 2022; Yathiraju, 2022; Yoo and Kim, 2021; Wang *et al.*, 2022).

4. Research objective

Out of all of the preceding related research works on ERP software customization discussed in Section 3, the only one that has proposed an algorithm [namely, the PRCE ERP software customization algorithm] is the one by Parthasarathy and Daneva (2016). The rest of these earlier studies have suggested conceptual or theoretical frameworks to determine ERP

customization options for the customer's organization and ERP vendors. To the best of our knowledge, no methodological approach for understanding the bias that occurs during the execution of the algorithm (with or without AI) designed to automate the estimation of the degree to which ERP software can be customized in quantitative terms has been published up to this point.

We now state our research objective as understanding:

- Algorithmic bias in an ERP software customization algorithm, namely the PRCE algorithm (Parthasarathy and Daneva, 2016). This algorithm does not use AI.
- Algorithmic bias an in AI-enabled PRCE algorithm.

In this research, the PRCE algorithm was first assessed for possible algorithmic bias. Thereafter, the AI version of the very same algorithm was developed (Section 5.2) and assessed for algorithmic bias. Following this, we present a roadmap for managing algorithmic bias during ERP software customization based on our lessons learned while experimenting with our above outlined research objectives.

5. Understanding algorithm bias in enterprise resource planning customization

5.1 Prioritized requirements customization estimation algorithm

The PRCE method is used to calculate the level of customization necessary for the ERP software to integrate seamlessly into the customer organization and meet all of their priority needs (Parthasarathy and Daneva, 2016). The main goal of the PRCE algorithm is to predict the degree of customization needed for the ERP software package using the client's needs. According to this assessment, there is a percentage difference between the customer's requirements and those of the ERP system. The customer's needs (commonly referred to as the customer's business processes) and the ERP system itself must both be modified to implement the ERP software efficiently.

The PRCE algorithm is presented in Figure 1. For a step-by-step, detailed description of the PRCE algorithm, please refer to Parthasarathy and Daneva (2016, pp. 477-481). In Figure 1, we use iRj (reads "i is related to j") to denote the relationship "r" between two elements, say, i and j. Based on requirements rating criteria, two elements, i and j are mapped and values are entered in the corresponding cell in the requirements traceability matrix. The minimum and maximum amount of modification, in terms of quantity, required for the ERP software to function within a company, would be determined by using the PRCE algorithm. In this algorithm, Step 1 through Step 4 deal with the process of creating the application requirements (ARs), process requirements (PRs) and design requirements (DRs) for the ERP software and the customer using set theory, while other stages are intended for matching the prioritized client requirements with the requirements included in the ERP software.

In the PRCE algorithm (Parthasarathy and Daneva, 2016), the customer's requirements are broken down into three levels: the application, the process and the design. ARs are the criteria that the software must meet to satisfy the needs of the customer. PRs are the tasks or actions that must be performed to meet the ARs. A group of matching PRs is found for each AR. The requirements (design constraints) necessary to carry out the PRs are known as DRs. The criteria that must be met during the software design phase are known as the "design requirements."

For each pair of CR and available ERP features (ER), the PRCE algorithm uses an ordinal scale that goes from 0 to 6. A score of 0 means that the customer requirement is not available, while a score of six indicates that the ERP software includes the customer

Step 1 Define a set D = {d | d is an prioritized application requirement from PR-SRS}

Step 2 Define a set F = {f | f is a prioritized process requirement from PR-SRS corresponding to d, $\forall d \in D$}

Step 3 Define a set G = {g | g is a design requirement from PR-SRS corresponding to f, $\forall f \in F$}

Step 4 Construct the sets X, Y and Z similar to D, F and G respectively using the software requirements specification (SRS) of the ERP software.

Step 5 Verify D=X, F=Y and G=Z using the steps 5.1 through 5.4.

Step 5.1 Form a requirements traceability matrix $L = (a_{ij})$ (i=1 to |D|, j=1 to |X|)

$$\text{Where } a_{ij} = \begin{cases} r & \text{if iRj as per RRC and } 1 \le r \le 6 \\ 0 & \text{otherwise} \end{cases}$$

Step 5.2 As in step 5.1, form the following two requirements traceability matrices:

$$M = (b_{ij}) \text{ (i=1 to |F|, j=1 to |Y|) Where } b_{ij} = \begin{cases} r & \text{if iRj as per RRC and } 1 \le r \le 6 \text{ and there exists } a_{ij} \neq 0 \text{ for each } b_i \\ 0 & \text{otherwise} \end{cases}$$

$$N = (c_{ij}) \text{ (i= 1 to |G|, j=1 to |Z|) Where } c_{ij} = \begin{cases} r & \text{if iRj as per RRC and } 1 \le r \le 6 \text{ and there exists } b_{ij} \neq 0 \text{ for each } c_i \\ 0 & \text{otherwise} \end{cases}$$

Step 5.3 Let AV1, AV2 and AV3 denote the Customer Reqts. Availability Index for L, M and N respectively. Calculate $AV1 = (a_{11}+a_{12}+\ldots.+a_{1|X|}) + (a_{21}+a_{22}+\ldots..+a_{2|X|}) +\ldots.. + (a_{|Y|1}+a_{|Y|2}+\ldots. +a_{p|X|})$ for L. Similarly, calculate AV2 and AV3.

Step 5.4 Let AT1, AT2, AT3 denote the expected value and TV1, TV2, TV3 denote the threshold value for L, M and N respectively. Calculate AT1 = |D| x 6 and TV1 = |D| x 5 for L, AT2 = |F| x 6 and TV2 = |F| x 5 for M, AT3 = |G| x 6 and TV3 = |G| x 5 for N.

Step 6 Let A1, P1 and D1 denote the maximum degree of customization and A2, P2 and D2 denote the minimum degree of customization required to achieve D=X, F=Y and G=Z respectively. Using L, calculate A1 = ((1–(AV1/AT1))*100)% and A2 = ((1–(AV1/TV1))*100)% if (AV1/TV1)≤1, otherwise A2=0. Similarly, calculate P1, P2 using M and D1, D2 using N.

Step 6.1 Calculate the maximum degree of customization required to have customer's requirements available in ERP software as MAX = (((A1+P1+D1)/3)*100)%.

Step 6.2 Calculate the minimum degree of customization required to have customer's requirements available in ERP software as MIN = (((A2+P2+D2)/3)*100)%.

Source: Parthasarathy & Daneva (2016)

Figure 1.
PRCE ERP software customization estimation algorithm

requirement as a standard feature. The numbers 1, 2, 3, 4 and 5 denote several methods for addressing the alignment problem. Values 3 and 4 are connected to the vendor's plans to improve future releases of their ERP package in a way that will decrease the future requirements for adaptation, whilst values 1 and 2 are tied to the context of the organization that will be deploying the ERP.

Guided by the previous research (Akter *et al.*, 2022), we now present the possible design bias (model, data and method) (labelled as B1–B5) one may experience with this PRCE ERP

software customization algorithm for business gains or without any such intentions as well. It should be noted that the design bias is also a key factor contributing to algorithmic bias (Akter *et al.*, 2022). The following observed bias may lead the PRCE algorithm to be unfair to the ERP project team and the customer organization.

- *B1*: The ordinal scale ratings (0–6) are provided by the ERP project team members in the PRCE algorithm and are subjective. Hence, it may depend on the knowledge or the ignorance of these team members.
- *B2*: As per the ordinal scale ratings, the threshold value is chosen as "5" for calculating minimum degree of customization. In practice, even a lesser value may satisfy the customer's requirements in some ERP projects. By modulating this threshold value from one project to another, the PRCE algorithm may estimate an altered value for minimum degree of ERP customization.
- *B3*: Configuration management has not been considered in the PRCE algorithm for mapping the availability of a customer's requirements in ERP. Some of the customer's requirements may be met with the help of configuration settings. Accordingly, the degree of customization may scale down.
- *B4*: When mapping the customer's requirements with that of those available in the ERP, one-one matching may not always be possible. Instead, occasionally, many-to-one or one-to-many must be exercised. This is uncovered in the PRCE algorithm.
- *B5*: Some requirements may change over time when the ERP project progresses towards deployment stage. In such cases, the computation carried out by the PRCE algorithm for estimating the maximum and minimum degree of customization should be re-calculated. This is unrealistic in practice.

5.2 Machine learning algorithm for enterprise resource planning customization

AI is a branch of computer science that focusses on making smart robots that can do things such as see, hear and understand natural language (Haider, 2021). Algorithms, models and systems that learn from data and make forecasts or judgments are called AI. ML is a branch of AI that uses algorithms and models to let computers learn and develop without being programmed (Zhou, 2021). Algorithms learn from data patterns and make forecasts or choices. Data preparation, model training and model evaluation comprise ML. Data preparation gathers, processes and organizes data for model training. Model assessment determines model accuracy and efficacy. Supervised, uncontrolled and reinforcement learning are techniques used in ML (Berry *et al.*, 2019).

In supervised training, each data point has a label or goal value. The algorithm predicts labels and values for new data using patterns and connections from labelled data. Supervised ML methods include logistic regression, linear regression, decision trees and SVMs. Unsupervised learning trains the algorithm on data without labels or goal values. The algorithm recognizes patterns and correlations in the data and groups similar pieces. K-means, hierarchical, main component and association rule learning are unsupervised ML methods. Reinforcement learning lets the model learn from the world and receive rewards or punishments. The algorithm optimizes rewards and penalties. Reinforcement learning includes Q-learning and SARSA (Mahesh, 2020).

In the current research, we looked into a number of well-known supervised ML techniques, such as support vector machines, k-nearest neighbours (KNN), random forests and extreme gradient boosting. We have opted to forego deep learning models in favour of more conventional ML techniques that offer a semblance of transparency because our goal is

to study the interaction of model, data and method bias in the PRCE algorithm. A model training procedure is not necessary with the straightforward and understandable instance-based algorithm known as KNN. The majority decision is made based on the class membership of a predetermined number of closest neighbours from the training data to identify the class for a specific test sample. A similarity metric (e.g. Euclidean distance) is used to identify which neighbours are the closest to one another. Excellent results can be obtained with a sufficient data set, but the technique struggles with unbalanced data. The method is especially sensitive to the number of neighbours used to establish the class of an evaluated instance.

The KNN algorithm is a supervised learning technique that can be used for both classification and regression (Soucy and Mineau, 2001; Uddin *et al.*, 2022). It is a simple and flexible method that is often used as a building block for more complex ML models. Based on a distance metric, such as Euclidean distance, the KNN method finds the k data points that are closest to a test data point. K is a hyperparameter that must be set before the model is trained. Following the identification of the KNN, the algorithm gives a label or a numerical value to the test data point based on the majority vote (for classification) or the average value (for regression) of the labels or values of the KNN.

Thus, in this research, we have developed the AI version of the PRCE ERP customization algorithm using KNN. This is referred to as the "KNN-ERP" algorithm, which is discussed in the next section.

5.2.1 k-nearest neighbours-enterprise resource planning algorithm

In the context of ERP software customization, we now present the AI version of the PRCE algorithm called the "KNN-ERP algorithm". This algorithm uses the KNN, an ML algorithm that is widely used in AI. The objective of the "KNN-ERP" algorithm is to predict the level of ERP customization at the initial stage of ERP implementation. This algorithm equates the classes such as number of ARs, number of PRs and number of DRs, which are extracted from the ERP customer's requirements, with that of the number of ARs, PRs and DRs of previously completed ERP projects of same domain available in the project repository. For instance, assume that there are 150 ERP projects in the banking domain in the ERP vendor's project repository.

We could compute the number of ARs, PRs and DRs from the customer's requirements for our new ERP project (say, "m-Bank"). This will be mapped against the number of ARs, PRs and DRs of those 150 ERP projects and the one that comes closest to the newer ERP project's ARs, PRs and DRs (say, ERP project "iBank") will be identified. The minimum and the maximum degree of software customization undergone by the completed ERP project "iBank" can be found in the project repository documentation. From this estimation, the project manager of the newer "m-Bank" ERP project could anticipate the degree of customization that this new project would require for successful ERP implementation in no time. This is the paramount importance of the AI version of the "PRCE" algorithm, particularly the KNN-ERP algorithm.

For smaller data sets, KNN can be a good choice because it is simple to implement and interpret and can provide accurate results with minimal tuning. It is also non-parametric, which means that it can be used without making any assumptions about the underlying distribution of the data. However, as the size of the data set grows, KNN can become computationally expensive and slow. This is because the algorithm must compute the distances between each data point and every other data point in the data set to identify the KNN. Additionally, if the data set has many features, the curse of dimensionality can cause the distances between data points to become less meaningful, making the algorithm less effective (Schott, 2020). As our data set is not particularly large and doesn't possess too

many features, we chose the KNN algorithm to predict the degree of customization of ERP projects.

The step-by-step procedure of the k-nearest neighbours-enterprise resource planning algorithm is detailed below:

Step 1: collect and pre-process the data.

Extract the data set (domain, such as banking, healthcare, insurance, textiles and education) from the ERP project database. Such a database will be maintained by every ERP vendor, where the details of ERP projects will be stored for future reference. This data set will be pre-processed to make it suitable for further processing by the algorithm.

Step 2: define the features.

Select the relevant features for the KNN-ERP customization algorithm. In our current KNN-ERP customization algorithm, we chose the number of ARs, number of PRs, number of DRs, degree of customization of ARs, PRs, DRs, MIN (minimum degree of ERP customization) and MAX (maximum degree of ERP customization).

Step 3: define the similarity metric.

Choose a similarity metric to determine the distance between the new ERP project and the previous ERP projects to match the features to choose the closest ERP project so as to predict customization for the new ERP project. A common metric for KNN is the Euclidean distance.

Step 4: determine the value of K.

Decide on the value of K, which is the number of closest neighbours to consider when making a prediction. This value will depend on the size of the training set and the complexity of the problem. For this KNN-ERP algorithm, the value of K is set to 3.

Step 5: train the model.

Train the KNN algorithm on the pre-processed data to create a model that can be used to make predictions on the degree of customization required for the new ERP project. The model is trained with all eight features mentioned in Step 2.

Step 6: test the model.

Evaluate the performance of the model using a test set of data to ensure that it is accurate and robust. The test set contains the number of ARs, PRs and DRs of new ERP request. Once the model has been tested and validated, it can be used to find the degree of customization of ERP systems as per the ARs PRs and DRs of the new ERP project. The values of degree of customization of ARs, PRs, DRs, MIN and MAX will be predicted by the trained KNN-ERP model based on the previously stored data set of similar completed ERP projects stored in the project database (mentioned in step 1).

A representation of KNN-ERP algorithm is shown in Figure 2. For instance, consider that there are some projects (say, ERP Project A, ERP Project B) from a specific ERP domain (say, Banking). Each project comprises features namely the number of ARs, PRs, DRs and degree of customization for ARs, PRs, DRs, MIN and MAX. When a new data point (new ERP project) arrives with the number of ARs, DRs and PRs, these features (values) will be matched with the similar features of all ERP projects available in project repository using the Euclidean distance metric. Based on its similarity with previously customized ERP projects, the degree of customization of the new data point (new ERP project) will be predicted as shown in Figure 2 for ARs, PRs and DRs individually, as well as the minimum degree of ERP customization and the maximum degree of ERP customization required for the new ERP project as a while.

5.2.2 Algorithmic bias in k-nearest neighbours-enterprise resource planning algorithm

Several sources of bias (S1–S3) that may impact the KNN-ERP algorithm for predicting ERP customization are discussed below:

S1: data collection

Algorithmic bias can occur in the KNN-ERP algorithm when the algorithm is trained on a data set extracted from an ERP project database where there are cases of requirement incompleteness and inconsistency. This may lead to biased or erroneous predictions for the new ERP project.

S2: feature selection

The features selected for the KNN-ERP algorithm can also impact the predictions if they were chosen with prejudice. For example, if instead of matching the three key features, namely the number of ARs, PRs and DRs, with the new data point (new ERP project), we choose to execute the algorithm with just two features, namely, the number of ARs and PRs, then the prediction of degree of ERP customization for the new ERP project may be less accurate and its robustness becomes doubtful.

S3: Anonymity

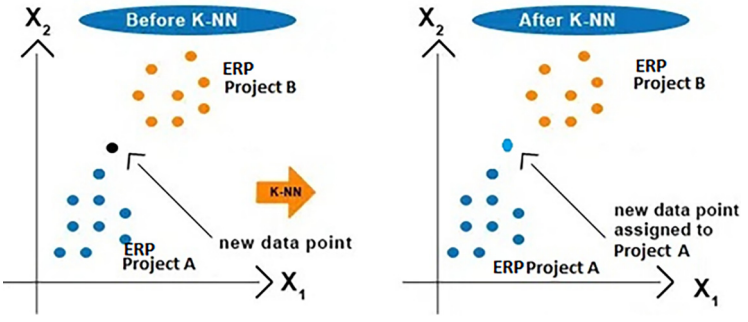
Software projects, notably AI-based projects, use anonymity, such as k-anonymity (Domingo-Ferrer *et al.*, 2021), to protect data. Anonymity removes a person's identity from data logs. For this, several competing and complementary privacy models exist. Anonymity may cause knowledge loss, which could bias ML models and algorithms (Slijepcevic *et al.*, 2021). Data distortion and knowledge loss affect the data's usefulness. Minimizing information loss is crucial for automated ML analysis that extracts significant patterns from data. As anonymity constraints increase, KNN-ERP performance usually decreases. The model's degradation depends on the ERP data set used for training and anonymization (Slijepcevic *et al.*, 2021).

S4: similarity metric bias

Euclidean distance, cosine similarity measure, Minkowsky correlation and Chi square are the most frequently used similarity measures in KNN. The prediction outcomes may also be impacted by the similarity measure selection. The most popular similarity measure, "Euclidean distance," has been used in our current KNN-ERP customization algorithm. In addition, the KNN-ERP customization method is likely to perform poorly for data sets that are not balanced. Therefore, if the algorithm is used to analyse such a data set, the results may also be biased.

6. Managing algorithmic bias in enterprise resource planning customization

The ERP project implementation team should work to comprehend and control the bias built into the ERP software customization algorithm, regardless of whether the algorithm uses AI or not. Grounded on the observations made on managing algorithmic bias in a recently published work (Townson, 2023), and also considering our understanding of ERP software



Source: Authors: Parthasarathy, S & Padmapriya, S. T.

Figure 2.
Prediction of degree of ERP customization using KNN-ERP algorithm

customization gained from our related research in this domain over the past one decade, we have identified a three-step process that can produce positive results for project managers of ERP implementations looking to either reduce or nullify the algorithmic bias when estimating the degree of ERP software customization. They are as follows:

Step 1: choice of input data set

Selecting a reasonable standard for fairness is difficult because it is impossible to be completely fair, and many decision-making groups do not yet have enough diversity. To ensure fairness in algorithms during customization, there is no method that is generally regarded as acceptable and generalizable. The selection of inputs to train the model for prediction should be made with the greatest care by the ERP implementation team. If companies want to ensure the development and test data sets that shape the algorithms are diverse enough, they need to ensure that sensitive attributes are covered and that the selection process has not skewed the data in any way. The final algorithm's fairness tests must take all aspects of ERP customization into account, including configuration management, changes in requirements over time and threshold values.

To accomplish this, the designers of the models must acknowledge the limitations of the data. The k in KNN is the number of nearest neighbours to a data point that are used in decision making (Slijepcevic *et al.*, 2021). The classifier output relies on the majority class of these neighbouring points. Data privacy methods require careful input data set selection. K-anonymity can cause knowledge loss and bias ML models (Domingo-Ferrer, 2021). ML results are hard to predict.

The classifier output relies on the majority of neighbouring points' class. Data privacy methods require careful input data set selection. The changes to ML results are hard to predict.

Step 2: examine the output

ERP vendors, with the help of implementation partners or their consultants, are supposed to check fairness in output regularly (i.e. the estimated degree of ERP software customization using an algorithm). Even when the consultants intend to predict the degree of customization in quantitative terms prior to ERP implementation, they need to underscore its adjoining risks. Bias in algorithms can result in disparate impacts on the outcome of the experiments. A two-model solution such as generative adversarial networks (GAN) is an efficient way to examine the outputs or the desired outcome of a data science project. A GAN is an ML model in which two neural networks fight with each other to improve their predictions (Aggarwal *et al.*, 2021). This makes it easier to compare the initial model to an adversary or auditor model that checks individual fairness. Both models converge on a more appropriate and fair solution.

Step 3: concurrent validation

It is crucial to regularly review outputs and validate them and look for unusual trends. Even a well-trained model that is used with inputs that change over time – in our case, changing requirements – can still produce results that are not good. If ERP consultants get results that are very different from each other, they should be taught how to spot bias in customization algorithms. When creating AI-enabled ERP customization, it is possible to unintentionally perpetuate bias. As one would usually anticipate from a brain built to simplify and generalize patterns, rare events are in fact unlikely, but not impossible. ERP vendors must consciously make up for unfairness if they genuinely want their algorithms to operate fairly across a diverse populace. It is impossible for algorithm designers working with ERP

customization vendors to fully eradicate bias. However, they can improve, broaden, examine and make necessary adjustments to their procedures to produce more justice as well as varied and equitable results.

7. Conclusions and future research

In this paper, we introduced the concept of algorithmic bias and discussed the design bias (model, data and method) in an algorithm meant for ERP software customization. In this context, we also developed an AI version of the PRCE ERP software customization algorithm and elucidated the probable algorithmic bias that might affect the outcome of the algorithm. We have mined possible bias for the PRCE ERP customization algorithm for both the cases – the one that is used as without any AI techniques and another one that used the KNN algorithm, a ML technique. The results of this research offer insight into the factors that contribute to design biases in an algorithm. Following this, we have also presented a three-step approach to help the ERP implementation team members manage the bias dynamically. Thus, we deem that this paper has addressed algorithmic bias to facilitate fair practices in ERP customization algorithms to prevent discriminatory and unfair outcomes.

In the era of AI, ERP project managers must have the skills, abilities and technological understanding necessary to properly investigate ways to better handle the adoption of ML applications during customization and exploit them in the right ways to produce fair results and maximize organizational performance. To prepare ERP implementation to address the underlying issues related to algorithmic bias, this paper paves the way for the application of dynamic managerial skills within ML-based applications.

7.1 Implications for research

From a theoretical standpoint, our study suggests that more investigation is required to address the algorithmic bias in the ERP customization algorithm when analysing the relationship between customization requirements and the features provided in the chosen ERP system. This connection, which has been researched from the perspective of IT and business alignment, is now revealed by studies using frameworks, conceptual models or algorithms. Previous research papers have recommended that algorithmic bias be given more attention by academics and industry professionals. The possibility of algorithmic prejudice during the ERP customization process has not, however, been studied to date.

The research that we are conducting right now goes beyond these previously documented approaches in a number of different ways. First, we take our own PRCE ERP customization algorithm (Parthasarathy and Daneva, 2016) and discuss the algorithmic bias from a design perspective (model, data and method) (Akter *et al.*, 2022). Further, we have also developed an AI version of the same algorithm using a ML technique, the KNN algorithm and thereafter discussed the algorithmic bias in this newer version. Next, a growing number of ERP projects today are linked to fast change in the ERP industry. As a consequence, we have arrived at the conclusion that future research should concentrate on finding ways to improve the efficiency with which the existing algorithm can manage complex and shifting requirements, as well as organize and prioritize them at any given moment. Therefore, determining whether the strategy can actually be implemented in a large-scale project setting should be an essential research priority. This research not only helps ERP project managers understand algorithmic bias but also suggests a three-step approach to either nullify it or reduce it reasonably before they make decisions on the customization choices based on such algorithms.

Naturally, a firm would examine a variety of factors to include in their decision-making process, with the customization estimation being only one of them. While it is true that the

algorithm will be effective, we believe that to properly compare packages, more sophisticated tool assistance is required. A promising avenue of research for the future would be to investigate the various possibilities pertaining to this matter and conduct case studies to gain an understanding of the type of automation that would be most effective for such comparisons.

7.2 Implications for practice

The minimum and maximum levels of customization will be estimated in numerical terms. This will help ERP consultants discuss available customization options openly with their customers. One could, for example, agree to help with the customization work or try to change the business processes of the organization so that they work better with the ERP software. Both ERP consultants and their customers will benefit in the long term from understanding and managing bias in ERP customization algorithms. This is because customers could use this as a logical starting point to negotiate their needs. Thus, we think that it will definitely pay off in the long term to understand and deal with bias in ERP customization algorithms. Even though it may be difficult to obtain objective values on the impact of algorithmic bias when evaluating the degree to which ERP can be customized, our research indicates that practicing project managers should try to think about it anyway because doing so will pay off in the long term. In conclusion, our research provides ERP consultants with a road map that can assist them in better grounding their recommendations for customization on their explicit knowledge about algorithmic bias and its influence over the results produced by such algorithms.

References

Aggarwal, A., Mittal, M. and Battineni, G. (2021), "Generative adversarial network: an overview of theory and applications", *International Journal of Information Management Data Insights*, Vol. 1 No. 1, p. 100004.

Akter, S., Dwivedi, Y.K., Sajib, S., Biswas, K., Bandara, R.J. and Michael, K. (2022), "Algorithmic bias in machine learning-based marketing models", *Journal of Business Research*, Vol. 144, pp. 201-216.

Ashurst, C., Doherty, N.F. and Peppard, J. (2008), "Improving the impact of IT development projects: the benefits realization capability model", *European Journal of Information Systems*, Vol. 17 No. 4, pp. 352-370.

Aslam, U., Coombs, C. and Doherty, N. (2012), "Benefits realization from ERP systems: the role of customization".

Baumgartner, M. and Thiem, A. (2020), "Often trusted but never (properly) tested: evaluating qualitative comparative analysis", *Sociological Methods and Research*, Vol. 49 No. 2, pp. 279-311.

Berry, M.W., Mohamed, A. and Yap, B.W. (Eds) (2019), *Supervised and Unsupervised Learning for Data Science*, Springer Nature.

Brehm, L., Heinzl, A. and Markus, M.L. (2001), "Tailoring ERP systems: a spectrum of choices and their implications", *Proceedings of the 34th annual HI International Conference on System Sciences*, *IEEE*, p. 9.

Crawford, J.T., Jussim, L., Cain, T.R. and Cohen, F. (2013), "Right-wing authoritarianism and social dominance orientation differentially predict biased evaluations of media reports", *Journal of Applied Social Psychology*, Vol. 43 No. 1, pp. 163-174.

Daneva, M. and Wieringa, R. (2010), "Requirements engineering for enterprise systems: what we know and what we don't know?", In Intentional perspectives on information systems engineering, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 115-136.

Doherty, N.F. (2014), "The role of socio-technical principles in leveraging meaningful benefits from IT investments", *Applied Ergonomics*, Vol. 45 No. 2, pp. 181-187.

Domingo-Ferrer, J., Sánchez, D. and Blanco-Justicia, A. (2021), "The limits of differential privacy (and its misuse in data release and machine learning)", *Communications of the ACM*, Vol. 64 No. 7, pp. 33-35.

Febrianto, T. and Soediantono, D. (2022), "Enterprise resource planning (ERP) and implementation suggestion to the defense industry: a literature review", *Journal of Industrial Engineering and Management Research*, Vol. 3 No. 3, pp. 1-16.

Goundar, S., Nayyar, A., Maharaj, M., Ratnam, K. and Prasad, S. (2021), "How artificial intelligence is transforming the ERP systems", *Enterprise Systems and Technological Convergence: Research and Practice*, Vol. 85.

Haider, L. (2021), "Artificial intelligence in ERP".

Hill, C., Miller, K., Benson, K. and Handley, G. (2016), *Barriers and Bias: The Status of Women in Leadership*, American Association of University Women.

Ibrahim, S.H., Duraisamy, S. and Sridevi, U.K. (2019), "Flexible and reliable ERP project customization framework to improve user satisfaction level", *Cluster Computing*, Vol. 22 No. S2, pp. 2889-2895.

Light, B. (2005), "Going beyond 'misfit' as a reason for ERP package customisation", *Computers in Industry*, Vol. 56 No. 6, pp. 606-619.

Lomborg, S. and Kapsch, P.H. (2020), "Decoding algorithms", *Media, Culture and Society*, Vol. 42 No. 5, pp. 745-761.

Lorenzoni, A., Santesteban, M., Peressotti, F., Baus, C. and Navarrete, E. (2021), "Dimensions of social categorization: inside the role of language", *Plos One*, Vol. 16 No. 7, p. e0254513.

Luo, W. and Strong, D.M. (2004), "A framework for evaluating ERP implementation choices", *IEEE Transactions on Engineering Management*, Vol. 51 No. 3, pp. 322-333.

Mahesh, B. (2020), "Machine learning algorithms-a review", *International Journal of Science and Research*, Vol. 9, pp. 381-386.

Mahmood, F., Khan, A.Z. and Bokhari, R.H. (2020), "ERP issues and challenges: a research synthesis", *Kybernetes*, Vol. 49 No. 3, pp. 629-659.

Norton, L.A., May Coulson-Thomas, Y., Coulson-Thomas, C.J. and Ashurst, C. (2013), "Ensuring benefits realisation from ERP II: the CSF phasing model", *Journal of Enterprise Information Management*, Vol. 26 No. 3, pp. 218-234.

Parthasarathy, S. and Daneva, M. (2014), "Customer requirements based ERP customization using AHP technique", *Business Process Management Journal*, Vol. 20 No. 5, pp. 730-751.

Parthasarathy, S. and Daneva, M. (2016), "An approach to estimation of degree of customization for ERP implementation using prioritized requirements", *Journal of Systems and Software*, Vol. 117, pp. 471-487.

Parthasarathy, S. and Sharma, S. (2014), "Determining ERP customization choices using nominal group technique and analytical hierarchy process", *Computers in Industry*, Vol. 65 No. 6, pp. 1009-1017.

Parthasarathy, S. and Sharma, S. (2016), "Efficiency analysis of ERP packages-Customization perspective", *Computers in Industry*, Vol. 82, pp. 19-27.

Parthasarathy, S. and Sharma, S. (2017), "Impact of customization over software quality in ERP projects–an empirical study", *Software Quality Journal*, Vol. 25 No. 2, pp. 581-598.

Parthasarathy, S., Sridharan, C., Chandrakumar, T. and Sridevi, S. (2020), "Quality assessment of standard and customized COTS products", *International Journal of Information Technology Project Management*, Vol. 11 No. 3, pp. 1-13.

Paulus, J.K. and Kent, D.M. (2020), "Predictably unequal: understanding and addressing concerns that algorithmic clinical prediction may increase health disparities", *NPJ Digital Medicine*, Vol. 3 No. 1, p. 99.

Rothenberger, M.A. and Srite, M. (2009), "An investigation of customization in ERP system implementations", *IEEE Transactions on Engineering Management*, Vol. 56 No. 4, pp. 663-676.

Rozado, D. (2020), "Wide range screening of algorithmic bias in word embedding models using large sentiment lexicons reveals underreported bias types", *Plos One*, Vol. 15 No. 4, p. e0231189.

Schott, M. (2020), "K-nearest neighbors (knn) algorithm for machine learning", In Capital One Tech.

Slijepcevic, D., Henzl, M., Klausner, L.D., Dam, T., Kieseberg, P. and Zeppelzauer, M. (2021), "k-Anonymity in practice: How generalisation and suppression affect machine learning classifiers", *Computers and Security*, Vol. 111, p. 102488.

Soucy, P. and Mineau, G.W. (2001), "A simple KNN algorithm for text categorization", "I", n *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 647-648, *IEEE*.

Staehr, L. (2010), "Understanding the role of managerial agency in achieving business benefits from ERP systems", *Information Systems Journal*, Vol. 20 No. 3, pp. 213-238.

Townson, S. (2023), "Manage AI bias instead of trying to eliminate it", *MIT Sloan Management Review*, pp. 1-4.

Tsamados, A., Aggarwal, N., Cowls, J., Morley, J., Roberts, H., Taddeo, M. and Floridi, L. (2021), "The ethics of algorithms: key problems and solutions", Ethics, Governance, and Policies in Artificial Intelligence, pp. 97-123.

Uddin, S., Haque, I., Lu, H., Moni, M.A. and Gide, E. (2022), "Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction", *Scientific Reports*, Vol. 12 No. 1, pp. 1-11.

Walsh, C.G., Chaudhry, B., Dua, P., Goodman, K.W., Kaplan, B., Kavuluru, R. and Subbian, V. (2020), "Stigma, biomarkers, and algorithmic bias: recommendations for precision behavioral health with artificial intelligence", *JAMIA Open*, Vol. 3 No. 1, pp. 9-15.

Wang, W.T., Luo, M.C. and Chang, Y.M. (2022), "Exploring the relationship between conflict management and transformational leadership behaviors for the success of ERP customization", *Information Systems Management*, Vol. 39 No. 2, pp. 177-200.

Wei, H., Oei, T.P. and Zhou, R. (2022), "Test anxiety impairs inhibitory control processes in a performance evaluation threat situation: evidence from ERP", *Biological Psychology*, Vol. 168, p. 108241.

Wei, R. and Pardo, C. (2022), "Artificial intelligence and SMEs: how can B2B SMEs leverage AI platforms to integrate AI technologies?", *Industrial Marketing Management*, Vol. 107, pp. 466-483.

Yathiraju, N. (2022), "Investigating the use of an artificial intelligence model in an ERP cloud-based system", *International Journal of Electrical, Electronics and Computers*, Vol. 7 No. 2, pp. 1-26.

Yoo, B.K. and Kim, S.H. (2021), "Analysis of impact on ERP customization module using CSR data", *Journal of Information Processing Systems*, Vol. 17 No. 3, pp. 473-488.

Zach, O., Munkvold, B.E. and Olsen, D.H. (2014), "ERP system implementation in SMEs: exploring the influences of the SME context", *Enterprise Information Systems*, Vol. 8 No. 2, pp. 309-335.

Zhou, Z.H. (2021), *Machine Learning*, Springer Nature.

Further reading

Parthasarathy, S. and Anbazhagan, N. (2008), "Evaluating ERP projects using DEA and regression analysis", *International Journal of Business Information Systems*, Vol. 3 No. 2, pp. 140-157.

Corresponding author
Sudhaman Parthasarathy can be contacted at: parthatce@gmail.com